SESSION

APPLICATIONS OF BIG DATA + SOFTWARE TOOLS

Chair(s)

TBA

Application-Agnostic Streaming Bayesian Inference via Apache Storm

T. Wasson[1] **and A. P. Sales**[1]
[1]Data Analytics and Decision Sciences, Lawrence Livermore National Laboratory, Livermore, CA, USA

Abstract— *Given the increasing rates of data generation, along with increasingly ubiquitous sensors to measure it, analytical capabilities must be developed to keep pace. Although existing techniques have had success with some machine learning and statistical inferential tasks, these tasks are generally either limited in scope or are not capable of inferring the full probability densities (necessary for many sophisticated statistical analytical approaches) at a fast enough rate in order to match that of data arrival.*

We present a scalable framework for performing statistical density estimation on-the-fly on streams of data. This is achieve via a parallelized Apache Storm implementation of a particle learning algorithm. We demonstrate how to construct such an approach from the Storm primitives, and build upon these with novel contributions to Storm. Importantly, although our approach is exemplified via our particle learning framework, the ideas herein are generically applicable and agnostic of underlying modeling choices.

Source code available for Storm extensions upon request. Contact `wasson3@llnl.gov`.

Keywords: streaming data processing; density estimation; online statistical inference; Apache Storm; Bayesian statistics

1. Introduction

In virtually every domain of science, advances in data collection technologies have been greatly outpacing advances in the development of capabilities to satisfactorily store, process, and analyze such large volumes of data. There exists a consensus that there is a great deal to be learned and gained from these data, but how to make sense and extract meaningful information from these data streams of ever increasing volume and complexity remains a major challenge. While batch techniques for retrospective analysis, like Hadoop, Spark, and others, have resulted in powerful nuanced analytical approaches [1, 2], many data sources are too voluminous and arrive at too high a rate to be stored and post-processed via batch approaches and must be processed on-the-fly by streaming tools. Numerous techniques and software packages exist for performing a variety of machine learning and statistical inference tasks on streaming data [3, 4, 5, 6], but these are generally limited to relatively focused (albeit extremely useful) methods, such as classification via decision trees, support vector machines, or random forests, clustering, and regression techniques.

We present an implementation of a Bayesian density estimation technique via the stream processing framework Apache Storm, which can perform inference on-the-fly on data streams. Density estimation is performed via a sequential Monte Carlo algorithm, namely particle learning, and can be used in order to accomplish useful tasks such as clustering,classification, anomaly detection, and drift detection. The sequential nature of this algorithm allows the statistical model to be updated with each observations that arrives in the data stream, such that data need not be stored for batch processing. The Storm implementation allows for parallel computations within the streaming framework, substantially easing the computational burden entailed by these types of sophisticated statistical modeling. Finally, we also present novel extensions to Storm itself to facilitate this implementation. Altogether, we have developed a substantially powerful tool for performing sophisticated statistical inference on streaming data, allowing novel analyses and providing novel Storm additions.

The remainder of this article is structured as follows. We describe Storm and its utility in Section 2.1. In Section 2.2, we describe ParticleStorm, with the statistical model being described in Section 2.2.1, the capabilities and functionalities of the individual components of ParticleStorm topology being described in Section 2.2.2, and how these components work together to form a cohesive whole in Section 2.2.3. Our contributions to the Storm project are listed in Section 3, and our final remarks are given in Section 4.

2. Approach

2.1 Apache Storm

Apache Storm [7] is a distributed fault-tolerant real-time stream processing framework. Using Storm, arbitrary event-based functions can be calculated on-the-fly on inbound data streams, with the calculations spread across compute clusters, including easy deployment on cloud computing frameworks such as Amazon Web Services. Storm has been used effectively for a broad variety of applications [8], including various data analytics, machine learning tasks, and continuous computation tasks.

Storm includes a collection of fundamental software and terminology abstractions necessary to describe and implement its capabilities. A complete computational system in Storm is a directed graph, called a *topology*. The nodes of

the topology are *spouts*, which emit data, and *bolts*, which ingest and perform computations on data, optionally emitting results of these computations downstream to other bolts. Data and other communications within a topology are carried within *streams* of *tuples*. Streams may be subscribed to by any bolt, except for *direct streams*, in which the producer can decide explicitly which bolt is to receive a given tuple. Spouts and bolts may produce, and bolts may consume, any number of streams. Tuples are vectors of arbitrarily-typed elements, which may contain data or its derivations, or may contain messages important for the control schemes overlaid on topologies to enforce fault tolerance, exactly-once processing, and other necessary overhead important to a resilient processing framework.

Storm is written in Java and Clojure, and hence designed to run in Java Virtual Machines, but it supports a protocol for communication with external processes via its multi-language (*multilang*) protocol. Bolts that communicate with external processes for data processing via the multilang protocol are called *ShellBolts*, and will be discussed further in Section 2.2.2.

Recent development in Storm has been primarily focused on *Trident*, an abstraction allowing implementation of many commonly-desired use cases with dramatically less work necessary for the developer. However, Trident imposes restrictions to achieve these benefits, including removing the explicit definition of bolts and their stream subscriptions, and hence the ability to enforce cycles and build nuanced control schemas of the user's design. As such, Trident is eminently practical for the majority of tasks, but insufficient for some complex situations such as those discussed below.

Storm is one of many stream processing frameworks presently available, including Apache S4, Samza, Spark Streaming, and others. We pursued development with Storm because it allowed the finest-grained control of processing unit (bolt) heterogeneity, the greatest control over data streams (specifically, allowing arbitrary communications within the topology), and maturity and support in the community.

2.2 Storm for density estimation

Density estimation is a particularly powerful technique for learning from unstructured data, and we discuss here how it is carried out via particle learning, followed by a description of how particle learning can be implemented in Storm.

2.2.1 Streaming density estimation using particle learning

Consider a state-space model that is evolving over time, where the true underlying model state, x, is unobservable and information about it is only obtained via noisy measurements, y, at each time step, t. The state vector at time t, x_t, given all observed measurements up to that time step, $y_{1:t}$, can be estimated via its *filtering distribution*:

$$p(x_t|y_{1:t}) = \frac{p(y_t|x_t)p(x_t|y_{1:t-1})}{\int p(y_t|x_t)p(x_t|y_{1:t-1})dx_t}, \tag{1}$$

where the *predictive distribution* of state x_t given the observed measurements up to the previous time step, $y_{1:t-1}$, is given by

$$p(x_t|y_{1:t-1}) = \int p(x_t|x_{t-1})p(x_{t-1}|y_{1:t-1})dx_{t-1}. \tag{2}$$

In effect, Equation (1) is a simple application of Bayes theorem, where the predictive distribution $p(x_t|y_{1:t-1})$ is treated as the prior for x_t before the arrival of measurement y_t. In most cases, Equations (1) and (2) are analytically intractable, but can be approximated by particle filtering.

Particle filtering [9] is a sequential Monte Carlo method in which the current state variable is estimated by a weighted average of a set of random i.i.d. samples, called *particles*, from the state variables obtained by its filtering density given in Equation (1)[1]. As the number of particles increases, the particle filter approximation converges to the actual distribution.

Let $\{x_t^{(i)}\}_{i=1}^N$ be a set of N particles generated from the filtering distribution. The predictive distribution can be approximated by

$$p(x_t|y_{1:t-1}) \simeq \frac{1}{N}\sum_{i=1}^{N}\delta_{x_t^{(i)}}, \tag{3}$$

where $\delta_{x_t^{(i)}}$ is the Dirac delta function centered at $x_t^{(i)}$. By substituting (3) into (1), the filtering distribution can be approximated via a discretization of $p(x_t|y_{1:t})$ into particles $\{x_t^{(i)}\}_{i=1}^N$ with probabilities $\{w_t^{(i)}\}_{i=1}^N$,

$$\begin{aligned} p(x_t|y_{1:t}) &\simeq \sum_{i=1}^{N} w_t^{(i)}\delta_{x_t^{(i)}}, \\ w_t^{(i)} &= \frac{p(y_t|x_t^{(i)})}{\sum_{i=1}^{N} p(y_t|x_t^{(i)})} \end{aligned} \tag{4}$$

Particle filters operate simply by iterating between (3) and (4) at each time step with the arrival of new observations. A common shortcoming of particle filters is that the weights of particles in regions of high posterior density steadily increase to the point that eventually a single particle dominates the filter, and the weights of all other particles become negligible [10]. This so-called degeneracy problem can be directly quantified via the effective sample size of the particle set,

$$N_{\text{eff}} = \frac{N}{1 + \text{Var}(\{w_t^{(i)}\}_{i=1}^N)}, \tag{5}$$

such that the smaller the effective sample size of a filter, the more severe the degeneracy problem. A brute force solution

[1]To be precise, particle filtering algorithms entail numerical approximations of the joint posterior distribution $p(x_{1:n}|y_{1:n})$. Equation (1) shows only the marginal $p(x_n|y_{1:n})$ for simplicity.

to this problem is to use a very large N, but this leads to prohibitively large computational burdens. A more effective approach entails eliminating particles with small weight via a *resampling* step.

The resampling step stochastically enriches the particle set with high-importance particles, by eliminating particles with small importance weights. It works by sampling N particles with replacement from the set of particles $x_t^{(i)}$ according to their respective weights. The weights of the new generation of particles is set to $1/N$. Samples with large weights are likely to be drawn multiple times, whereas those with small weights are likely to be drawn very few times or not at all. Thus, particle filtering can be seen as a type of *survival of the fittest* algorithm, where higher weight particles are likely to produce more "offspring."

While resampling attenuates the degeneracy problem, it may lead to sample impoverishment. That is, because high-weight particles are likely to be drawn multiple times, over time the diversity of the samples is drastically reduced. Sample impoverishment is detrimental to the filter accuracy, as it results in worse approximation of the state distribution. Liu and Chen (1998) [11] provide a detailed discussion of the merits of resampling.

Particle learning (PL) [12] is a type of particle filtering algorithm that overcomes both the degeneracy and the sample impoverishment drawbacks of common particle filters. In fact, Carvalho *et al.* (2010) [12] demonstrated that PL's accuracy is not only superior to standard particle filter algorithms, but is comparable to MCMC samplers. PL improves upon traditional particle filtering algorithms in two ways:
Conditional sufficient statistics, s, are used to represent the posterior distribution of unknown parameters, θ, which is learned (hence, particle learning) as new observations arrive. The particles now are represented by $\{z_t^{(i)} = (x_t^{(i)}, s_t^{(i)}, \theta_t^{(i)})\}_{i=1}^N$, which are generated from the predictive distribution $p(x_t, \theta | y_{1:t-1})$ (likewise, $\widetilde{z}_t$ is sampled from the filtering distribution $p(x_t, \theta | y_{1:t})$.
The Resample-Propagate algorithm (shown in Table 1) is used in order to obtain exact samples from the particle approximation. Performing resample first and propagate second reduces approximation errors, because states are only propagated after being informed by the new observation t_{t+1}. Hence, only "good" particles are propagated.

We use the PL algorithm for composite mixture models, where each mixture component is a composite of independent distributions for each element of the response and predictor arrays. This approach enables modeling of data that includes multiple disparate feature types into a single probability model without resorting to complicated embeddings that would preclude sequential analysis. This model is described in detail in Sales *et al.* (2013) [13].

Table 1: Particle learning algorithm. Initialization is performed once, and steps 1 through 3 are repeated for each observation that arrives in the data stream.

Step	Task	Description
Step 0	Initialization	Set the starting values of the N particles $\{z_t^{(i)} = (x_t^{(i)}, \theta_t^{(i)})\}_{i=1}^N$
Step 1	Evaluation	Evaluate new observation y_{t+1} under the current model, $p(y_{t+1} \mid z_t^{(i)})$
Step 2	Resample	Resample $\{\widetilde{z}_t^{(i)}\}_{i=1}^N$ with weight $w_t \propto p(y_{t+1} \mid z_t^{(i)})$
Step 3	Propagate	$z_t^{(i)}$ from $p(z_t^{(i)} \mid \widetilde{z}_t^{(i)})$

2.2.2 Storm implementation

We have implemented streaming density estimation via particle learning in Storm to yield a tool entitled ParticleStorm. ParticleStorm functions in either inference mode, in which model parameters are updated after each data point is processed and resampling may or may not occur (though propagation always does), or evaluation mode, in which all parameters are fixed and data points are evaluated as quickly as possible under the current ensemble of models. Because particle learning in inference mode requires model parameter updates, and those updates require knowledge gained from the entire ensemble en masse, direct asynchronous communication between some (but not all) bolts is a necessity. Indeed, ParticleStorm has a number of characteristics that make it somewhat different from the majority of Storm frameworks, and in aggregate preclude the use of Trident. Specifically:

- ParticleStorm relies on an external C++ executable implementation of particle learning, entitled PF, modified to perform task-specific functions within the larger distributed Storm topology.
- ParticleStorm requires exactly-once computation of data points.
- ParticleStorm requires that between data points, parameters of some bolts are updated and possibly retrieved or overwritten.
- ParticleStorm must be switchable between inference and evaluation mode on-the-fly via an external control mechanism

Hence, ParticleStorm is constructed from the base spout and bolt abstractions.

It is important to note that Storm is quite complex, but much of that complexity is usually hidden from the developer, particularly for common or straightforward applications. For the implementation of ParticleStorm, we extended Storm to accommodate desirable properties, such as synchronicity and exactly-once data processing. Our extensions make heavy use of Storm's CoordinatedBolt class and other aspects of transactional topologies [14]. These include a number of control mechanisms, in terms of additional bolts and streams, to facilitate guaranteed exactly-once message

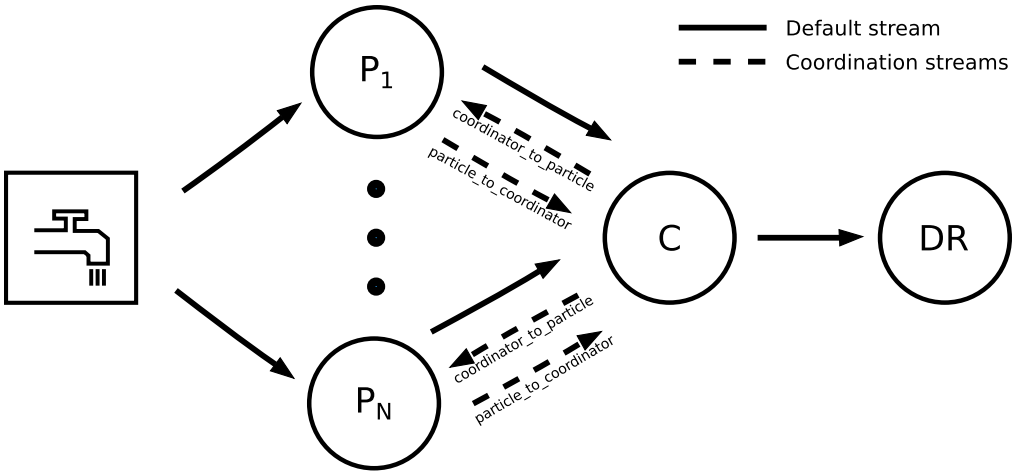


Fig. 1: ParticleStorm Storm topology. ParticleStorm is composed of a spout, a model-specific number of N ParticleBolts with one ParticleBolt per particle, a CoordinatorBolt, and a DataRecorder. The `default` stream flows through the entirety of the topology, carrying first data points and then log likelihoods derived from those data points. Two coordination streams connect each ParticleBolt to the CoordinatorBolt. Tuples of data are generated by the spout and passed to all ParticleBolts, where they are evaluated under each particle. The log likelihoods of the data points are passed on `default` to the CoordinatorBolt, which combines them and in turn passes the complete model log likelihood along to the DataRecorder, also on `default`. The coordination streams `coordinator_to_particle` and `particle_to_coordinator` are used to allow the CoordinatorBolt to communicate selectively with each ParticleBolt, performing operations including sending and retrieving model parameters during inference. Walkthroughs of typical processing runs are given in more detail in Section 2.2.3.

passing. For example, usage of CoordinatedBolts implies the creation of a wrapper CoordinatedBolt and its "delegate" bolt, being the bolt of the developer's design. The CoordinatedBolt intercepts communications to and from the delegate and, along with its own use of coordination tuples passed on implicitly-created coordination streams, tracks how many inbound tuples a delegate can expect to receive from its upstream bolts before it can be confident in having received all tuples exactly once.

Additionally, ParticleStorm relies on transactional spouts, necessary to ensure that an entire 'batch' of tuples is processed exactly once and to allow potential replaying of tuples upon failure, which transparently create separate spout coordinator and spout emitter tasks, and also use special-purpose batch initialization and commit tuples. Storm topologies are transparently augmented with an 'acker' bolt to handle acknowledgments of tuples and hence be able to trigger proper replaying of a data tuple if a descendant tuple fails. We explicitly acknowledge the importance and relevance of these usually-hidden complexities, but going forward, we choose to set them aside and focus on the explicit portions of our topology in subsequent descriptions and figures.

The (explicit) ParticleStorm topology (Figure 1) is composed of a spout, one or more ParticleBolts, a CoordinatorBolt, and a DataRecorder bolt, interconnected by several communications streams and one data stream. We describe the components of the topology here, and give examples of its functionality in Section 2.2.3 to illustrate how the components operate in concert to perform inference and evaluation.

Streams: ParticleStorm has three types of streams: `default`, `coordinator_to_particle`, and `particle_to_coordinator`, which function as follows.

- `default` carries data observations, represented as pipe-delimited feature vectors, and calculations derived from the data observations.
- `coordinator_to_particle` is a direct stream by which the CoordinatorBolt may communicate with ParticleBolts. This stream carries issued commands from the CoordinatorBolt, with replies expected to arrive on `particle_to_coordinator`. Tuples on this stream have three fields, and are of the form `[ID, command, content]`. `ID` is a unique identifier used to track responses to this command. `command` is one of `requestParameters`, `assignParameters` or `propagate`. When `command` is `requestParameters` or `propagate`, `content` is empty. When `command` is `assignParameters`, `content` contains the model parameters to be assigned to the destination ParticleBolt.
- `particle_to_coordinator` is a direct stream by which ParticleBolts may communicate with the CoordinatorBolt. This stream carries replies from ParticleBolts to the CoordinatorBolt, in response to commands issued on `coordinator_to_particle`. Tuples on this stream have two fields, and are of the form `[ID, content]`. As with `coordinator_to_particle`, `ID` is a unique identifier, and is identical to the `ID` in the `command` to which this tuple is responding. `content` is the response to the specific `command` received. When `command` is `assignParameters` or `propagate`, `content` is `ack`. When `command` is `requestParameters`, `content` contains the model parameters of this ParticleBolt.

ParticleStorm spout: The spout in ParticleStorm can be any TransactionalSpout [14], but must also incorporate the Storm Signals framework [15]. Storm Signals allow asynchronous communication with the spout outside of the traditional streams / tuples mechanism. This communication allows the spout to be paused and resumed as necessary, which is important during initialization and inference (described in Section 2.2.3), and allows a user to selectively pause the processing in the topology.

ParticleBolts: ParticleBolts are the fundamental source of parallelism employed in ParticleStorm. Each ParticleBolts hosts one particle in the particle learning model, and hence the number of ParticleBolts, equal to the number of particles, is model-dependent and determined at runtime. ParticleBolts extend ShellBolts, as the underlying modeling is done in the external PF binary executable. ParticleBolts subscribes to the `default` stream from the spout and `coordinator_to_particle` stream from the CoordinatorBolt, and outputs `default` and `particle_to_coordinator` streams. Data tuples received on `default` are scored under the modeled particle, and log likelihoods are subsequently emitted on `default`. Commands from the CoordinatorBolt are received on `coordinator_to_particle` and responses to those commands are emitted on `particle_to_coordinator`.

CoordinatorBolt: The CoordinatorBolt is the driver of ParticleStorm. There is exactly one CoordinatorBolt in the ParticleStorm topology, and it is responsible for tasking ParticleBolts to perform operations along with interpreting their output. Like ParticleBolts, the CoordinatorBolt extends ShellBolt and delegates work to PF. The CoordinatorBolt subscribes to the `default` and `particle_to_coordinator` streams from each ParticleBolt, and outputs `default` and `coordinator_to_particle` streams. In both evaluation and inference mode, the CoordinatorBolt receives log likelihoods from all ParticleBolts, weighing them appropriately to produce an overall model log likelihood for a data point, and emits that output. In inference mode, the CoordinatorBolt will determine whether a resample step is necessary. If so, the new vector of particles is sampled, and lists are created of particles to be overwritten and particles to provide parameters to do the overwriting. The CoordinatorBolt will emit a `requestParameters` command to each 'overwriting' particle, received the particle's parameters on `particle_to_coordinator`, and then emit an `assignParameters` command to overwrite the appropriate particle and await acknowledgment (indicating success), at which point the resample step is complete. ParticleBolts will then be tasked to update their parameters via a `propagate` command.

DataRecorder: The DataRecorder is responsible for processing, and potentially saving, the output of the data evaluation. It subscribes to the `default` stream from the CoordinatorBolt, which provides the log likelihood of each data point evaluated under the entire model. Our DataRecorder bolt can be set to either save results in an HDFS store or discard them and instead save the trained models.

2.2.3 ParticleStorm runtime modes and descriptions

As discussed previous, ParticleStorm operates in two discrete modes, being evaluation and inference. The two modes are largely identical in practice, with the primary differences being that in inference mode, resampling and propagation steps are included after each data point is evaluated.

As we describe the functionality of ParticleStorm, we make two simplifying choices for clarification: first, common underlying tuple chatter present in all Storm transactional topologies is omitted, and second, we present the steps in the algorithm as if they were sequential. Importantly, Storm is quite asynchronous, and handling this asynchrony in a way that is fault-tolerant and dependable is nontrivial. Indeed, correctly handling asynchrony was the motivation for much of our novel additions to Storm itself, discussed later in Section 3.

We describe evaluation and inference modes together in Table 5. First, all components take part in the Initialize process, described in Table 4. This process is composed of the CoordinatorBolt initializing the particle learning model, either from a given initial model or *de novo*, and transmitting the appropriate model parameters to each ParticleBolt via the AssignParameters process (Table 2). AssignParameters and RequestParameters (Table 3) are

Table 2: **AssignParameters** in ParticleStorm. Parameter assigning scheme. This occurs in initialization and inference mode.

Component	Action
CoordinatorBolt	**Emit direct** parameters on stream **`coordinator_to_particle`** to **ParticleBolt**
CoordinatorBolt	Wait for `ack`
ParticleBolt	**Receive** parameters on stream **`coordinator_to_particle`**
ParticleBolt	Set parameters
ParticleBolt	**Emit direct** `ack` on stream **`particle_to_coordinator`** to **CoordinatorBolt**
CoordinatorBolt	**Receive** `ack` on stream **`particle_to_coordinator`** from **ParticleBolt**
CoordinatorBolt	End wait

Table 3: **RequestParameters** in ParticleStorm. Parameter requesting scheme. This occurs in inference mode and optionally when saving models.

Component	Action
CoordinatorBolt	**Emit direct** `requestParameters` on stream **`coordinator_to_particle`** to **each ParticleBolt**
CoordinatorBolt	Wait for `parameters`
ParticleBolts	**Receive** `requestParameters` on stream **`coordinator_to_particle`**
ParticleBolts	**Emit direct** `parameters` on stream **`particle_to_coordinator`** to **CoordinatorBolt**
CoordinatorBolt	**Receive** `parameters` on stream **`particle_to_coordinator`** from **ParticleBolt**
CoordinatorBolt	End wait

Table 4: **Initialize** in ParticleStorm. This occurs at startup in both inference and evaluation modes.

Component	Action
CoordinatorBolt	Process initial model files or set parameters *de novo*
CoordinatorBolt	**AssignParameters** to all ParticleBolts
CoordinatorBolt	Enable **spout** via **Storm signals**

Table 5: Evaluation and inference mode functionality of ParticleStorm. We omit error checking and housekeeping tuples integral to all Storm topologies for clarity.

Control	Component	Action
	All components	**Initialize**
	Spout	**Receive enable** on stream **Storm signals**
For each data tuple		
	ParticleBolts	**Receive data** on stream **`default`**
	ParticleBolts	**Emit log likelihood** on stream **`default`**
	CoordinatorBolt	**Receive all log likelihoods** on stream **`default`**
	CoordinatorBolt	Calculate aggregate **log likelihood**
	CoordinatorBolt	**Emit aggregate log likelihood** on stream **`default`**
If inference mode		
If resample necessary		
	CoordinatorBolt	Calculate **Overwriting Particles**
	CoordinatorBolt	Calculate **Overwritten Particles**
	CoordinatorBolt	**RequestParameters** from **Overwriting Particles**
	CoordinatorBolt	**AssignParameters** to **Overwritten Particles**
End if		
	CoordinatorBolt	**Emit direct** `propagate` on stream **`coordinator_to_particle`** to **ParticleBolts**
End if		
	DataRecorder	**Receive aggregate log likelihood** on stream **`default`**
	DataRecorder	Record **aggregate log likelihood**
End for		

complementary functions by which the CoordinatorBolt sets or retrieves model parameters from the individual ParticleBolts. They involve communicating over the `coordinator_to_particle` and `particle_to_coordinator` direct streams to issue commands and receive responses.

After successful initialization, the CoordinatorBolt signals the spout via Storm Signals to indicate that the topology has been fully configured and is ready to process. At this point, the spout emits data tuples (one-at-a-time in ParticleStorm, as particle learning requires model updates per-data-point), which are received by all of the ParticleBolts via an 'all grouping' [16]. Each ParticleBolts evaluates the data tuple under its particle alone and emits the log likelihood on the `default` stream. The CoordinatorBolt receives the log likelihoods, aggregates them, and emits the overall log likelihood of the data point on the `default` stream, which the DataRecorder receives and records per the developer's design.

In evaluation mode, this data point is now fully processed, and the next data point is begun. In inference mode, however, the resample and propagate steps are undertaken first. If the CoordinatorBolt determines that resampling is necessary, a new vector of particles is calculated and differences from the previous vector are determined. Particles to spawn descendants, or overwriting particles, are queried for their parameters via RequestParameters. Particles to be overwritten then have these new parameters assigned to them via AssignParameters. Finally, regardless of whether resampling was necessary, the CoordinatorBolt emits a `propagate` command to each ParticleBolt on the `coordinator_to_particle` stream and awaits `ack` responses on the `particle_to_coordinator` stream. This marks the complete processing of this data point in inference mode, and the topology is now ready for the next data point.

3. Extensions to the Storm framework

In the process of developing ParticleStorm, it was necessary to develop a collection of additional capabilities for the Storm framework itself. These primarily concerned extension of preexisting capabilities to function inside of transactional topologies. Independent of the direct benefits of ParticleStorm itself, these extensions add useful functionality of Storm and will be contributed to the larger Storm community.

Although Storm allows for transactional topologies, and allows for ShellBolts, it does not allow for transactional ShellBolts. Hence, from the Storm ShellBolt class, we created a BatchShellBolt. This implied the modification of CoordinatedBolt to work around synchronization issues, because coordination tuples intrinsic to the functionality of CoordinatedBolt can otherwise be inadvertently processed before a data tuple is completely processed by the delegate shell process. Additionally, we extended the Storm multilang protocol to include 'housekeeping' commands between CoordinatedBolt and its delegate process, to inform the delegate of batch completion and allow the delegate to inform CoordinatedBolt of acknowledgment and completion of batch-finishing steps.

We also modified the ShellBolt class to allow direct execution of binary executables included in the Storm deployable uberjar. Previously, only system-level executables could be called, and they would be called on scripts included in the uberjar (e.g., Python or Ruby programs).

Finally, Storm Signals provides functionality to traditional Storm spouts, but not to transactional spouts, so we developed a transactional Storm Signals spout.

4. Discussion

As various aspects of our world are becoming increasingly measured with innumerable sensors of varying types, streaming data is becoming ubiquitous, and is vastly increasing in volume. Performing sophisticated analytics on these types of data is challenging and often infeasible because these computations are usually burdensome, and cannot keep up with the inflow rate of data. Storm, and other streaming frameworks, enable parallel processing of streaming data, allowing real-time analysis on data streams.

Here, we have presented ParticleStorm, an implementation of the particle learning algorithm in the Storm stream processing framework, a first of its kind. Particle learning is a sequential Monte Carlo algorithm that enables Bayesian statistical modeling to learn posterior probability densities via online inference at scale. In particular, ParticleStorm implements particle learning for composite mixture models, which allow nuanced models to be learned of phenomena with different numbers and different types of features, and to use those models to evaluate new data, with direct extension to tasks including clustering, classification, regression, anomaly detection, and drift detection.

Storm is still in its infancy, and although it has garnered a significant following, with users of the likes of Google, Yahoo, and Groupon, the literature on Storm is still scarce, which can present a high barrier to entry, especially for tasks outside of the traditional comfort zone addressed by Trident. One of the goals of this manuscript has been to reduce the entrance difficulty by providing a guide to other developers interested in uses of Storm beyond its most common utilities.

This implementation was possible via our extensions to the Storm project. In particular we have added the capability of Storm to delegate processing to external binary executables while still enforcing fault-tolerant exactly-once (transactional) processing. ParticleStorm is intentionally modular and easily extensible, allowing various pre- and post-processing extensions to be easily integrated. Its ideas, and the Storm extensions developed to allow its implementation, provide benefit for future efforts to use Storm, or other streaming frameworks, for complex statistical inference and machine learning tasks.

5. Acknowledgments

We would like to thank Vera Bulaevskaya and Daniel Merl for their invaluable contributions to this work. This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344.

References

[1] Apache Mahout: Scalable machine learning and data mining. https://mahout.apache.org/ [Accessed: 2014-04-19]

[2] Apache Spark - Lightning-Fast Cluster Computing. http://spark.apache.org/ [Accessed: 2014-04-19]

[3] Jubatus: Distributed Online Machine Learning Framework. http://jubat.us/ [Accessed: 2014-04-19]

[4] SAMOA. https://github.com/yahoo/samoa [Accessed: 2014-04-11]

[5] Storm-Pattern. https://github.com/quintona/storm-pattern [Accessed: 2014-04-11]

[6] Trident-ML. https://github.com/pmerienne/trident-ml [Accessed: 2014-04-11]

[7] Storm. http://storm.incubator.apache.org/ [Accessed: 2014-04-17]

[8] Storm applications in industry. https://github.com/nathanmarz/storm/wiki/Powered-By [Accessed: 2014-04-19]

[9] N. J. Gordon, D. J. Salmond, and A. F. M. Smith, "Novel approach to nonlinear/non-Gaussian Bayesian state estimation," *Radar and Signal Processing, IEE Proceedings F*, vol. 140, no. 2, pp. 107–113, Apr. 1993.

[10] A. Doucet and A. M. Johansen, "A tutorial on particle filtering and smoothing: fifteen years later," in *The Oxford Handbook of Nonlinear Filtering*, 2011, pp. 656–704.

[11] J. S. Liu and R. Chen, "Sequential Monte Carlo Methods for Dynamic Systems," *Journal of the American Statistical Association*, vol. 93, pp. 1032–1044, 1998.

[12] C. M. Carvalho, M. S. Johannes, H. F. Lopes, and N. G. Polson, "Particle learning and smoothing," *Statistical Science*, vol. 25, no. 1, p. 88âĂŞ106, 2010.

[13] A. P. Sales, C. Challis, R. Prenger, and D. Merl, "Semi-supervised classification of texts using particle learning for probabilistic automata," in *Bayesian Theory and Applications*, P. Damien, P. Dellaportas, N. G. Polson, and D. A. Stephens, Eds. Oxford University Press, Jan. 2013.

[14] Transactional topologies in Storm. https://github.com/nathanmarz/storm/wiki/Transactional-topologies [Accessed: 2014-04-17]

[15] P. T. Goetz. Storm signals. https://github.com/ptgoetz/storm-signals [Accessed: 2014-04-18]

[16] Storm concepts. https://github.com/nathanmarz/storm/wiki/Concepts [Accessed: 2014-04-19]

Multicore Construction of k-d Trees for High Dimensional Point Data

Victor Lu[1] and John C. Hart[2]

[1]HERE, a Nokia business unit
[2]University of Illinois at Urbana-Champaign

Abstract

This paper presents the first parallelization of FLANN's k-d tree for approximate nearest neighbor finding of high dimensional data. We propose a simple node-parallel strategy that acheives surprisingly scalable speedups on a range of inputs and hardware platforms. When combined with speedups from SIMD vectorization, our approach can achieve up to 91x total speedup over the existing FLANN implementation on a 32-core machine.

1. Introduction

The prevalence of digital cameras and Internet image hosting services, such as flickr, has created an explosion of online digital imagery, and with it many exciting new ways to utilize these large image databases. For example, 3-D reconstructions of the city of Rome have been built by finding and registering matching elements in the hundreds of thousands of photos on flickr tagged with the keyword "rome" [1]. Other examples use millions of photographs to perform scene completion [16], recognise panoramas in image collections [6], and infer labels on unknown images given a collection of labeled images[5].

These techniques are all built around the ability to find similar images to a given image, based on some kind of large vector representation of the image. Entire images can be represented by a low-resolution version of the image [26] or by a GIST descriptor [24]. Localized regions within an image can be represented by the concatenation of its underlying RGB values or by vectors computed using SIFT [19] or HOG feature transforms [10]. Similarity between images or their regions can then be measured by the Euclidean distance of their vectors.

Hence, finding similar images or parts of images amounts to solving the *nearest neighbor problem*: Given P a set of n k-dimensional data points in R^k, construct a data structure that helps us quickly find the *nearest neighbor* $p^* = \min_{p \in P} d(p, q)$ to any query point $q \in R^k$.

When dealing with high dimensional point data, such as image and region descriptors, existing nearest neighbor methods invariably suffer from the *curse of dimensionality,* which degrades search time to that of a brute force search. To regain algorithmic efficiency, *approximate nearest neighbor* methods find query results within a user specified error bound of the exact nearest neighbor. This is often acceptable for image searches since the descriptor vector distance is not necessarily the "perceptual distance" between two images or image regions.

The k-d tree[3] is a popular method for finding exact and approximate nearest neighbors. Its hierarchical data structure can be constructed in $O(n \log n)$ time and supports queries in $O(\log n)$ time. Once constructed, the nearest neighbor to a query point can be quickly found by examining only the data points residing in nearby leaf nodes and culling entire subtrees that are too far away. By examining a restricted number of leaf nodes, search is further accelerated, but at the risk of missing the exact nearest neighbor and being left instead with an approximate one.

While k-d trees speedup an otherwise brute force search, its construction can still represent a significant bottleneck in a variety of applications. Several recent methods in example-based inpainting [9], super resolution upsampling [15], non-local mean denoising [7], and object detection [20] must first construct a k-d tree for each received image in order to facilitate subsequent nearest neighbor queries into the image. The interactivity of these methods thus depends very much on how quickly the k-d tree can be constructed.

The parallelism found in modern multicore CPUs offers the hope of accelerating k-d tree construction, but is not yet realized in any existing high dimensional k-d tree implementations, such as the widely used FLANN (Fast Library for Approximate Nearest Neighbors) [22] and ANN [21]. The latest version 1.8.0 of FLANN parallelizes across separate queries and contains a GPU k-d tree builder specifically for 3-d points, but construction of high dimensional

k-d trees remains single threaded.

We present here the first parallelization of FLANN's high dimensional k-d tree builder. This paper is structured as follows. Section 2 reviews previous methods for parallelizing k-d tree construction. Section 3 describes our node-parallel strategy and its implementation. Section 4 demonstrates the scalability of our approach. Section 5 illustrates the importance of our work in a concrete real world application: logo detection.

2. Related Work

Many methods exist for finding nearest neighbors in high dimensional space, some more useful than others in specific situations. For example, hashing approaches, such as locality sensitive hashing (LSH) [11], have been investigated for their theoretical and qualitative benefits though they can underperform compared to alternatives in practical situations [23]. Vantage point (VP) tree [29] methods have been shown to achieve favorable search efficiency on image patches [18], but may take longer to build than k-d trees: when partitioning, VP trees must compute full vector distances to a chosen *vantage point*, whereas k-d trees split on an axis aligned plane which requires examining only a single vector component. A brute force search using the GPU can find exact nearest neighbors more quickly than a k-d tree [14], but cannot benefit from the further speedups enabled by approximate methods. Special purpose methods such as PatchMatch [2] outperform alternatives on their special cases (for PatchMatch that of finding similar image regions). Our work does not claim to be the "size that fits all," but instead we accelerate the situations where k-d trees are most useful.

Deciding on the proper nearest neighbor method for a given task may require much trial and error. The FLANN library implements a variety of these methods, while providing a mechanism for their automatic selection [23]. Our parallel k-d tree builder can be used as a drop in replacement for FLANN's k-d tree builder, once again benefitting the situations where k-d trees are most useful.

Much of the previous work on parallel construction of k-d trees have focused on low dimensional (3-d) versions, and focus their parallel performance on the computation of a surface area heuristic (SAH) over all elements to find the appropriate position of each splitting plane. For example, GPU methods for computing SAH k-d trees for accelerating ray tracing [30, 8, 28] construct the top levels of the tree in a breadth-first manner that streams through all elements at each level to compute the best splitting plane positions. Such an approach would not work well for FLANN-style computation of approximate nearest neighbors, which uses a small (e.g. 100-element) subset of points to discover the dimensions of greatest variance.

3. Method

Briefly, FLANN's recursive k-d tree construction algorithm proceeds as follows. On each recursive step, the algorithm picks one of the five dimensions with highest variance, estimated using a random subset (e.g. 100) of the node's data points, and splits its data along this dimension at its mean value, estimated on the same 100 element subset. Random subset selection is achieved by randomizing the list of vectors just once at the start of build and picking the first 100 at each recursive step. A node is made a leaf if it contains exactly one point.

We parallelize computations *across* nodes by mapping nodes to parallel *tasks* and *within* nodes by vectorizing its mean and variance estimation steps. Parallel tasks are spawned dynamically as new child nodes are created, while a task scheduler (here TBB [17]) takes care of mapping their executions onto physical cores. Section 3.1 describes the details of implementing this strategy.

Standard k-d tree builders such as in FLANN expect an explicit listing of its input vectors. When feature vectors are defined on overlapping windows in an image (*e.g.* 32×32 patches), explicit listings become especially memory *inefficient*, as each pixel value is relisted each time it is overlapped by a window. For example, a 1024×768 RGB image takes just 2.25MB, whereas an explicitly listing of its 32×32 patches requires up to 2.09GB! Section 3.2 describes modifications for avoiding this explicit listing, thus achieving orders of magnitude savings in memory.

3.1. Parallelization and Vectorization

Our implementation leverages two recent advances in programming tools for utilizing multicore parallelism:

Support for nested task parallelism in the form of libraries and language extensions such as TBB, Cilk Plus, OpenMP and WOOL allow programs to dynamically spawn tasks and tasks to spawn additional tasks, while a runtime scheduler, such as[13] [4], takes care of mapping tasks to physical processors. This style of parallel programming maps naturally to node parallel k-d tree construction, where tasks encapsulate the processing of a node and tasks are spawned when recursing on children nodes.

Auto vectorization capabilities of modern compilers coupled with preprocessor directives and the `restrict` keyword provide an almost effortless way in many cases for utilizing the wide vector units (now 8-wide) in recent processors. In our implementation, we vectorized the mean and variance computation during tree build and the distance computations during traversal. Specifically, with the Intel C++ Compiler (`icc`), we use the `restrict` keyword to assure the compiler that source and target arrays do not overlap, we added `#pragma simd`'s before for-loops and used the `-vec-report2` compiler option to check whether vectorization took place. Vectorization speedups is

slightly sublinear due to overheads such as moving single byte `chars` into 4 byte vector register slots.

We avoid racing on a global random number generator (RNG) state and suffering the penalties of false sharing, by using a reentrant RNG. We explicitly pass a RNG state into each node task, and pass the updated RNG state to the left child task and an arbitrarily offseted RNG state to the right child task. In practice, search performance does not degrade from this pseudo-random hack.

During parallel tree build, all threads will be simultaneously making requests to allocate new nodes. To handle this in a scalable fashion, we use TBB's scalable allocator.

Computing mean and variance requires scratch space with size proportional to k. To avoid dynamically allocating this space for each task, we maintain preallocated per thread scratch space using TBB's `enumerable thread specific` template.

3.2. Memory Efficient Indexing of Image Patches

The problem at hand is stated in the following generalized setting: Given R a raster grid of length d subvectors $v_{i,j} \in R^d$ (Eq. 1), we define the vector $\mathbf{v} \in R^{M \times N \times d}$, at each $M \times N$ window on the raster grid, as the concatenation of the $v_{i,j}$ subvectors covered by the window (Eq. 2). There may be multiple R's of different rectangular shapes, but all must have the same subvector length d. The goal is then to constuct a k-d tree on the set of all such $\mathbf{v}$'s without having to explicit list them but by instead operating directly on the raster grids.

$$R = \begin{bmatrix} & \vdots & & \vdots & \\ \cdots & v_{i,j} & \cdots & v_{i,j+N-1} & \cdots \\ & \vdots & \ddots & \vdots & \\ \cdots & v_{i+M-1,j} & \cdots & v_{i+M-1,j+N-1} & \cdots \\ & \vdots & & \vdots & \end{bmatrix} \quad (1)$$

$$\mathbf{v} = [v_{i,j}, \cdots, v_{i+M-1,j}, \cdots, v_{i,j+N-1}, \cdots, v_{i+M-1,j+N-1}] \quad (2)$$

To make concrete, for 32×32 RGB patches, we have $M = N = 32$ and $d = 3$. When considering the Felzenszwalb variant [12] of the HOG feature vector, we have $M = N = 8$ and $d = 31$. In both cases, the plurality of raster grids may correspond to different images or separate levels in a pyramid.

We assume in memory the raster grids are laid out in a single array `pyr` as the concatenation of the raster grids, each of which is itself a concatenation of its subvectors in column major order.

In standard builders, each vector is represented by an `offset` into the array of vectors and its `i`-th component is indexed by `offset + i`. When reordering a list of vectors, such as during partitioning or during the initial randomizing of list ordering, the array of `offset`'s is rearranged to avoiding the massive data movement of directly rearranging the array of vectors.

In our modified builder, in addition to an `offset` into `pyr` specifying the start of the top left subvector of a window, we also record for each vector a `stride`, which is the number of array elements in a column of subvector in the level that the vector is in. The `index` of the `i`-th component of a vector represented by `offset` and `stride` is then computed in C/C++ as (see Figure 1):

$$\begin{aligned} \texttt{index} &= \texttt{offset} \\ &+ \texttt{i/d/M} * \texttt{stride} \\ &+ \texttt{i/d\%M} * \texttt{d} \\ &+ \texttt{i\%d} \end{aligned} \quad (3)$$

In practice, one never has to evaluate the full expression each time a vector component is accessed. During partitioning, when a set of vectors is split along the `i`-th dimension, a large portion of the computation in Eq. 3 is constant across iterations and can thus be moved outside the loop (Code 1). When computing the mean and variance of a set of vectors, indexing becomes even simpler, since most components in the same vector are in fact contiguous in `pyr` (Code 2). We also observed that vectorizing the inner loop of Code 2 is profitable since the loop usually iterates over a sufficiently large number of contiguous elements in `pyr` (248 for HOGs, 96 for 32×32 RGB patches).

It is sometimes useful to consider the set of all unit normalized vectors (*i.e.* $\frac{\mathbf{v}}{||\mathbf{v}||}$), as in [20]. Since each subvector is shared by multiple vectors, the unit length normalization cannot be pre-applied to the subvectors beforehand. Instead, we can precompute and store per vector "normalization constants" in a separate array and index it with `offset / d` each time a component is accessed and normalize it using the retrieved constant.

4. Results

We evaluated our parallel k-d tree builder by characterizing its performance on a range of inputs and hardware platforms.

Test inputs. We considered 128-d *SIFT* keypoint descriptors [19], 384-d *GIST* image descriptors [24], 1024-d 32×32 tiny images, and 4096-d 64×64 image patches. For SIFT, we used the first 0.5M, 1M and 5M SIFT vectors from `cd` in Stewenius *et al.*'s dataset [25]. For GIST and tiny images, we used the first 0.5M, 1M and 5M GIST vectors and tiny images from the Tiny Images Dataset[26]. For image patches, we randomly selected two subsets of size 0.1M and 1M from Winder *et al.*'s dataset [27].

Hardware platforms. Experiments were performed on a `desktop` machine representative of a consumer level

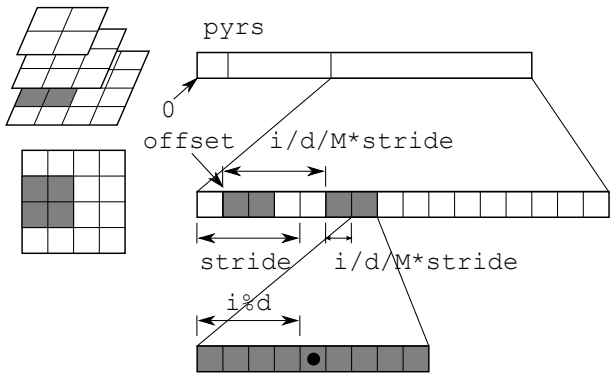


Figure 1. Layout of HOG pyramid in memory. A vector defined on a 2×2 cell window and its array elements in memory are shaded in gray.

```
int c1 = i / d / M;
int c2 = i / d % M * d + i % d;
for (int j = 0; j < n; j++) {
   int idx = offsets[j] +
      c1 * strides[j] + c2;
   if (pyr[idx] < split_val) {
      ... // sort left
   } else {
      ... // sort right
   }
}
```

Code 1. Iterating over i-th components of a set of `n` vectors specified by arrays `offsets` and `strides`

```
int idx = offset;
int width = M * d;
for (int j = 0; j < N; j++) {
   // following loop can be easily vectorized
   for (int l = 0; l < width; l++) {
        ... // work on pyr[idx + l]
   }
   idx += stride;
}
```

Code 2. Iterating over components of single vector specified by `offset` and `stride`

computer and a high end `server` machine (Table 1). All programs were compiled using `icc` version 12.1.0, with options `-O3` and `-xSSE4.2` on `server` and `-xAVX` on `desktop`.

Figure 2 compares the single threaded running time ($P = 1$) of our parallel builder against FLANN version 1.7.1's k-d tree builder. FLANN compiled "fresh out of the box" was not auto-vectorized but was easily modified (Section 3.1) to allow for the compiler to do so. Figure 2 shows the huge speedups achievable by ensuring the compiler indeed vectorizes. Unsurpisingly, once vectorized, FLANN's k-d tree builder runs at virtually the same speed as our parallel builder at $P = 1$. This comparison verifies that our single threaded running time, relative to which subsequent parallel speedups shall be computed, is indeed competitive.

Name	Machine description
`desktop`	Intel Core i5-3550 @ 3.30GHz (4 cores, 8 vector lanes) 16 GB RAM 64-bit Fedora Linux 16, kernel 3.2.9-2
`server`	Intel Xeon L7555 @ 1.87 GHz (4×8 cores, 4 vector lanes, 24 MB L3) 64 GB RAM 64-bit Sci. Linux 6.2, kernel 2.6.32-220

Table 1. Machines used in this work

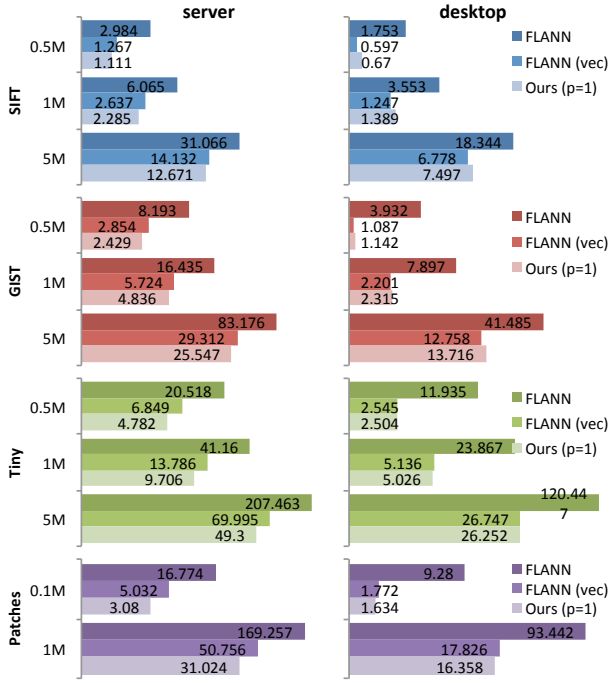


Figure 2. Comparison of serial k-d tree build times in seconds achieved by FLANN version 1.7.1 compiled "fresh out of the box," a FLANN modified to ensure auto vectorization by the compiler, and our parallel builder with number of threads set at $P = 1$.

Figure 3 reports parallel speedups relative to "Ours($P = 1$)" in Figure 2. As shown, our parallel k-d tree builder achieves scalable speedup and tremendous time savings across all chosen test inputs and hardware platforms. Compared to the non-vectorized "fresh out of the box" FLANN, our parallel k-d tree builder is up to 91.5x faster (Figure 4).

Not shown in figure 3, is that for smaller input sizes, we actually experience a slight parallel *slow down*. This is probably due to excessive stealing and limited parallelism in small inputs. But in most use cases this is okay since small inputs already build in less than a second.

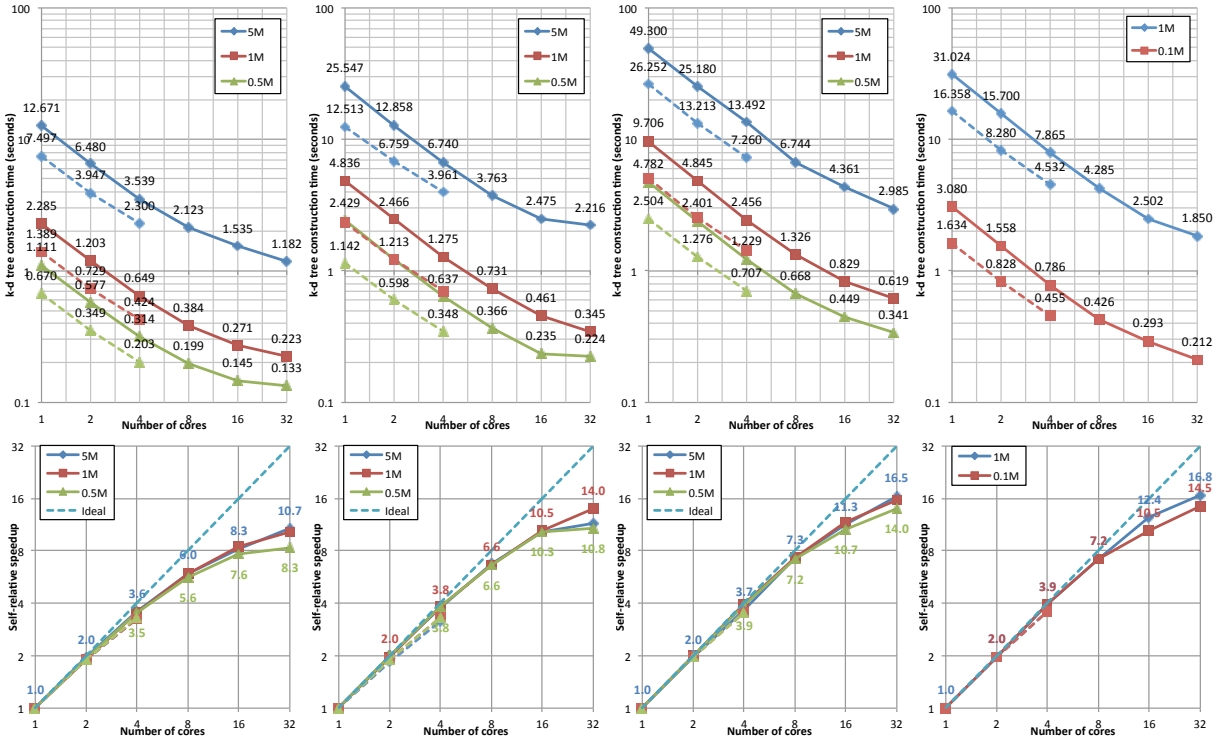


Figure 3. Absolute k-d tree build time in (top row) and self relative speedup (bottom row) for various input sizes, point dimensions and machine configurations. From left to right, point data type are SIFT feature descriptors (128-d, `uchar`), GIST image descriptor (384-d, `float`), 32×32 tiny images (1024-d, `uchar`), and 64×64 image patches (4096-d, `uchar`). Data points corresponding to solid lines were collected on `desktop` while dashed lines on `linux-server` (see Table 1). Speedup is relative to "Ours" in Figure 2

5. Application: Logo Detection

We examine the benefits of using our parallel k-d tree builder in a larger application by applying it to the logo detector described in [20], which works as follows. First, a set of *part vectors* are trained, each corresponding to a specific part of a specific logo class. Given a novel image, the image is then searched for patches whose HOG vector is sufficiently close in Euclidean Distance to any of the part vectors. This search can then be performed using either a k-d tree or any other nearest neighbor method.

We used a 4×10 core Intel Xeon E7-4860 machine running at 2.27 GHz to measure detection time over a range of core counts. We reimplemented the logo detector in [20] entirely in C++ and compiled using gcc 4.4.7 with option `-O2`. Both brute force and k-d tree based logo detection are parallelized across part vectors and distance computations vectorized using SSE intrinsics. The unit length normalization required in [20] was implemented as described at the end of Section 3.2. Following [20], we train a set of 512 part vectors. Detection was performed on a 1024×768 image from the FlickrLogos32 dataset.

Figure 5 shows detection time with and without a parallelized k-d tree builder. As core count increases, the time to build a k-d tree serially quickly dominates the overall detection running time, thus limiting further parallel speedups. And as the easily parallelized brute force detection continues to scale linearly, the k-d tree detectors advantage over a brute force detection quickly diminishes and is in fact overtaken at 32 cores. Thus, as the number of cores increase, a parallelized k-d tree construction is crucial for helping k-d tree methods stay competitive with a massively parallelizable brute force method.

References

[1] S. Agarwal, N. Snavely, I. Simon, S. Seitz, and R. Szeliski. Building Rome in a day. In *ICCV*, 2009.

[2] C. Barnes, E. Shechtman, D. B. Goldman, and A. Finkelstein. The generalized PatchMatch correspondence algorithm. In *ECCV*, 2010.

[3] J. L. Bentley. Multidimensional binary search trees used for associative searching. *C. ACM*, 18(9):509–517, Sept. 1975.

[4] R. D. Blumofe and C. E. Leiserson. Scheduling multithreaded computations by work stealing. *J. ACM*, 46(5):720–748, Sept. 1999.

[5] O. Boiman, E. Shechtman, and M. Irani. In defense of Nearest-Neighbor based image classification. In *CVPR*, 2008.

[6] M. Brown and D. Lowe. Recognising panoramas. In *ICCV*, 2003.

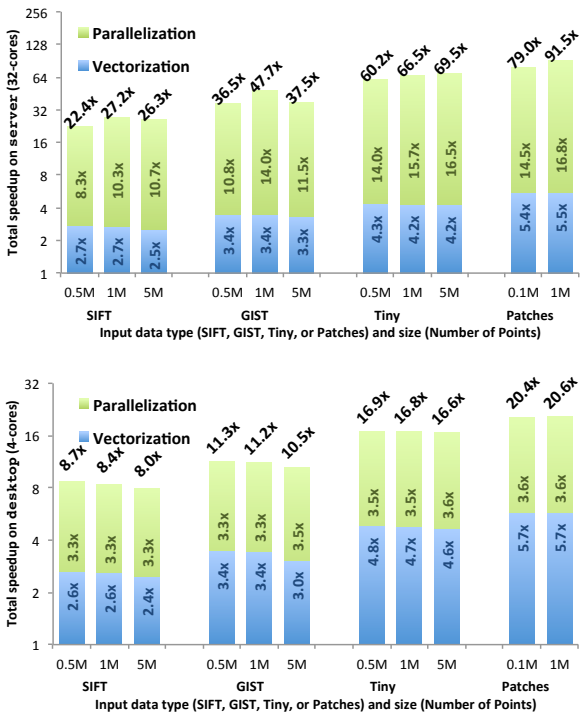


Figure 4. Total speedup of our node parallel k-d tree builder after parallelization and vectorization. Speedups here computed relative to "FLANN" in Figure 2

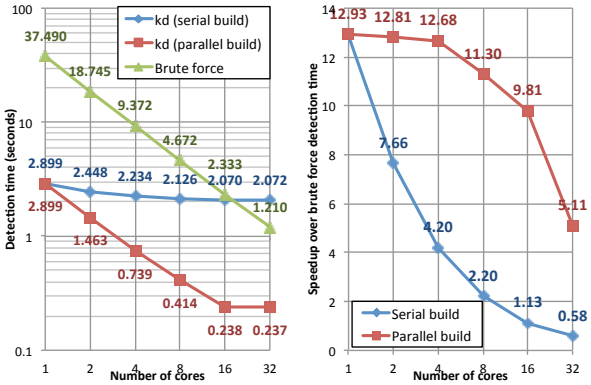


Figure 5. Scalability of overall detection time (left) and speedup over brute force detection (right), with and without parallel k-d tree build. In both plots, horizontal axis is the the number of cores.

[7] A. Buades, B. Coll, and J.-M. Morel. A non-local algorithm for image denoising. In *CVPR*, 2005.

[8] B. Choi, R. Komuravelli, V. Lu, H. Sung, R. L. Bocchino, S. V. Adve, and J. C. Hart. Parallel SAH k-D tree construction. In *HPG*, 2010.

[9] A. Criminisi, P. Perez, and K. Toyama. Region filling and object removal by exemplar-based image inpainting. *Transactions on Image Processing*, 2004.

[10] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005.

[11] M. Datar, N. Immorlica, P. Indyk, and V. S. Mirrokni. Locality-sensitive hashing scheme based on p-stable distributions. In *SCG*, 2004.

[12] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *PAMI*, 2010.

[13] M. Frigo, C. E. Leiserson, and K. H. Randall. The implementation of the Cilk-5 multithreaded language. *SIGPLAN Not.*, 33(5):212–223, May 1998.

[14] V. Garcia, E. Debreuve, F. Nielsen, and M. Barlaud. K-nearest neighbor search: Fast GPU-based implementations and application to high-dimensional feature matching. In *ICIP*, 2010.

[15] D. Glasner, S. Bagon, and M. Irani. Super-resolution from a single image. In *ICCV*, 2009.

[16] J. Hays and A. A. Efros. Scene completion using millions of photographs. In *SIGGRAPH*, 2007.

[17] Intel. Threading Building Blocks. http://www.threadingbuildingblocks.org/.

[18] N. Kumar, L. Zhang, and S. Nayar. What is a good nearest neighbors algorithm for finding similar patches in images? In *ECCV*, 2008.

[19] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, Nov. 2004.

[20] V. Lu, I. Endres, M. Stroila, and J. C. Hart. Accelerating Linear Classifiers with Approximate Range Queries. In *WACV*, 2014.

[21] D. M. Mount and S. Arya. ANN. A Library for Approximate Nearest Neighbor Searching. http://www.cs.umd.edu/~mount/ANN/.

[22] M. Muja and D. G. Lowe. FLANN - Fast Library for Approximate Nearest Neighbors. http://www.cs.ubc.ca/research/flann/.

[23] M. Muja and D. G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *VISSAPP*, 2009.

[24] A. Oliva and A. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *IJCV*, 42(3):145–175, May 2001.

[25] H. Stewenius. UK-Bench Recognition Homepage. http://vis.uky.edu/~stewe/ukbench/data/.

[26] A. Torralba, R. Fergus, and W. Freeman. 80 million tiny images: A large data set for nonparametric object and scene recognition. *PAMI*, 30(11):1958 –1970, nov. 2008.

[27] S. Winder, G. Hua, and M. Brown. Learning Local Image Descriptors Data. http://www.cs.ubc.ca/~mbrown/patchdata/patchdata.html.

[28] Z. Wu, F. Zhao, and X. Liu. SAH KD-tree construction on GPU. In *HPG*, 2011.

[29] P. N. Yianilos. Data structures and algorithms for nearest neighbor search in general metric spaces. In *SODA*, 1993.

[30] K. Zhou, Q. Hou, R. Wang, and B. Guo. Real-time KD-tree construction on graphics hardware. In *SIGGRAPH Asia*, 2008.

Design and Implementation of a Cloud-Based Big Data Programming Service

Yen-Yu Lin, Shu-Ming Chang, Su-Shien Ho, Ying-Ti Liao, Hsing-Chang Chou,
Shih-Chang Chen, Xuan-Yi Lin, Jiazheng Zhou, and Yeh-Ching Chung
Department of Computer Science
National Tsing Hua University
Hsinchu, Taiwan, R.O.C.
{raylin, shuming, sushien, eiteki, hchou, shcchen, xylin, jzhou}@sslab.cs.nthu.edu.tw
ychung@cs.nthu.edu.tw

***Abstract*—With the rise of cloud computing in recent years, more computing resources have been brought from enterprises to normal users. Cloud computing plays a major role in big data programming, and makes it feasible for everyone to deal with massive data. However, there are always some obstacles keeping unskilled users from utilizing the power of cloud computing. For example, the experts in statistics need to analyze big data, which heavily relies on computation power, but they may not necessarily possess the ability to build up reliable, flexible and scalable clusters. To overcome the gap, we address the challenges and design a user-friendly service to let users execute their program on website or from eclipse plugin. Considering to big data, we offer two storage services for different scale of datasets. Furthermore, we support not only certain kinds of programming system, such as MapReduce and MPI, but also more general execution environments and languages, including CUDA, Java and C. For design evaluation, we implement the service and prove that the architecture takes the advantages of cloud computing and it has ability to deal with big data programming.**

Keywords—Big Data; Cloud computing

I. INTRODUCTION

Cloud computing leads a trend which spreads from industries to academics [1]. The concept of big data helps enterprise gets better understanding of customers and establishes the great paradigm of data processing [2]. In industry, enterprises spend lots of cost to train IT staff to adapt the revolution of the cloudified big data architecture, and IT specialists may need to set up MPI or Hadoop cluster for different purposes. For academia, building up an execution environment with inadequate resources is relatively more difficult than that for industry. Therefore, the first barrier [3] between user and infrastructure needs to be solved before starting big data programming. This barrier can be caused by several reasons, such as the lack of physical machines, the unskillful server hosting, or security issues. In order to deal with this problem, one of the best solutions is finding an already hosted service to process big data. Unfortunately, existing services often provide only specific programming language or frameworks. Besides, operation over the terminal brings user another obstacle, which changes user's coding experience and working style. Cloud computing lets computing resources available for all user. However, the mainstream framework keeps untrained users away from the land of big data. To solve this problem, we propose and implement a user-friendly interface which tries to help users to start from scratch easily. The users are not necessarily familiar with server administration.

Big data programming is a time consuming process. In traditional way, a program is asked to be written and compiled on server side. Take MapReduce computations [4] for example, developers often write programs with Eclipse. The execution environment, however, is restricted to a Hadoop cluster, which reduces the efficiency of development. Developers iteratively do the same task to modify programs, i.e., programs are edited before exporting to .jar file, and later transferred to remote server. Furthermore, big data programming is utilized in diverse domains in different languages or with different implementations. Most popular programming paradigms/languages, such as MPI or R [5] are also widely used. To support these types of programs, we implement a unified platform, which is able to execute various kinds of languages without any extra configuration.

In addition to the convenience of usage, security is also an critical issue to be addressed. Both industry and academia own their confidential big data. In order to protect the confidential data, users are faced with a difficult question that whether the platform is secure and trustworthy. As mentioned above, solving the technical problems to ensure safety are one of the main considerations in our service design. In the proposed service, we isolate users from terminal operations which provides advantages to both service providers and users. First of all, users will not notice what really happens in the back-end, while focusing on their works. Secondly, adding a layer of APIs between client and service makes sure that programs are in control. The features of our service also make architecture flexible, scalable and maintainable.

This paper is organized as follows. Section II presents the related work including architectures or products that provide development environment. In section III, we propose an architecture of our service and describe the details of each components. Besides, the usage of service from the perspective of user is also explained. Section IV presents the implementation and evaluation of the proposed service. Finally, the conclusions and future work are given in the section V.

II. RELATED WORK

The Apache Hadoop framework is by far the most-utilized platform for big data processing. It provides a highly-efficient, scalable, and fault-tolerant architecture for that programmers can store their data and perform computation operations upon the data [6]. This framework not only provides flexible and configurable settings, but also leverages and optimizes available hardware resources. Although Hadoop is the de-facto solution for big data processing, the progress of setting up a Hadoop cluster and configuring software services is time-consuming and labor-intensive.

In recent years, there are many advancements in the area of web-based online development environments. The user experience of programming has been changed since mobile devices and web applications are popular now. Koding [7] is an example which is a revolutionized web-based online development environment. Different from normal online editor, Koding gives users a virtualized server and several gigabytes personal storage that could be considered as a standalone and private machine. On the website, users can invite co-workers to join the shared project and collaborate simultaneously. However, every coin has two sides, this website returns immediate results after every changes of developer's codes. On one side of this feature, developers get an debug-friendly playground and it increases programming productivity. On the other side, Koding is designed for dynamic language programming, which is not suitable for big data programming.

A product similar to Koding is called CoderPad [8], which supports more languages than Koding. It uses virtualization technique to offer users isolated operating systems. Although developers could not access terminal as they are accustomed to, developers are able to configure environment as they want. Coderpad supports not only dynamic languages but also static languages such as C or Java.

The services mentioned above both are designed to help users overcome the technical gaps and provide user-friendly development platforms. For big data programming, these kinds of services cannot fulfill basic needs in coding. Editor is the most important tool which affects the quality of development. The quality of current online editors are not good enough to replace a full-featured IDE like Eclipse. Besides, programs written in Java or R often utilize third-party libraries for specific purposes. Thus, for best development experience, developers usually do their work in local desktop environment.

III. SYSTEM DESIGN AND IMPLEMENTATION

Our service is designed as a middleware between infrastructure and client. The design of our service is to allow users to submit and run their big data programs as easy as possible. We anticipate users do all works at client side, and need not to handle anything in console mode. Therefore, accesses to terminal and storage are offered through APIs. A centralized OpenID service is used to authenticate user accounts throughout all components in the proposed system.

Fig. 1 shows the overview of system architecture. Section A introduces aspects for user-side environment, and describes the scenario of usage. Section B details the architecture of proposed service and its components. Section C describes the runtime environments.

A. Client-Side Environment

There are three components at the client side: a browser, an Eclipse plugin and a storage client. Both browser and Eclipse plugin play a similar role within whole service, and storage is the place which stores user's programs and big data.

1) Storage

Our service provides two kinds of storage with different characteristics to store programs and data. One is SSBox, a Dropbox-like storage service, that gives user a sharable and portable directory synchronized to remote server. Another storage is used for storing pure data on Hadoop distributed file system (HDFS).

Users develop programs in the local SSBox directory and programs are immediately synchronized to the remote server through storage client. If a program requires small-scale data as input, the input dataset can directly save into the same directory. On the contrary, when users require a large-scale and high-performance storage, users can access personal HDFS [9] with Eclipse plugin. Because Eclipse plugin has not only job submission feature, but also an HDFS explorer to handle the operations on HDFS.

The proposed service collects all results into cloud storage after user's program finished. An important mission of the storage client is to automatically sychronize output data from server side. The storage client checks any modification in server side, and synchronizes the changes.

2) Eclipse Plugin and Browser

The Eclipse plugin provides a graphical and user-friendly interface, which assists users launching programs easily. The plugin asks users to login before any operations. After login, users will be granted the permissions to access any components of the service. An authenticated user inputs necessary information, such as job type, application path/arguements, input path and output path, into the job submission form to submit a job. The paths are used to indicate locations of application and dataset in the storage.

In usage workflow, user use cloud storage directory as Eclipse work folder. Anytime when program be compiled, user can launch the program by this plugin. In the back-end, the plugin sends job information to service, and the service executes specified program to process specified input data on the cloud storage. To deliver the same experience as use in terminal, at the client side, users can get the instant console messages redirected from server side.

Our service provides a web-based portal, and the functionality of the portal is designed similar to Eclipse plugin. Consequently, web browser is an alternative submission agent to submit jobs.

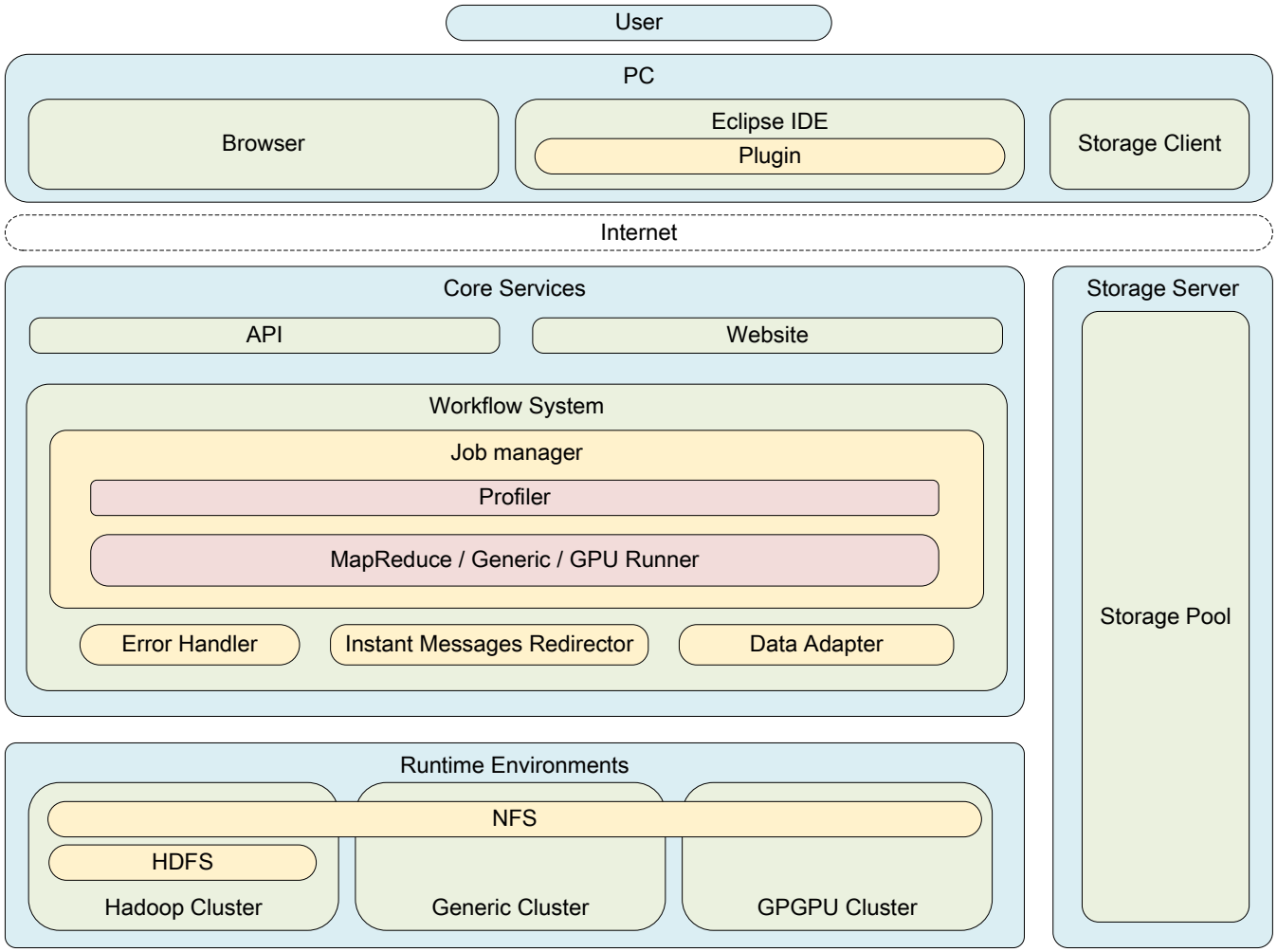


Fig. 1. System architecture overview.

B. Job Manager

Job manager is the core of our service. This isolated component is wrapped by REST API, and is the only way to connect to computational resources, and storage. Job manager is in charge of internal communications, account authentication, and program launch. To developers, job manager establishes an exclusive tunnel to send real-time messages. To the internal components, job manager controls the concurrency of active jobs, file transferring, and global error handling.

1) REST API

The outer layer of job manager is wrapped by REST API and the APIs are open worldwide. In this paper, we implement the APIs to Eclipse plugin and web-based portal. This interface provides two major features. First feature is the universal account authentication mechanism [10] which utilizes OpenID and integrates to a LDAP server. Every OpenID is connected to a unique LDAP account, and a user will be granted full access permission if his account is authenticated successfully by our authentication API. The second feature is job submission. We design a general description format for diverse programs. Once users put programs on our storage, users can follow the description format and send a request to our REST API to trigger programs.

2) Website

As mentioned above, we address the recommended development environment Eclipse for MapReduce program. However, big data programming is not restricted to MapReduce and Hadoop framework. For example, some legacy programs written in C language are not feasible to port to MapReduce. Instead, MPI is a better choice to speed up C programs and increase utilization of distributed computing power [11]. We assume that MPI programs are not always edited on Eclipse, so our service provides another interface for those users who don't install Eclipse.

In addition to supporting the same features as Eclipse plugin, X Window redirection function is implemented and given on website. For the programs using X library, we provide a personal display which won't conflict with others. If users turn on X support, users are capable to receive real-time image redirected by service.

3) Workflow System

Workflow system is the core component of job manager. The system controls a series of processes for diverse program

executions. Once the workflow system accepts a job submission, the job information will be passed through profiler and the workflow system dispatches the job to different runner according to the loading.

a) Profiler

Jobs are described with certain format by user. When workflow receives a submission, profiler will analyze the application type, input size, input source and arguments, etc. Then, profiler generates temporarily environment variables for each job at runtime, and each job is standalone and can decrease the reciprocal effect. After job's profile is established, the profiler will send job information to the certain program runner.

b) Data Adapter

Counter to storage client, data adapter retrieves the program and input data according to job settings. Our service provides two kinds of storage services. A cloud storage which synchronizes files to user's local directory and the HDFS explorer. Because our service aims at dealing with big data but not general input dataset, we assume that the best storage to store data is HDFS. So, if users put their input data at local storage client, data adapter will attempt to retrieve corresponding directory from cloud storage and save to HDFS. In the other hand, if users already upload input data through HDFS explorer, data adapter will treat these data ready-to-run, and do nothing.

c) Instant Message Redirector

Instant console and running status are always the critical information to developer. Therefore, we implement a real-time messages communication mechanism. Once Eclipse plugin or website client connects to the server, it establishes a unique connection. Both two clients include the same library and use long-polling socket to communicate with the server. To present messages given in console, Eclipse shows every standard output from service in console perspective. In order to send real-time console, we design a messages redirection policy. Every client is asked to join a room named by job id. If the server finds out any messages from execution engine, workflow system will redirect the messages and broadcast in the room. The redirector manages all messages from the server, so we hide other unnecessary messages for client. With the message redirector, users can get the same experience as in terminal mode.

d) Program Runner

Fig. 1 shows three different runners within workflow system. Job submission is not executed before sending by profiler, so runner takes the responsibility for job execution. Each job type has its own running environment. Like generic runner, the runner is capable to compile user's program and run the program with correct settings resolved by profiler. The settings include arguments, internal input path, internal output path, compilation options, etc. All types of runners have separated job queues. Owing to serve big data program, job queue is designed for leverage computation power and I/O loading. Once job started, program runner will generate few lines of commands according to settings, then execute the program.

e) Error Handler

Like normal workflow system, any stage could occur errors. To handle these errors/exceptions, we design an error handler as a global event listener. The error handler is not tend to replace the message redirector. Error handler is a standalone and fault-tolerant service, which keeps working even the message redirector is failed. Because of the isolation design of service, users only need to know whether their job is running or failed. Therefore, the job of error handler is to simply send an error message to users if error occurrs, and trigger the post process to clean up job data.

C. Runtime Environments

Three clusters of different runtime environments are shown in the bottom of Fig. 1. To share the big data, we assume that the better storage for the data is HDFS. Therefore, the HDFS installed on the Hadoop cluster shares the file system to others by using NFS. Clusters mount the HDFS at the same path, then job runner can map the input paths to internal paths. Each virtual machine has the same storage settings; however, operations to files will raise permission errors. To overcome the authorization problem, an LDAP service is used to manage local accounts. Consequently, an OpenID account maps to a local account, and permissions of this account are granted by LDAP service. After users are authenticated and authroized, workflow system then gets the privilege to walk through clusters.

Hadoop cluster accepts only MapReduce programs, and GPU cluster with CUDA library installed can run CUDA programs. Generic cluster is designed for gerneral-purpose jobs. In addition to supporting MPI, our generic cluster accepts customized script. We define some environment variables for users, and users are free to write their own script to compile programs, specify input data, run batch task and so on.

IV. EVALUATION

To evaluate our design, we have implemented and deployed the proposed service under the name of SSBDS, which is available as a part of the UniCloud project [12]. Currently there are dozens of users and hundreds of various big data computation jobs, including MapReduce, MPI and R programs have been carried out on the platform. As the part of evaluation, we prepare three types of programs.

A. MapReduce Program with Eclipse Plugin

First program is a MapReduce job that takes 160MB raw text as input data, and calculates the word counts.

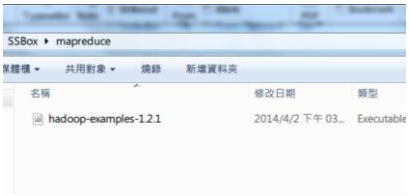


Fig. 2. MapReduce application at local directory.

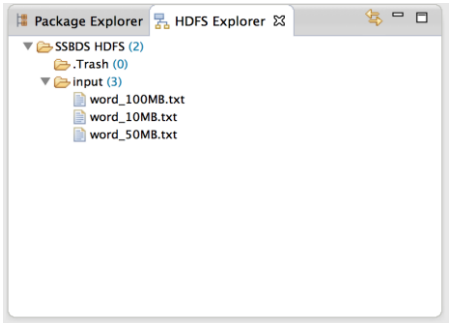


Fig. 3. The HDFS explorer.

Fig. 2 shows a Hadoop MapReduce program, which is placed at the synchronized directory of the cloud storage service, SSBox. In the local directory, the executable program is exported as .jar file. Like Dropbox client, the storage service synchronizes all files under the directory to storage server.

Our input data is ready and already uploaded to remote HDFS with Eclipse plugin. As figure 3 shows, we uploaded a directory named "input", which include 3 text files. Like the file explorer, all operations are done by the plugin.

Users configure the submission settings in our plugin as shown in Fig. 4. The path of the program is the relative path in cloud storage.

Fig. 5 shows the messages redirector which redirects the console in real-time in console perspective.

In Fig. 6, after a job is completed, the output directory specified above is synchronized with the result data. Users can retrieve the result data without any HDFS commands.

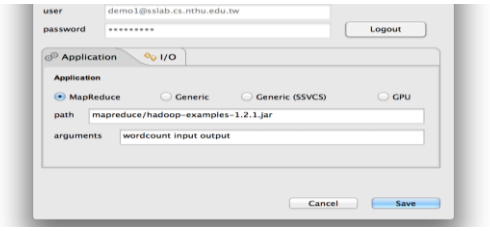


Fig. 4. Job submission dialog of the Eclipse plugin.

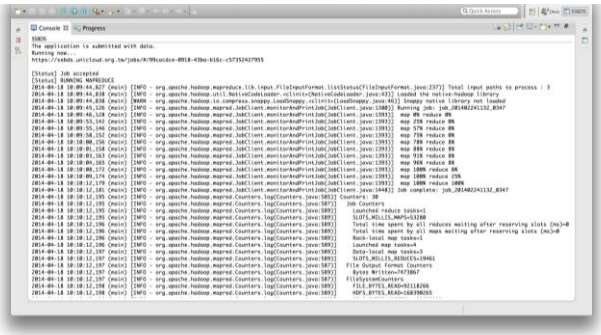


Fig. 5. Real-time console redirection.

Fig. 6. Output is synchronized to local directory.

Using desktop to submit a job and view result at local directory brings users a friendly environment, as they run the programs locally.

B. *R Program with Service Website*

Like Eclipse plugin, our website service provides similar user experience. In this section, we submit a R program via website with local input data. Besides, our R program uses X Window to draw charts, which are redirected by our service.

As shown in Fig. 7, we write a customized script and place R script at the same directory called "R". To evaluate the other cloud storage functionality, we put small-scale input data at the "R_input" folder as shown in Fig. 8.

Fig. 7. The R application.

Fig. 8. R input data.

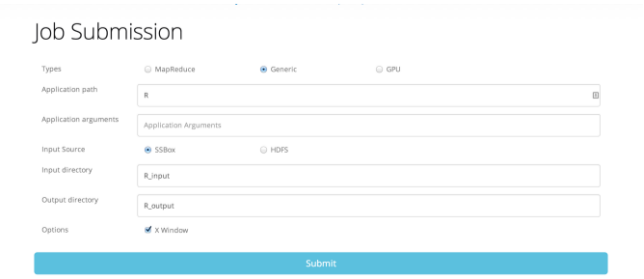


Fig. 9. Submission page on the website.

Fig. 10. Real-time console on the website.

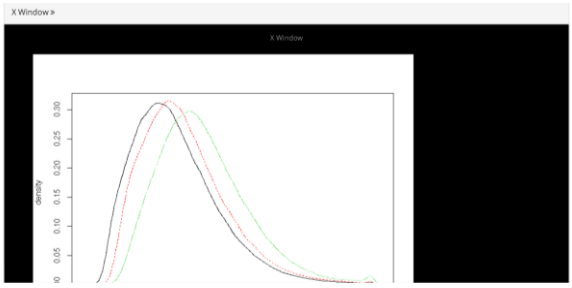


Fig. 11. X Window redirection.

The submission shown in Fig. 9 requires the same fields as in Eclipse submission settings. We specify "R" as program path, and "R_input" as input path.

With clear interface, users can get the current stage of job and consoles as shown in Fig. 10.

Besides console redirection, our service redirects X Window to client through browser. As shown in Fig. 11, the chart drew by R application is forwarded to client.

V. CONCLUSIONS AND FUTURE WORK

As big data became popular in recent years, the demands of massive data processing increase. We notice that the curve of building up big data development environment is extreme steep. Therefore, we propose and implement the service aiming at providing a robust and friendly development environment. To store user's massive data, we design two cloud storage spaces for different purposes that adapt the advantages of the distributed file system. We implement a storage client to synchronize user data to remote server automatically. Users can launch their programs since both programs and input data are located at our cloud. In order to send execution information, we design a message redirector to forward consoles to client. In addition to storage and message solutions, our scalable, fault-tolerant, and high-available infrastructure can provide the stability that is capable to run dozens of diverse programs at once. At last, once a job is completed, all results of the job will be synchronized to local. To provide a user-friendly and isolated environment, we provide different mechanisms to overcome the gaps, and try to satisfy most situations.

In the current stage of our service, we still have a long way to support various use cases. Without reaching computing servers, users cannot customize and modify settings to suit their own requirements. As mentioned in related work, virtualization technique can be another choice to build a personal virtual machine with our service installed. Secondly, our service to execute big data programs is not yet convenient for repetitive scientific jobs. Like for MapReduce jobs, a regular analysis model or programs would be used in several datasets with dozens of various arguments. Delegating execution cannot solve this kind of problem and save user's time. Therefore, adapting description language like BPEL (business process execution language) as submission could obviously enhance user experience.

Moreover, we attend to show all the messages as possible as we could; however, our messages redirector cannot determine whether the message is necessary for user or not. A more powerful mechanism is needed for messages passing, to give user more comprehensive usage.

In the future, we plan to add more features to improve our service and keep refining the environment to suit everyone's requirements.

REFERENCES

[1] Agrawal, D., S. Das, and A. El Abbadi. Big data and cloud computing: current state and future opportunities. in Proceedings of the 14th International Conference on Extending Database Technology. 2011. ACM.

[2] Lohr, S., The age of big data. New York Times, 2012. 11.

[3] Aloisioa, G., et al., Scientific big data analytics challenges at large scale. Proceedings of Big Data and Extreme-scale Computing (BDEC), 2013.

[4] Dean, J. and S. Ghemawat, MapReduce: a flexible data processing tool. Communications of the ACM, 2010. 53(1): p. 72-77.

[5] Team, R.C., R: A language and environment for statistical computing. R foundation for Statistical Computing, 2005.

[6] Bhandarkar, M. MapReduce programming with apache Hadoop. in Parallel & Distributed Processing (IPDPS), 2010 IEEE International Symposium on. 2010. IEEE.

[7] Koding Inc. Koding. 2014; Available from: https://koding.com.

[8] Woo, V. CoderPad. 2013; Available from: https://coderpad.io.

[9] Shvachko, K., et al. The hadoop distributed file system. in Mass Storage Systems and Technologies (MSST), 2010 IEEE 26th Symposium on. 2010. IEEE.

[10] Jeng, Y.-L., An OpenID Based Authentication Mechanism in a Distributed System Environment.

[11] Schadt, E.E., et al., Computational solutions to large-scale data management and analysis. Nature Reviews Genetics, 2010. 11(9): p. 647-657.

[12] UniCloud. 2014. Available from: https://www.unicloud.org.tw.

ARIANA: Adaptive Robust and Integrative Analysis for finding Novel Associations

V. Abedi[1], **M. Yeasin**[1], **and R. Zand**[2]
[1]Department of Electrical and Computer Engineering, Memphis University, Memphis, TN, USA
[2]Department of Neurology, University of Tennessee Health Science Center, Memphis, TN, USA

Abstract *- We have developed an integrated system, called ARIANA (Adaptive Robust and Integrative Analysis for finding Novel Associations). It is an efficient and scalable knowledge discovery tool designed to provide a range of services in the general areas of text analytics in biomedicine. It integrates literature mining and ontology mapping to find network of semantically related entities. The source for the literature data is PubMed and the ontology is from the MeSH database. Empirical studies were performed to evaluate the performance of ARIANA. Based on subjective and objective measures of evaluation ARIANA was able to discover knowledge relevant to the query.*

Keywords: Literature Mining; Knowledge Discovery; Latent Semantic Analysis (LSA); Hypothesis Generation; Multi-gram Dictionary, Ontology Mapping.

1 Introduction

The effective mining of biological literature can provide a range of services such as *hypothesis generation* or *semantic sensitive retrieval of information*. This helps to understand the potential confluence of different diseases, genes, risk factors as well as biological processes. The utility, in the sense of usability and scalability, of semantic-sensitive knowledge discovery (KD) tools are the tremendous increases in scientific publications and the diversity of the concepts. A plethora of the state-of-the-art applications on improving information retrieval (IR) and users' experience were reported in contemporary literatures and was succinctly reviewed in a recent survey by Lu et al. [1]. A total of 28 tools, targeted to specific needs of a scientific community, were assessed to compare functionality and performance. The common underlying goal of them all is to improve the relevance of search results, to provide a better quality of service as well as to enhance the user experience with the PubMed database [2]. Though these applications were developed to minimize "information overload", the question of scalability and improving KD require further research.

STRING - a Search Tool for the Retrieval of Interacting Genes/Proteins [3] and iHOP [4] were not among the 28 tools reviewed. Both applications translate unstructured textual information into more computable forms and cross-link them with relevant databases. However, the underlying techniques cannot capture the semantic relationship among entities. Existing techniques still lack the ability to effectively present biological data in easy to use form [5] and further KD by integrating heterogeneous sources of data. To effectively reduce information overload and complement traditional means of knowledge dissemination, it is imperative to develop robust, scalable and high precision applications that are versatile enough to meet the specific needs of a diverse community. The utility of such a system would be greatly enhanced with the added capability of finding semantically similar concepts related to various risk factors, side-effects, symptoms and diseases. There are a number of challenges in developing such a robust yet versatile tool. One of the main challenges is to create a fully integrated and a functional system that is specific to a targeted audience, yet flexible enough to be creatively used by a diverse range of users. To be effective, it is necessary to map the range of concepts using a set of criteria to a "dictionary" that is specific to the community. Second, it is important to ensure that the KD process is scalable with the growing size of data, and is effective in capturing the semantic relationship and network of concepts.

ARIANA is an efficient and scalable knowledge discovery (KD) tool providing a range of services in the general areas of text analytics in biomedicine. The core of ARIANA is built by integrating semantic-sensitive analysis of text data through ontology mapping (OM), which is critical for preserving specificity of the application and ensuring the creation of a representative database from an ocean of data for a robust model. In particular, the Medical Subject Headings ontology[6] was used to create a dynamic data-driven dictionary specific to the domain of application, as well as a representative database for the system. The semantic relationships among the entities or concepts are captured through a parameter optimized latent semantic analysis (POLSA). The KD and the association of concepts were captured using a Relevance Model (RM). The input to ARIANA can be one or multiple keywords selected from the MeSH and the output is a set of associated entities for each query.

The dynamic data-driven (DDD) concepts were introduced starting from the domain specific "dictionary creation" to the "database selection" and to the "threshold selection" for KD using RM. The key idea is to make the system adaptive to the growing amounts of data and also to the creative needs of diverse users. The key features distinguishing this work from closely related works are (but are not limited to): i) flexibility

in the level of abstraction based on the user's insight and need; ii) broad range of literature selected in creating the KD module; iii) domain specificity through mapping ontology to create DDD and application specific dictionary and its integration with POLSA; iv) presentation of results in an easy to understand form through RM, implementation of DDD concepts and modular design throughout the process; and v) extraction of hidden knowledge and promotion of data reuse. In essence, ARIANA attempts to bridge the gap between creation and dissemination of knowledge. Case studies were performed to evaluate the efficacy of the computed results.

A total of 276 Headings from the MeSH database are selected for the system. Furthermore, a multi-gram dictionary is constructed for the co-occurrence analysis. ARIANA allows users to query the system using any word(s) in the multi-gram dictionary. In our previous work [7] ninety-six common associated factors were selected through a literature review from numerous medical articles by two domain experts; this process created bias in the model and posed a major challenge on the system's scalability. Using the MeSH hierarchy to select the entities alleviates these problems.

2 Methodology

ARIANA has four main components: Data Stratification, Ontology Mapping (OM), Parameter Optimized Latent Semantic Analysis (POLSA), and Relevance Model (RM). Figure 1 shows the architectural view of ARIANA's backbone. First, a very large database is compiled based on domain knowledge and the choice of the Headings. Following this, the OM is used to generate a context specific dictionary and select a subset of MeSH entries, also referred to as Heading List, that are neither too specific nor too general. Based on the dictionary words a stratified database of titles and abstracts is curated using an automated process. Heading List and Dictionary are used as input to the POLSA to find semantic-sensitive association of concepts. The POLSA is used to capture pseudo semantic relationships among all entities using higher-order co-occurrences. Based on the user's query all the Headings are rank listed. The RM uses the ranked list of Headings and categorizes them into three groups of association: strong, potential and unknown, respectively. Subsequent subsections discuss the details of all the modules.

2.1 Data Stratification

Medical Subject Headings (MeSH) ontology is the main input to this framework. Heading List which contains a total of 276 Headings, and Dictionary are both built based on the MeSH ontology through an OM process. The dataset for of 276 Headings is downloaded from PubMed and stored in MySQL database. There are three tables (Heading, HeadingPMID, and PMIDContent) in the database. The Heading table holds the MeSH name and ID as well as the latest article associated with the Heading (this information can be used for efficient updating of the database); The HeadingPMID contains the list of all PMIDs that are linked to the Headings, where PMID is the unique identifier of PubMed abstracts. Finally the PMIDContent table contains information about each entrie: title, abstract, year and associated MeSH.

Using the Heading List, titles and abstracts for the past fifty years of publications are downloaded from the PubMed and stored in a MySQL database on a server. In our previous work, we have only looked at literature from the past twenty years and some obvious associations were missed. The expanded set of publication addresses that issue.

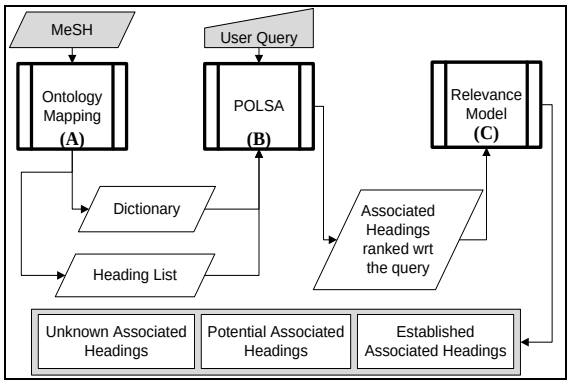


Figure 1. Flow Diagram of the ARIANA's backbone.

2.2 Ontology Mapping Module

The key function of OM is to use domain specific keywords mapped to a concise dictionary to achieve specificity required for practical applications. The input to this process is the MeSH ontology and the output is a multi-gram dictionary and a subset of Heading List. The use of a multi-gram dictionary is key towards creation of a data-driven model for finding novel associations through higher order co-occurrences. Multi-gram dictionary assures some level of semantics based on order of words, which is lost in statistical models that are based on Bag-of-Words. This overcomes the limitations of LSA-based model created using monogram dictionary, where performance of the system tends to deteriorate when multiple words are used as queries [8]. To create the multi-gram dictionary, first MeSH node identifiers are extracted, and then using a Perl script, the text file containing node identifiers is parsed to construct the mono, bi and tri-gram dictionary (see Figure 2). As the last filtering step, duplicates, stop words, words ending with a stop word, words starting with a stop word or number, and all words of length two or less characters are removed. For instance, using the MeSH identifier "Reproductive and Urinary Physiological Phenomena" the followings eight dictionary words are constructed: 1. Reproductive and Urinary, 2. Urinary Physiological Phenomena, 3. Urinary Physiological, 4. Physiological Phenomena, 5. Reproductive, 6. Urinary, 7. Physiological, 8. Phenomena. Two of the eight words are tri-grams, two are bi-grams, and the remaining four words are mono-grams. The size of the multi-gram dictionary is 39,107. Selection of a Heading List to create the representative model is also a critical step.

To create the Heading List, careful analysis by medical expert is performed and a subset of MeSH entries is selected to create the model. The majority of the Headings in the MeSH are in the Diseases[C] and Chemicals and Drugs

Chemicals and Drugs[D] categories. The subset of selected Headings – Heading List – is comprehensive and contains headings from an array of subjects including Diseases[C]; Chemicals and Drugs Chemicals and Drugs[D]; Psychiatry and Psychology Psychiatry and Psychology[F]; Phenomena and Processes Phenomena and Processes[G]; Anthropology, Education, Sociology and Social Phenomena Anthropology, Education, Sociology and Social Phenomena[I]; Technology, Industry, Agriculture Technology, Industry, Agriculture[J]; Named Groups Named Groups[M]; and Health Care Health Care[N]. A total of 276 Headings are selected (see table A1). The main focus when selecting the Headings is to include headings that are of general interest and that are relatively specific at the same time. Most of the headings are from the "C" category as the goal of the study is to enhance our understanding of diseases and their interactions. In our pilot project [7], the set of ninety-six entities is very diverse in terms of specificity, thus creating a systemic bias in the database. The systemic bias caused the score for generic entities (for which a large body of literature is available) to be lower compared to entities with limited information.

2.3 Parameter Optimized Latent Semantic Analysis

The POLSA module is the statistical method used to create a model based on the collection of text data. POLSA is based on Bag-of-Word (BOW) model; however, the integration of OM and utilization of multi-gram dictionary improves its domain specificity and semantic sensitive associations. In BOW model the order of words is lost, it is with the use of domain specific ontology that specific multi-gram words are incorporated in the dictionary (i.e. "chicken anemia virus").

The 276 Heading List, as well as the Dictionary are the input to the POLSA (see Figure 3). Each of those are parsed to create a term-frequency-inverse-document frequency (TF-IDF) matrix. The pre-processing step is minimized and does not include stop word removal and stemming; that is due to the structure of the dictionary as it contains multi-gram words which may have stop words within them. The TF-IDF matrix is then used to create the encoding matrix using a singular value decomposition. A user query, which can be any word in the dictionary (or a multiple of words from the dictionary), can be an input to this module. Using the encoding matrix, the query and the database are translated into the *eigen* space to compute the rank of each heading with respect to the query. The encoding matrix is created by keeping *eigen* vectors that preserves 90% of the total energy. Cosine similarity is used to compute the relevance score between the query and the Headings.

2.4 Relevance Model

The relevance model proposed in this paper is a logical extension of disease model originally reported in our previous work [7]. It is an intuitive, simple and easy to use statistical analysis of rank values to compute the strongly related, related, and not related concepts with respect to a user-query. Figure 4 illustrates the core concepts of the implemented relevance model. The concepts in this system are a subset of Medical Subject Headings and the user-query is constrained by the MeSH ontology. The underlying assumption is if concepts are highly associated then there is a large body of literature available to corroborate existence of their association. Similarly, if two concepts or biological entities are not well documented then they are only weakly associated.

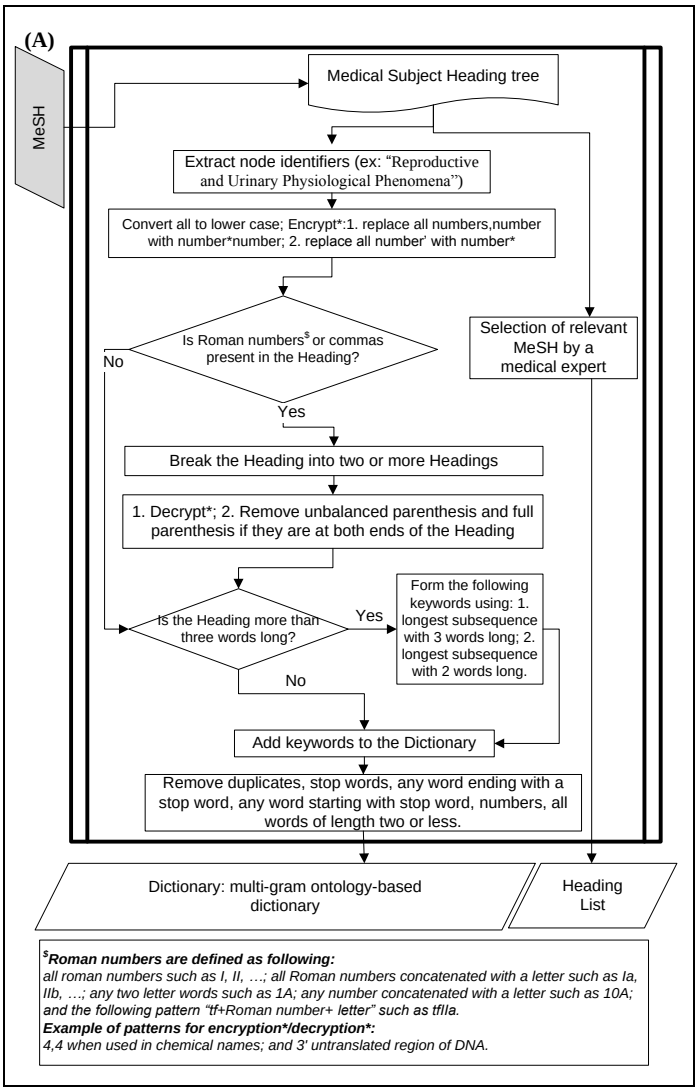


Figure 2. Module A: Ontology Mapping

Furthermore, since the distribution of relevance scores is a function of user queries, then the cut-off value to separate highly, possible and weakly associated entities must be determined dynamically. This requires a simplified yet effective model to ensure scalability; therefore, it was assumed that the distribution of the ranked list can be viewed as a mixture of Gaussian and the partition can be computed using the DDD threshold. In particular, the distribution of relevance scores of the Headings for a given query was approximated as a tri-modal Gaussian distribution. The separation of the three distributions allows implementation of the DDD cut-off system. In our previous work [7] a curve fitting approach was

used to estimate the parameters of the tri-modal distribution and the cut-off values. In this work, fuzzy c-mean clustering approach was implemented to achieve the same goal but in a more robust and scalable manner. This method is much faster and can provide a finely tuned mean to evaluate the results on demand. Furthermore, this DDD cut-off value determination can also be integrated in other IR systems.

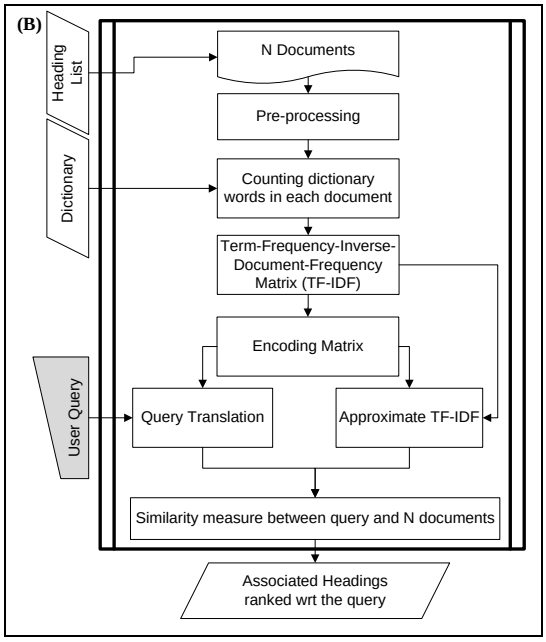


Figure 3. Module B: POLSA

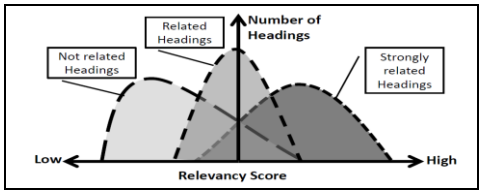


Figure 4: Statistical disease modeling hypothesis.

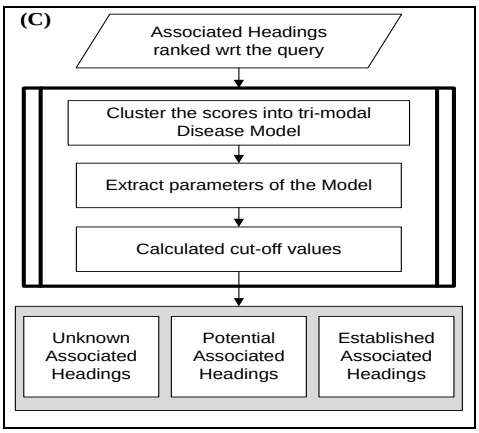


Figure 5. Module C: Relevance Model

The list of associated Headings that are ranked with respect to a user-query is used as input to the RM (see Figure 5). The top ranked Headings are strongly associated with the query and the Headings ranked at the bottom do not have significant evidence to support their association to the query. The Headings that are between the two extremes are the ones that might or might not be associated with the query as there is some supportive evidence for their association. These weak associations are important in the KD process and call for further investigation by domain experts.

The similarity measure between the query and all the associated Headings is used to cluster the Headings into three categories. Fuzzy c-means clustering is applied to group the associated headings using the MATLAB built-in-function. Using the clustering, the scores are first grouped into two clusters and based on the membership values of these two clusters, Algorithm 1 is used to assign each Heading to one of the three groups in the RM. The cosine cut-off values estimated through this process are dynamic and data-driven, hence the cut-offs are subject to change as the dataset expands. The input is the limit that is defined by an expert to separate the known and unknown Headings and place them into the possible Heading group (i.e. the gray zone), a conservative limit threshold of 0.9 was chosen to analyze the results (value of j in Algorithm 1).

```
SET a and b as cluster membership for headings such that sum(a)≥sum(b)
SET j as the limit to select headings in gray zone
FOR each heading
  IF a≤ b          THEN SET c to 1;   END IF
  IF abs(a-b) ≤ j  THEN SET d to 1;   END IF
END FOR
FOR each heading
  IF c=1           THEN SET group to high_Assoc;
  ELSIF d=1        THEN SET group to possible_Assoc;
  ELSE             SET group to no_Assoc;
  END IF
END FOR
```

Algorithm 1. Grouping Headings by fuzzy c-mean clustering

2.5 Evaluation

Evaluation of such analysis is challenging yet very important. The system was evaluated through comprehensive literature review and then further verified by an expert in the field. The test case is Ischemic Stroke and a specialized physician in Vascular Neurology validated the findings.

2.6 Information Retrieval and Knowledge Discovery

A series of queries are used to study and evaluate the potential and scope of the system. Using the test cases we have shown an example of knowledge discovery (KD) and the power of information retrieval (IR) with ARIANA. The goal is to detect level of noise when running general queries such as common diseases. The results of the analysis are presented in and further discussed in the subsequent sections.

3 RESULTS

ARIANA's main objective is to find the semantic sensitive network of associations among concepts and enhance the

quality of KD. Four different diseases were used as case studies to illustrate the utility and the scope of ARIANA. Diseases used for the study are Ischemic stroke (IS), Parkinson's Disease (PD), Lymphoma (LY), and Migraine (MG). Results obtained from the IS analysis are compared with literature and evaluated by a medical expert (Section 3.1). Results from PD, LY and MG are displayed and shortly discussed in Section 3.2. These examples demonstrate how this system can be used to extract information that can be forgotten – bridging the knowledge gap.

3.1 Case I: Single Query

Figure 6 displays the results with "Ischemic Stroke" as query. Table 1 lists the 18 selected headings with their respective relevance scores. The two cut-off values – obtained by the DDD system – place six Headings into the high and twelve Headings into the possible association group. All the Headings are directly or indirectly associated with IS. In two cases the indirect association was not clear and literature search was performed (stroke & Intermittent Claudication[9]; stroke & Cyanosis[10]). Table 2 lists lower ranking Headings for up to a relevancy score of 0.01. Majority of the Headings in table 2 have known association with IS.

3.2 Case II: Multi-Query

A number of diseases were used as query to find the network of associations. The summary of results is presented in Figure 7. In some cases the association is to a certain degree or it is indirect, through other Headings. Only in few instances the Headings do not have any known association with the query. *Parkinson's Disease (PD):* The list of ranked Headings for PD highlights the fact that this is a neurodegenerative disease, affecting movement and muscle functions. The identified elements also highlight that this disease is likely associated with environmental factors (i.e. heavy metal poisoning, cadmium poisoning, MPTP poisoning). *Migraine (MG):* The top three ranked Headings are: coffee, tea and sexually transmitted diseases (STD) with score of 0.689, 0.592 and 0.286 respectively. The first two associations are expected; yet, the association between MG & STD is less predictable. In a recent study[11] 200 HIV/AIDS patients were studied and among them 53.5% reported headache symptoms and 44% were diagnosed with MG. A strong correlation between the severity of the HIV and the strength and frequency of the MG attacks was also found. Interestingly, this specific article[11] is not in the current data model; hence this is a clear example of KD. *Lymphoma (LY):* LY begins in the cells of the immune system. Generally, LY seems to be highly associated with different types of infections because patients with a weakened immune system have a higher chance of LY. Interestingly LY and PD are associated with cadmium poisoning. The risk of developing childhood acute lymphoblastic leukemia were increased with exposure to cadmium in the drinking water[12]. Some associations, such as cadmium and PD or LY, are known but can be considered buried in the ocean of publications as they are not usually referenced in medical protocols. ARIANA has potential for data reuse by extracting existing associations and improving the quality of IR.

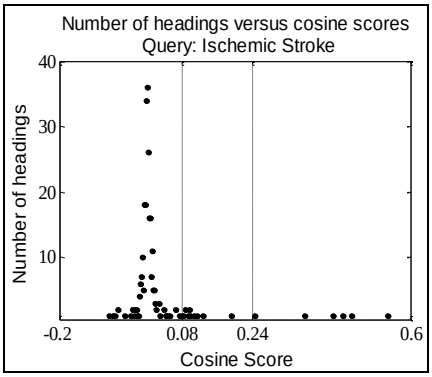


Figure 6. Histogram representation of cosine scores for the 276 Headings (query: Ischemic Stroke)

4 Discussion

ARIANA is a system targeting a large scientific community: medical researchers, epidemiologists, biomedical scientific groups as well as junior researchers with focused interests. The tool can be used as a guide to broaden one's horizon by identifying seemingly unrelated entities. ARIANA provides the relations between query word(s) and the 276 Headings using literature data from PubMed. The key idea was to make the design efficient, modular and scalable. The framework can be expanded to incorporate a much larger set of Headings from the MeSH. In addition, a DDD system is implemented to group the ranked Headings into three groups for every query. The DDD system can be applied in other systems to improve the quality of IR. As a consequence of incorporating a context specific multi-gram dictionary, the sparsity of the data model is lower and the size of the dictionary is significantly smaller as compared to if all combination of English words were taken into consideration.

The features and functionalities of ARIANA are compared and contrasted with the State-of-the-art systems. In a recent survey where 28 applications were reviewed[1], five used clustering to group the search results into topics, another five used different techniques to summarize the results and present a semantic overview of the retrieved documents. The following are a subset of the scope and potential of these tools. The tools that are based on clustering are fundamentally different from ARIANA, while the rest of the tools have some similarities in their scopes and designs.

One of the systems, Anne O'Tate, [13] uses post processing to group the results into predefined categories such as MeSH topics, author names, year of publication. Even though this tool can be very helpful in presenting the results to the user, it does not provide the additional steps to extract semantic relationship.

The McSyBi[14], clusters the results to provide an overview of the search and to show relationship among the retrieved documents. It is reported that LSA is used with limited implementation details; furthermore, only the top 10,000

publications are analyzed. XplorMed[15] allows the users to further explore the subjects and keywords of interest. MedEvi[16] provides ten concept variables as semantic queries. XplorMed puts a significant limit (>500) on the number of abstracts to analyze. MEDIE[17] provides utilities for semantic search based on deep-parsing and, returns text fragments to the user. This is conceptually different from ARIANA. EBIMED[18] extracts proteins, GO, drugs and species, and identifies relationships between these concepts based on co-occurrence analysis.

Table 1. List of Headings ranked and grouped for query IS.

Medical Subject Headings	MeSH tree number	Relevancy	
		Score	Level
Cerebrovascular Disorders	C14.907.253	0.550	High
Vascular Diseases	C14.907	0.466	High
Mobility Limitation	C23.888	0.447	High
Myocardial Ischemia	C14.907	0.424	High
Athletes	M01.072	0.359	High
Hemorrhage	C23.550	0.245	High
Mycotoxicosis	C21.613.680	0.191	Possible
Hyperemia	C14.907.474	0.128	Possible
Neuroleptic Malignant Syndrome	C10.720.737	0.116	Possible
Arterial Occlusive Diseases	C14.907.137	0.106	Possible
Pain	C23.888.646	0.099	Possible
Intermittent Claudication	C23.888.531	0.096	Possible
Nervous System Neoplasms	C10.551	0.096	Possible
Personality	F01.752	0.094	Possible
Azotemia	C23.550.145	0.088	Possible
Preconception Injuries	C21.676	0.087	Possible
Cyanosis	C23.888.248	0.082	Possible
Emphysema	C23.550.325	0.081	Possible

Table 2. List of Headings at different scores for query IS.

Medical Subject Headings	Range of relevancy scores
Thyroid Diseases; Metabolic Syndrome X; Hypovolemia; Defense Mechanisms; Neurotoxicity Syndromes; Age Groups; Autoimmune Diseases of the Nervous System	0.08 to 0.041
Neoplasms; Minority Groups; Socioeconomic Factors; Alcohol-Related Disorders; Tumor Virus Infections; Peripheral Vascular Diseases; Hepatitis A; Intestinal Diseases, Parasitic	0.04 to 0.021
Dermatitis, Occupational; Physical Fitness; Neurocutaneous Syndromes; Socialization; Carbon Tetrachloride Poisoning; Mycoses; Muscular Diseases; Immunocompetence; Trauma, Nervous System; Movement Disorders; Bone Diseases, Endocrine; Heart Murmurs; Skin Temperature; Metabolism, Inborn Errors; Quality of Life; Arbovirus Infections; Child, Abandoned; Rheumatic Diseases; Arthritis, Rheumatoid	0.02 to 0.01

Among all reviewed tools[1], EBIMED is closely related to ARIANA ; yet, that system focuses only on proteins, GO annotations, drugs and species as concepts. ARIANA differs from EBIMED in a number of ways. First, ARIANA provides a systematic way of data stratification based on domain knowledge and application constraints. Second, it uses OM to create a DDD dictionary, which in turn produces a better model and also helps in finding crisp association of concepts. Third, it computes the associations based on higher order co-occurrence analysis and introduction of RM to present the results into an easy to use and understandable manner. In addition to that, since MeSH provides an hierarchical structure, ARIANA could be expanded to include a large number of Headings.

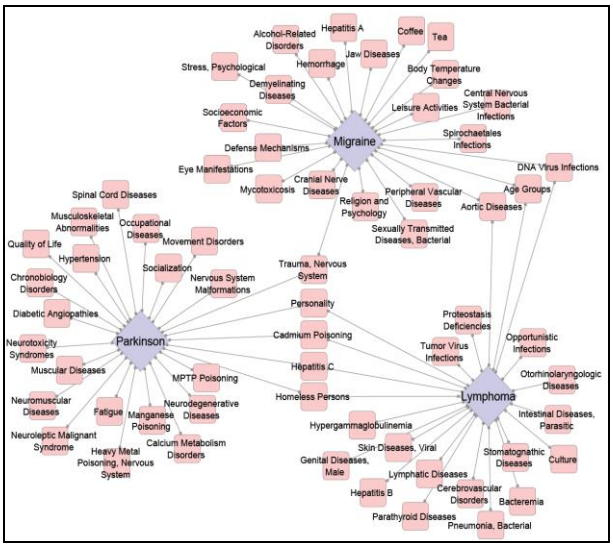


Figure 7. IR and KD using three queries. The queries are in blue diamonds, and the associated concepts are pink squares.

A careful analysis of the results (some not shown) is performed to identify the potentials and limitation of the tool and detect avenues for improvement. Based on the test cases, it was observed that ARIANA has the potential to extract hidden knowledge, generate hypotheses and help in KD. The test cases also suggest that the DDD system can be expanded for the data stratification. The DDD system could use a measure such as specificity to select the best representative Headings. In fact, in some cases the computed relationships missed the known associations (ex: between atherosclerosis & tobacco). This is likely due to the Heading selection. For instance "Tobacco Use Disorder" is used in ARIANA; yet, there are seven headings related to smoking that were not part of the model: tobacco, smokeless; tobacco smoke pollution; smoking; smoking cessation; tobacco use disorder; tobacco use cessation; and tobacco. Each of these are unique in the MeSH tree. Furthermore, some publications may only be linked to one or a few of these Headings. The automatic selection of Headings could minimize this problem.

5 Conclusion

KD and finding semantically related association through a global literature analysis is critical in the advancement of translational and collaborative research. Features such as scalability, context specificity and broad coverage are key in

building a system that is robust and can adhere to the high standards of interdisciplinary projects. ARIANA has the potential to bridge the gap between data and knowledge and advance our understanding of systems and biological phenomena at different level of granularity. We are currently working on expanding the data model of ARIANA and implementing a user-centric Web Services for this tool.

6 Acknowledgements

This work was supported by the Electrical and Computer Engineering Department and Bioinformatics Program at the University of Memphis, as well as by NSF grant NSF-IIS-0746790. Authors like to thank Muthukuri, KR and Faisal, FE for programming support.

References

[1] Z. Lu, "PubMed and beyond: a survey of web tools for searching biomedical literature.," *Database (Oxford).*, vol. 2011, p. baq036, 2011.

[2] "PubMed." [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed.

[3] A. Franceschini, D. Szklarczyk, S. Frankild, M. Kuhn, M. Simonovic, A. Roth, J. Lin, P. Minguez, P. Bork, C. von Mering, and L. J. Jensen, "STRING v9.1: protein-protein interaction networks, with increased coverage and integration.," *Nucleic Acids Res.*, vol. 41, pp. D808–15, 2013.

[4] J. M. Fernández, R. Hoffmann, and A. Valencia, "iHOP web services.," *Nucleic Acids Res.*, vol. 35, no. Web Server issue, pp. W21–6, Jul. 2007.

[5] R. B. Altman, C. M. Bergman, J. Blake, C. Blaschke, A. Cohen, F. Gannon, L. Grivell, U. Hahn, W. Hersh, L. Hirschman, L. J. Jensen, M. Krallinger, B. Mons, S. I. O'Donoghue, M. C. Peitsch, D. Rebholz-Schuhmann, H. Shatkay, and A. Valencia, "Text mining for biology--the way forward: opinions from leading scientists.," *Genome Biol.*, vol. 9 Suppl 2, p. S7, 2008.

[6] "Medical Subject Headings." [Online]. Available: http://www.ncbi.nlm.nih.gov/mesh.

[7] V. Abedi, R. Zand, M. Yeasin, and F. E. Faisal, "An automated framework for hypotheses generation using literature," *BioData Mining*, vol. 5. p. 13, 2012.

[8] S. T. Dumais, "Latent Semantic Indexing (LSI) and TREC-2," in *The Second Text REtrieval Conference (TREC-2)*, 1994, pp. 105–115.

[9] P. L. Antignani, "Treatment of chronic peripheral arterial disease," *Curr Vasc Pharmacol*, vol. 1, pp. 205–216, 2003.

[10] E. De Dominicis, M. Boschello, G. Trevisan, and R. De Nardis, "[Threatened paradoxical embolism: its direct visualization by two-dimensional echocardiography].," *G. Ital. Cardiol.*, vol. 25, no. 6, pp. 733–6, Jun. 1995.

[11] K. E. Kirkland, K. Kirkland, W. J. Many, and T. A. Smitherman, "Headache among patients with HIV disease: prevalence, characteristics, and associations.," *Headache*, vol. 52, no. 3, pp. 455–66, Mar. 2012.

[12] C. Infante-Rivard, E. Olson, L. Jacques, and P. Ayotte, "Drinking water contaminants and childhood leukemia.," *Epidemiology*, vol. 12, pp. 13–19, 2001.

[13] N. R. Smalheiser, W. Zhou, and V. I. Torvik, "Anne O'Tate: A tool to support user-driven summarization, drill-down and browsing of PubMed search results.," *J. Biomed. Discov. Collab.*, vol. 3, p. 2, Jan. 2008.

[14] Y. Yamamoto and T. Takagi, "Biomedical knowledge navigation by literature clustering.," *J. Biomed. Inform.*, vol. 40, no. 2, pp. 114–30, Apr. 2007.

[15] C. Perez-Iratxeta, P. Bork, and M. A. Andrade, "XplorMed: a tool for exploring MEDLINE abstracts.," *Trends Biochem. Sci.*, vol. 26, no. 9, pp. 573–5, Sep. 2001.

[16] J.-J. Kim, P. Pezik, and D. Rebholz-Schuhmann, "MedEvi: retrieving textual evidence of relations between biomedical concepts from Medline.," *Bioinformatics*, vol. 24, no. 11, pp. 1410–2, Jun. 2008.

[17] T. Ohta, K. Masuda, T. Hara, J. Tsujii, Y. Tsuruoka, J. Takeuchi, J.-D. Kim, Y. Miyao, A. Yakushiji, K. Yoshida, Y. Tateisi, and T. Ninomiya, "An intelligent search engine and GUI-based efficient MEDLINE search tool based on deep syntactic parsing," in *Proceedings of the COLING/ACL on Interactive presentation sessions* -, 2006, pp. 17–20.

[18] D. Rebholz-Schuhmann, H. Kirsch, M. Arregui, S. Gaudan, M. Riethoven, and P. Stoehr, "EBIMed--text crunching to gather facts for proteins from Medline.," *Bioinformatics*, vol. 23, no. 2, pp. e237–44, Jan. 2007.

Appendix

Table A1. The 276 Headings selected from MeSH ontology.

Category | MeSH tree number	
C	01.252; 01.252.100; 01.252.200; 01.252.300; 01.252.354; 01.252.377; 01.252.400; 01.252.410; 01.252.620; 01.252.810; 01.252.825; 01.252.847;; 01.703; 01.908; 02.081; 02.109; 02.182; 02.256; 02.325; 02.330; 02.440.420; 02.440.435; 02.440.440; 02.440.450; 02.440.470; 02.597; 02.705; 02.782; 02.800; 02.825; 02.839; 02.928; 02.937; 02.968; 03.105; 03.300; 03.335; 03.432; 03.518; 03.582; 03.600; 03.695; 03.752; 03.858; 03.908; 04; 05.116; 05.182; 05.321; 05.330; 05.390; 05.500; 05.550; 05.651; 05.660; 05.799; 06; 07; 08; 09; 10.114; 10.177; 10.228.440; 10.228.470; 10.228.662; 10.228.854; 10.281; 10.292; 10.314; 10.500; 10.551; 10.562; 10.574; 10.668; 10.720.150; 10.720.475; 10.720.606; 10.720.737; 10.886; 10.900; 12.294; 12.706; 12.758; 12.777; 12.777.419; 12.777.829; 12.777.892; 12.777.967; 13.351; 13.703; 14.240; 14.260; 14.907; 14.907.109; 14.907.137; 14.907.150; 14.907.184; 14.907.253; 14.907.320; 14.907.474; 14.907.489; 14.907.514; 14.907.585; 14.907.617; 14.907.940; 14.907.952; 15.378; 15.604; 17.300; 18.452.076; 18.452.174; 18.452.284; 18.452.394; 18.452.565; 18.452.584; 18.452.603; 18.452.625; 18.452.648; 18.452.660; 18.452.750; 18.452.811; 18.452.845; 18.452.915; 18.452.950; 18.654.301; 18.654.422; 18.654.521; 18.654.726; 18.654.940; 19.053; 19.149; 19.246; 19.297; 19.391; 19.642; 19.700; 19.874; 20.111; 20.111.163; 20.111.197; 20.111.199; 20.111.525; 20.111.567; 20.111.590; 20.111.759; 20.111.809; 20.543; 21.111; 21.447; 21.447.080; 21.447.270; 21.447.426; 21.447.653; 21.447.800; 21.613; 21.613.068; 21.613.097; 21.613.127; 21.613.165; 21.613.177; 21.613.380; 21.613.455; 21.613.589; 21.613.618; 21.613.647; 21.613.680; 21.613.705; 21.613.756; 21.613.809; 21.613.932; 21.676; 21.739.100; 21.739.225 OR 21.739.300 OR 21.739.635; 21.739.912; 21.866; 23.550.073; 23.550.081; 23.550.145; 23.550.274; 23.550.325; 23.550.414; 23.550.421; 23.550.429; 23.550.449; 23.550.455; 23.550.526; 23.550.568; 23.550.695; 23.550.737; 23.888.119; 23.888.144; 23.888.176; 23.888.192; 23.888.208; 23.888.248; 23.888.307; 23.888.369; 23.888.388; 23.888.447; 23.888.475; 23.888.512; 23.888.531; 23.888.550; 23.888.646;
D	01; 02; 03; 04; 05; 20; 25;
F	01.393; 01.525; 01.752; 02.830.071; 02.830.855; 02.830.900; 02.880; 02.940;
G	03.180.134; 12.248; 12.460; 12.470; 13.750.829.855; 13.750.844; 14.760; 14.930; 14.940;
I	I01.800;I01.880.143;I01.880.298;I01.880.371;I01.880.552;I01.880.787;I01.880.813;I01.880.840;I02;I03;I03.350;I03.450;I03.621;I03.883;
J	01.516; 02.200.100; 02.200.300; 02.200.325; 02.200.700; 02.200.712; 02.200.806; 02.200.900; 02.500;
M	01.060; 01.066; 01.072; 01.085; 01.097; 01.102; 01.106; 01.108; 01.111; 01.120; 01.135; 01.142; 01.150; 01.169; 01.189; 01.276; 01.325; 01.385; 01.729; 01.755; 01.785; 01.848; 01.873;
N	01.824; 06.230

A Software Toolkit for Stock Data Analysis Using Social Network Analysis Approach

Junyan Zhang[1], **Donglei Du** [2], **and Weichang Du**[1]
[1]Faculty of Computer Science,[2]Faculty of Business Administration
University of New Brunswick, Fredericton, New Brunswick, Canada

Abstract— *In this paper, we design an online analytical toolkit benefiting from the domain of Social Network Analysis. The objective is to provide a network centric perspective for analyzing stock data in facilitating portfolio management. The core process of this toolkit is to build a social network of stocks from NYSE and NASDAQ. In this network, each node is a stock and the weight of an edge is decided by the correlation coefficient calculated based on the historical daily returns between the two stocks involved. After filtering less significant nodes from user definitions, a proposed portfolio index is then generated to be an appropriate managed portfolio to users. This approach is implemented as a client-server online toolkit, and finally evaluated through a case study on simulating trend of real portfolios (DJIA). The more matched peaks and valleys between both lines indicates the more similarities of both portfolios.*

Keywords: Social Network Analysis, Big Data, Software Toolkit

1. Introduction

Over their long history, various stock exchange markets (like *NYSE* and *NASDAQ*) have accumulated enormous data, quickly entering the new big data era. This massive stockpile of information opens innumerable ways to improve efficiency and decision-making, and presents massive opportunities for investors to discover new stock investment strategies.

Stock data analysis is very complicated and multifaceted in recent times. The research on stock data is deserved because the analysis result based on the stock price history of a target company can provide future trend prediction for the prospective buyers, offer insight into this company's prosperity, and make significant contributions to forecast the performance and the volatility of this stock. The prediction of a stock's future trend is a complex problem which is interested by financial theorists and investors, and they have been dealing with this issue for many years. There are several software toolkits such as *R-Studio* and *SPSS* which help professional experts investigate on this research topic via some statistical models, using techniques in machine learning and data mining to propose heuristic solutions. However, when it comes to non-professional investors, it is too complicated for them to do such analysis on these toolkits.

Since researchers and analyzers have attached more importance to the relationship among numerous stocks from markets, social network analysis seems to be a new approach in stock big data analysis which contributes more to the investigation of the properties of market entities by generating a social network of all stocks from a stock exchange market. Social network analysis (*SNA*) is a multi-disciplinary subject with many applications in diverse fields of economic life[1]. A social network is a social structure made up a set of social actors(such as individuals or organizations) and a set of the dyadic ties between these roles. The network is used to describe a social structure determined by such interactions, and the edges links any given social unit represent the convergence of various social contacts from that unit.

So the motivation of our proposed approach is to design an online toolkit, Social Network Analysis Online Toolkit (*SNAOT*), which is aiming to provide an innovative way for professional experts and stock investors to do stock big data analysis based on social network analysis. By using this toolkit, not only the professional financial experts, but also the non-professional stock investors can create their personal social network models and design their own applications based on a generated network.

This paper structured as follows. Section 2 introduces the necessary knowledge on basic social network theory, and the Pearson and Spearman correlation coefficients. Section 3 describes *SNA* based modeling for financial data, which is also the design model of this toolkit. Next in Section 4 shows the high-level design of the toolkit, including requirements and system architecture. Section 5 applies the developed toolkit to the stock market as a case study by constructing a stock network, and then carrying on various network analysis implemented via language R. Finally, Section 6 summarizes this paper work and discusses further research work.

2. Background

2.1 Mathematic Model of Social Network

We consider a large group N of nodes, i =1,2,...,n, the relationships among them are distributed along a social network g represented by adjacency matrix G. For each pair of nodes i, j, $g_{ij} = 1$ if node i has a direct relationship with node j, then there exists an edge between i and j, and

$g_{ij} = 0$ vice versa. For most of the analysis, we assume that the graph is un-directional, i.e. $g_{ji} = g_{ij}$.

Fig. 1 is a network example made up by six nodes and several edges between them. From the figure we create a symmetric matrix as shown in Fig. 2 which describes the relations between different nodes.

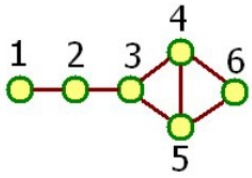


Fig. 1: A Social Network Example

$$G1 = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 \end{pmatrix}$$

Fig. 2: The Symmetric Matrix of Fig. 1

Each element in matrix *G1* measures the existence of a edge in the network. We define deg_i to represent the number of edges at i, so $deg_i = \sum_j g_{ij}$. If the network is directed, we should distinguish the number of nodes which node i points with the number of nodes which i points towards, i.e. the *outdeg* and *indeg*. Formally, $indeg_i = (1^T G)_i^T = \sum_j g_{ji}$ and $outdeg_i = (G1)_i = \sum_j g_{ij}$. In this network, the *deg* of all nodes, which can be named as the total degrees of the social network in Fig. 4 is shown in Table. 1. As shown in the table, when the total degree of a node is larger, we say that this node has more significant meaning.

Table 1: Total Degrees of Fig. 1

Nodes	Total Degree
1	1
2	2
3	3
4	3
5	3
6	2

2.2 Correlation Coefficients

As is shown in the Section 1, a social network is consist of a set of characters with a set of edges connecting each pair of nodes. In this special application domain, each character in a social network represents a stock. For any pair of stocks i and j, the presence of an (undirected) edge (i, j) means there is a correlation between i and j, and the strength/weight of this correlation is measured by either the Pearson or the Spearman correlation between the two stocks i and j. The correlation of two sets of data shows whether they are related or not, how strongly, and in what way. In statistics, the relationships among variables are denoted by correlation coefficients. In the following subsections we describe two methods of calculating correlation coefficients adopted in this thesis work.

2.2.1 Pearson Correlation Coefficient

The Pearson product-momentum correlation coefficient is a measure of the linear dependence between two random variables (real-valued vectors). Historically, it is the first formal measure of correlation. Nowadays, it is still one of the most widely used measures of linear correlations[3].

The Pearson linear correlation coefficient γ of two variables *X* and *Y* is formally defined as the covariance of both variables divided by their product of standard deviations which acts as a normalization factor. Equation (1) was defined by Karl Pearson:[4]

$$\gamma = \frac{\sum_{i=1}^{n}(X_i - \overline{X})(Y_i - \overline{Y})}{\sqrt{[\sum_{i=1}^{n}(X_i - \overline{X})^2][\sum_{i=1}^{n}(Y_i - \overline{Y})^2]}} \quad (1)$$

where $\overline{X}$ represents the average of *X* ($\overline{X} = \frac{1}{n}\sum_{i=1}^{n} X_i$), and same for $\overline{Y}$.

2.2.2 Spearman Rank Correlation

Given two time series *X* and *Y*, the Pearson product-moment correlation coefficient is calculating linear dependence between the two time series. Spearman rank correlation, measures the strength of monotone relations between two time series. This correlation coefficient is distribution-free, it does not assume any probability distribution of the original data. Compared with Pearson correlation coefficient, the Spearman correlation coefficient ρ lays emphasis on ranked variables and can be computed as follows. There are two time series *X*= $\{X_i$, *i*=1,...,*N*$\}$ and *Y*= $\{Y_i$, *i*=1,...,*N*$\}$. A new rank variable R_i^X is defined by *X*, where R_i^X is equal to the order of the sequence X_i. A new rank variable R_i^Y is defined for *Y*, where R_i^Y is equal to an element of the sequence *Y*. Equation (2) defines Spearman rank correlation coefficient:[5]

$$\rho = \frac{\sum_i (R_i^X - \overline{R}^X)(R_i^Y - \overline{R}^Y)}{\sqrt{\sum_i (R_i^X - \overline{R}^X)^2 (R_i^Y - \overline{R}^Y)^2}} \quad (2)$$

where $\overline{R}^X$ and $\overline{R}^Y$ are means of two corresponding variables.

3. SNA Based Modeling for Stock Data

3.1 Social Network Creation

There are several items in each downloading table which records a stock's history prices. In our social network modeling, we choose "Adjust Close" to be the unique characteristic of each stock node. All corporate actions, such as stock splits, dividends and distributions will be amended to this item. So this item offers a useful standard to check the historical returns, and also gives analysts an accurate representation of a stock's equity value[7].

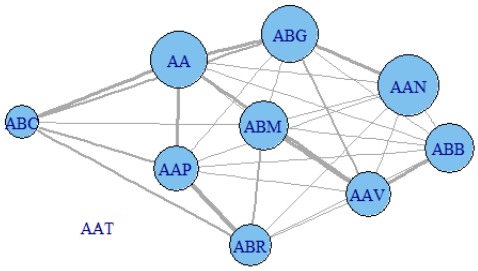


Fig. 3: Social Network Graph Instance

Fig. 3 Assuming there is a group of 10 stocks from the market, Fig. 3 shows a social network that consists of these stocks. In this network, 9 stock nodes are connected with certain degrees of links, and only one stock is no circle and separated from others. The links which connect stock nodes or vertices are called edges. The weight of an edge is defined by a correlation value calculated by the historical data of the two stocks in a fixed time period. The social network is a non-directional network. A thicker edge in the network represents a higher correlation. The radius of a node is determined by the total of correlation values with other nodes in the network.

In our social network model, the weight of an edge is defined by the correlation between the linked pair. Furthermore, the accurate definition of an edge weight should be the correlation of two stocks adjusted closed prices within the fixed time period. However, each stock has a different price range. To normalize the price range, we use daily returns, which is widely used in quantifying stock trend performance, to replace the "Adjust Close" in the model.

Based on Equation (3), the two input vectors of adjusted close prices must have the same data size, for calculating both Pearson and Spearman correlations. Different vector sizes will lead to zero denominator of the correlation equation. For this concern, we set all correlations of a node which has a shorter data size to be 0. For example, the single node *AAT* which is separated with others in Fig. 3 has a shorter data size.

$$DailyReturn = \frac{Closed\_Today - Closed\_Yesterday}{Closed\_Yesteday} \tag{3}$$

We assume that there are two sequences of daily returns of two stocks: *X* and *Y*. The following table shows the pseudo code of calculating node weight algorithm between them[8].

Algorithm 1 Pseudocode of Calculating Edge Weight

$sum\_sq\_x \leftarrow 0$
$sum\_sq\_y \leftarrow 0$
$sum\_coproduct \leftarrow 0$
$mean\_x \leftarrow x[1]$
$mean\_y \leftarrow y[1]$
for $i \leftarrow 2$ *to* N **do**
  $sweep \leftarrow (i-1)/i$
  $delta\_x \leftarrow delta\_x + x[i] - mean\_x$
  $delta\_y \leftarrow delta\_y + y[i] - mean\_y$
  $sum\_sq\_x \leftarrow sum\_sq\_x + delta\_x * delta\_x * sweep$
  $sum\_sq\_y \leftarrow sum\_sq\_y + delta\_y * delta\_y * sweep$
  $sum\_coproduct \leftarrow sum\_coproduct + delta\_x * delta\_y * sweep$
  $mean\_x \leftarrow mean\_x + delta\_x/i$
  $mean\_y \leftarrow mean\_y + delta\_y/i$
end for
$pop\_sd\_x \leftarrow sqrt(sum\_sq\_x/N)$
$pop\_sd\_y \leftarrow sqrt(sum\_sq\_y/N)$
$cov\_x\_y \leftarrow sum\_coproduct/N$
$weight\_x\_y \leftarrow cov\_sd\_y/(pop\_sd\_x * cov\_sd\_y)$

3.2 Subgraph Generation

As mentioned, correlation coefficient is a value which is defined to describe the relationship between variables in statistical terms. The absolute value of a correlation coefficient defines the magnitude of the relationship between a pair of variables. The following are some basic properties[11].

- The value of a correlation coefficient ranges from -1 to 1.
- The strongest linear correlation coefficient is -1 or 1.
- The weakest linear correlation coefficient is 0.

In general, the higher the absolute value of a correlation coefficient, the stronger relationship between the two variables in the correlation. The following tables show some classifications of values of correlation coefficients. Table. 2 shows a categorisation defined by Dancey and Reidy [6].

The Dancey and Reidy Classification of correlation in Table. 2 indicates that not every correlation of a network deserves to investigate. After creating the network from a stock data set is obtained, some filtering mechanisms can be adopted to remove those nodes with weak correlation values

Table 2: Dancey and Reidy's (2004) Classification

Value of the Correlation Coefficient	Strength of Correlation
1	Perfect
0.7-0.9	Strong
0.4-0.6	Moderate
0.1-0.3	Weak
0	Zero

and then to generate a specific subgraph from the original network. One such filtering solution is to set a threshold criteria to delete those less important nodes. We assume that all edges with small absolute correlation values are removed, and then generate a subgraph with the existing nodes and a subset of edges. Fig. 4 shows such subgraph which is generated from a correlation threshold criteria of 0.3.

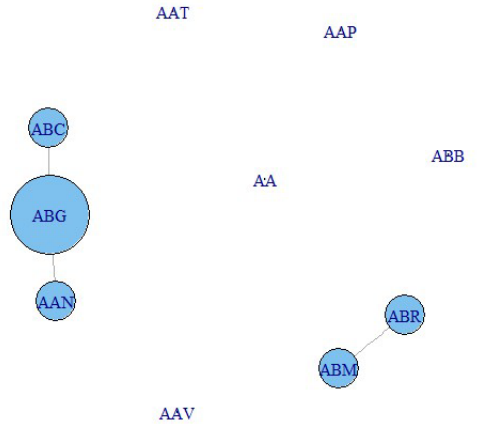


Fig. 4: A Generated Subgraph after Data Filtering at 0.3

In the following, we compare the original network as shown in Fig. 3 and the generated sub-network as shown in Fig. 4.

- Nodes base: More discrete nodes are shown in the subgraph than the original network. The biggest node is different, that means the biggest node in the original network may not be the node which has most significant correlations with others.
- Edge base: The number of edges is reduced when a threshold criteria for filtering is set. The diameter of the node also differs because of the different threshold criteria.

3.3 Market Index Trend Simulation

With different subgraphs generating from various social networks, researchers can do specific financial analysis on those subgraphs. In our toolkit, we extend our financial analysis application to the simulation of market index trend.

DJIA is an index of 30 large publicly owned companies and widely recognized to be indices of the whole US markets[9]. This application is to choose several stocks from a subgraph, which have the maximum or minimum total of correlations. With these stocks making up an index, a user can track the trend of this index in history and compare it with *DJIA*. The purpose of this application is to verify if a specific subgraph from the entire social network can simulate the historic trend of *DJIA* based on social network analysis.

4. Toolkit Functionality Design

The Social Network Analysis Online Toolkit Framework (*SNAOTF*) can be viewed as four modules as shown in Fig. 5:

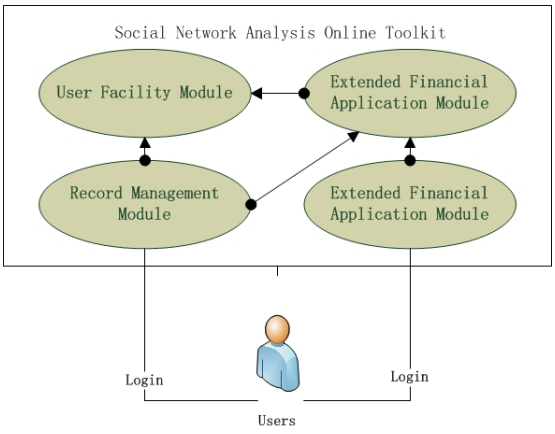


Fig. 5: Toolkit Framework Design

- Social Network Analysis Module
- Extended Financial Application Module
- Record Management Module
- User Facility Module

SNAOTF provides an efficient way to analyse social network modeling and at the same time provides a systematic way to manage and visualize the analysis results. In the following subsections, we discuss these modules in details in terms of their roles, responsibilities, and relationship among them.

4.1 Social Network Analysis Module

The responsibility of this module is to create a social network of the whole stock markets and generate subgraphs from user definition. The output of this module can be used as inputs of extended application module. In general, the social network analysis module contains two services: network creation service and subgraph generation service.

The network creation service deals with the requirement of creating the overall social network. Each stock is defined as a node in the network, and each edge which links two given nodes is defined by the correlation coefficient between the corresponding time series of their daily returns[5]. So the correlation calculating method and time range of the network are input parameters to this service. Each new network modeling requirement will follow the same creation process

in this module, and this service will generate the target social network and simultaneously save the data of network into the record management module for future use.

The subgraph generation service needs certain input parameters to configure the subgraph generation, including the correlation filtering threshold criteria and the number of required nodes in the result index. Moreover, if users want to do collaborative activities with other users, they can set the authority of their subgraphs to be public. Then this service will generate the required subgraph and save the result subgraph and authority into the record management module for future use.

4.2 Extended Financial Application Module

This module provides pre-designed functionalities to support stock analysis applications based on social network analysis results. The input to this module is a generated subgraph from the *SNA* module or a saved subgraph from the record management module, and the output is displayed online or transferred to the user facility module for visualization or downloading. In general, the extended financial application module contains two services: built-in application service and user-defined application service.

There are two built-in stock analysis applications supported by the built-in application service: Portfolio Investment Yield Analysis and Market Index Trend Simulation. For each application, the service integrates and performs the variable definition, data transformation, and calculation process. To use this service, users need to specify some options to configure applications.

The user-defined application service is designed for supporting user-defined applications. The service provides a text area using generated subgraphs for users to write their own scripts for their own specific analysis tasks. The result of executing user-defined scripts can be displayed online or recorded for future use.

4.3 Record Management Module

There are four record management services designed in this module to record actions in *SNAOT*. All operations from the other three modules will be recorded in this module. Each management service role is described as follows.

- User information Management: Any one who wants to use the toolkit must first register as a user and login to the toolkit. This service records users' account information and statuses.
- *SNA* Record Management: All created social networks and generated subgraphs are stored in this service for future uses and research.
- Application Record Management: All results of executing built-in and user-defined applications are stored in this service for future uses and research.
- User Facilities Record Management: The results of executing stock data trace service and stock data comparison service are stored in this service for future uses and research.

4.4 User Facility Module

This module provides several services to help users to support non-SNA based stock analysis: stock price trace service, stock data comparison service and data visualization and downloading service.

The functionality of tracing historical stock data is a common service of realtime stock analysis systems. The stock price trace service in this module supports two patterns of different stock data search: list by market search and advanced search.

For stock price comparison, the stock data comparison service supports comparison among several stocks and application results on some specific items. The comparison result can be presented in a chart or a line graph. By using this service, users can compare several stocks' historical prices to analyse and help to predict the trend of some stocks in the future.

The data downloading service handles the functionality of packaging created networks and subgraphs from *SNA* processes, as well as results of applications from extended application module, let users download these packages for further analysis on their computers. The data visualization service can be used to draw line graphs of results generated from extended application module. Such visualized graphs can give users visualized comparisons and an obvious visual perception versus a data table with data listed.

5. Case Study

The Dow Jones Industrial Average (*DJIA*), is a stock market index, which is created by Wall Street Journal editor and Dow Jones & Company co-founder Charles Dow [9]. *DJIA* represents trading transactions of the selected 30 publicly owned component companies in a standard market day[10]. Since *DJIA* based investments are widely accessible in equities through exchange-trade funds(*ETFs*), this case study can also verify if the mock index can be one of the methods to represent stock market activities. The objective of this case study is to discover one or more subgraphs from the overall social network of stock market data in a fixed time period which consists of different sets of components than *DJIA* but have similar trading trend history to *DJIA*'s.

5.1 Solution Design

The input to this case study is a social network of entire stocks from *NYSE* and *NASDAQ* exchange markets. Each node in this network is a stock entity, and the correlation coefficient value of any two stocks history trends in a limited time range shows the relationship between the both nodes.

The network can be shown in a 2-dimensional correlation coefficient matrix.

The first task is to filter out insignificant correlation values from the matrix. As described in Section 3, a correlation coefficient has its range of strength; the bigger value, the more strength correlation.

The next task is to sort the filtered matrix by selecting some nodes to structure the mock index. There are several sorting algorithms for ranking nodes. In our case, we choose the following two sorting methods. The first one is ranking nodes from high to low by calculating total correlations of a node. Since every node has one vector of correlation coefficients with other nodes, we add up all edge weights of a node. The higher sum of correlations a node has, the more significant will it be in the network. We consider those nodes with most significant correlations in the network to be representatives of the market, and compare their history trends to *DJIA*'s. In the other method, we regard 30 *DJIA* components as an entity, and capture a submatrix with correlations only related to these 30 components. Using the submatrix we sort the correlation totals from high to low. We consider that the higher sum of a stock indicates the more total correlations with *DJIA* components. A mock index with the nodes ranked by this sort method may also has similar trends with *DJIA*'s. The task of the process is to use the generated mock index to compare with *DJIA*. Generally, it's impossible for our mock index to trace every point with DJIA's, in this case, we try to test overlapping inflexion points of both trend graphs from the mock index and *DJIA*, to find whether the mock index reaches peaks or lows at almost same days with *DJIA*'s.

The final task is to find the mock index which has the most similar trend to *DJIA*. For this task, we do not need to traverse every potential mock index to search. Instead, we can first set an subset and add a constant number of stocks into the mock index. Each time we modify the index, we count the same inflexion points of the two graphs, and set this value to be the characteristic value. Then we draw a line graph with the inflexion points to be the Y-axis, the quantity of stocks in the index to be the X-axis. We record the maximum and check the peaks and lows of the generated line. If the group of peaks and lows are both monotonously going down, the trend of this line is also monotonously going down, and the quantity of index stocks which gives us the maximum will be the optimal size of index. Otherwise, we record this maximum and move on to next subset to repeat the above operations.

5.2 User-define Application Service Realization

Using the built-in application service, users can test different correlation criteria or sorting methods one by one manually. To do it automatically, we can use the user-defined application service to find the optimal subgraph by defining and running a script. In this subsection, we describe the script to find the optimal subgraph in terms of flexible quantity of nodes or components in the mock index and correlation criteria ranges.

To design such a script, only the methods which have been already embedded in the built-in application service can be chosen to add into a script.

Fig. 6 shows the script. The correlation filtering criteria is set to the range from 0.4 to 0.9, and the toolkit generates a subgraph once each 0.1 subset. We also set a subset of 20 stocks, which means to check the trend of same inflexions for every 20 stocks. If the trends of peaks and valleys of the two indexes are both monotonously going down, the pre-searched maximum volume of stocks will be the optimal size of the mock index. The sorting method is set using the 2 sorting methods on stock ranking. The time range for comparing the index trends with *DJIA* is still defined in year 2012.

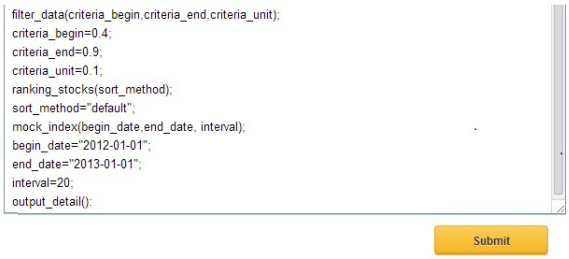


Fig. 6: Case Study: Autoset Criteria

Fig. 7 shows the result of executing the script. The optimal size of the index of the index this case is 17, using the sorting method of "Sort by relations with *DJIA*", and 94 out of 124 inflexions are matched in year of 2012.

User_define App Result

Method	Criteria	Nodes	Same_inflexion	Total_inflexion
sort in network	0.4	41	91	124
sort in network	0.5	18	90	124
sort in network	0.6	73	89	124
sort in network	0.7	34	82	124
sort in network	0.8	35	76	124
sort in network	0.9	16	77	124
sort by djia	0.4	3	83	124
sort by djia	0.5	17	94	124
sort by djia	0.6	22	91	124
sort by djia	0.7	6	85	124
sort by djia	0.8	0	0	124
sort by djia	0.9	0	0	124

Fig. 7: Case Study: Autoset Result

We choose the first line result and Fig. 8 shows the generated line graph from the visualization service. From the trends of both indexes we conclude that both them have a similar history trend, which exactly the mock index has 91 same valleys of peaks from the total 124 inflexions of *DJIA* index.

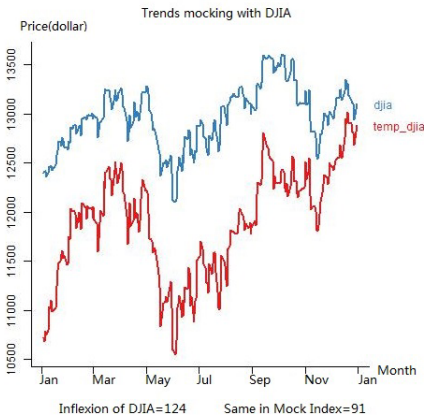


Fig. 8: Case Study: Mocking Trends in Graphs

6. Concluding Remarks

This paper describes the design and implementation of the online software toolkit based on social network analysis approach to facilitated stock data analysis, namely Social Network Analysis Online Toolkit, or "*SNAOT*".

In the thesis, we first gave the description of *SNA* based modeling for stock data. In this social network modeling, we chose "Adjust Close" attribute as the key element of each stock node. We then discussed the details about nodes and edges of the social definitions of the social network, and how to calculate the weights of edges. After the creation of social network, we moved on to generation of subgraphs from the social network by filtering out insignificant nodes based on generated subgraphs, we can apply stock analysis applications. The toolkit provides several built-in applications, and also supports user-defined applications in R/Java based scripts.

We then gave a description of the design of the toolkit. The system design of this toolkit consist of four modules for processing user requests. The core of the four modules is the social network analysis module. This module is for creating the social network of the stock data from *NYSE* and *NASDAQ* stock markets. This module also help users to define, choose and generate diverse subgraphs from the full network. The extended application module includes the following built-in application: Market Index Trend Simulation and Portfolio Investment Yield Analysis, as well as software components to support user-defined applications. The record management module is the center of *SNAOT*. There are four record management services to record actions in *SNAOT*. All operations from the other three modules will be recorded in this module. The user facility module provides several services to help users do non-SNA based stock analysis.

In the case study, we investigated *DJIA* mock stock index using the toolkit. The input of this case study is the social network of stocks from *NYSE* and *NASDAQ* exchange markets. First task in this work flow is to filter out insignificant correlation values of the network, by using the two sorting methods to sort nodes from the network. We then select some stocks with top total correlations to structure the mock index. To simulate the *DJIA*, we used the same formula of *DJIA* to calculate daily price of the mock index and then compared with DJIA's historic trend, in terms of matching peaks and valleys.

Currently, our toolkit only supports *NASDAQ*, *NYSE*, FTSE100 and Hang Seng stock markets. The next step is to add more stock markets into our toolkit, and users can design their own network components and create social networks based on their choices of stocks. Another extension is the functionality of generating dynamic mock indexes. The current mock index from the subgraph generation service is static, which one can use to mock the trend of a target index in a time range. For dynamic mock indexes we need to investigate an algorithm which will automatically generate subgraphs with dynamic time range to optimize the index components during the simulation. This algorithm may lead to match more peaks and valleys of a target index trend. In our implementation, we deployed this toolkit on a single server.

References

[1] D. Bgenhold, "Social network analysis and the sociology of economics: Filling a blind spot with the idea of social embeddedness" *IEEE American Journal of Economics and Sociology.*, vol. 72, pp. 293-318, Apr. 2013.

[2] F.Bloch, and N.Querou, "Pricing in social networks" *IEEE Games and Economic Behavior.*, vol. 80, pp. 243-261, Jul. 2013.

[3] P. D.Lena and L.Margara, "Optimal global alignment of signals by maximization of pearson correlation" *IEEE Information Processing Letters.*, vol. 110, pp. 679-686, May. 2010.

[4] P.Dutilleul, J.D.Stockwell, D.Frigon and P.Legendre, "The Mantel Test versus Pearson's Correlation Analysis: Assessment of the Differences for Biological and Environmental Studies" *IEEE Journal of Agricultural, Biological, and Environmental Statistics.*, vol. 5, pp. 131-150, Jun. 2000.

[5] O.Shirokikh and G.Pastukhov and V.Boginski and S.Butenko, "Computational study of the US stock market evolution: a rank correlation-based network model" *IEEE Computational Management Science.*, vol. 10, pp. 81-103, Jun. 2013.

[6] Dancy, C. P. and J. Reidy, "Statistics without maths for psychology" *IEEE Statistics without maths for psychology.*, 2004.

[7] Adjusted Closing Price on Investopedia. [Online]. Available: http://www.investopedia.com/terms/a/adjusted_closing_price.asp

[8] Talk:Correlation on Wikipedia. [Online]. Available: http://en.wikipedia.org/wiki?title=Talk:Correlation

[9] Dow jones industrial average on Wikipedia. [Online]. Available: http://en.wikipedia.org/wiki/Dow_Jones Industrial_Average

[10] Sullivan, Arthur and Steven M. Sheffrin, "Economics: Principles in action" *IEEE Upper Saddle River, New Jersey 07458: Pearson Prentice Hall.*, 2003.

[11] Linear correlation coefficient on Stat Trek. [Online]. Available: http://stattrek.com/statistics/correlation.aspx

[12] H. Markowitz, "Portfolio Selection" *IEEE The Journal of Finance.*, vol. 7, pp. 77-91, Nov. 1952.

A Visual Decision-Guided Analytics Tool for Finding the Viable Shortest Path over Geospatiotemporal Data

Chun-Kit Ngan
Engineering and Information Science Division
Great Valley School of Graduate Professional Studies
The Pennsylvania State University
Malvern, PA, USA
cxn20@psu.edu

***Abstract*— We propose a Visual Decision-Guided Analytics tool that combines quantitative analysis and qualitative methodologies to determine the viable shortest path over geospatiotemporal data for rescue and recovery missions. Specifically, we first extend the object-oriented spatial-temporal data model as a multidimensional OLAP cube that enables military operators to analyze unified geo-data objects from multiple dimensions, such as time, space, and location, to help them make a better decision on routes. Second, we enhance the capability of the PostGIS query which allows the operators to (1) simulate the occurrence of events, (2) visualize and report geo-data objects, and (3) solve decision optimization problems based upon their events of interest. Third, we integrate optimization programming into geo-data visualization to determine the viable shortest path for rescue and recovery missions. This integration enables military units to make a visual decision on routes based upon the time-distance optimality. Finally, we develop a visual display that enables the operators to analyze other crucial factors, e.g., vehicle types, weather severity, soldiers' specialties, etc., which are required to be interpreted by human perception, cognition, and knowledge to select the best path among all the viable routes for rescue and recovery missions.**

Keywords*— *decision-support; visual-analytics; optimization; query; OLAP-model

I. INTRODUCTION

In the past decade, military units in nations successfully analyzed integrated geo-data objects to execute decision-making operations. Consider a range of peace-keeping operations that nations' military forces need to accomplish in order to maintain peace in their countries. These operations include large scale protection operations, peace enforcement, nation assistance, and freedom of navigation [1]. To accomplish these operations, it is important for each military unit, such as Army, Navy, Marine, and Air Force, to cooperate with one another to analyze unified geo-data objects, e.g., satellite imagery, vector/raster maps, temporal data, 3D objects, etc., to support optimal decision-making based upon different unit forces' data sources. One critical operation that military troops need to determine is to find the viable shortest path to reach a target location for rescue and recovery missions so that medical and health services can be efficiently delivered to victims who can receive supports and supplies in the aftermath of natural disasters. To support such a decision-making operation, it is important for military researchers to develop a geo-data analytical methodology that is reliable and useful for the operation. The main challenge in such a methodology is how to *expressively* and *effectively* model and analyze those unified geo-data objects such that diverse military units can analytically make a concerted decision together. This is exactly the focus of this paper.

To support such a decision-making process, a number of researchers has proposed and developed different approaches to model geo-data objects, which can be roughly divided into two categories: On-Line Transaction Processing (OLTP) and On-Line Analytical Processing (OLAP). The former OLTP approach mainly focuses on facilitating and managing transaction-oriented applications that are used for data entry and retrieval processing. For example, Li and Cai [2] proposed an OLTP-based object-oriented spatial temporal data model, from which geo-data objects can be described by a theme, space, and time. Although this data model is efficient for a fast transactional processing analysis, it is not designed for supporting decision-making analytics. The latter OLAP approach enables military operators to analyze multidimensional data interactively from multiple perspectives. For instance, Vaisman and Zimanyi [3] proposed a MultiDim model based on a Spatial On-Line Analytical Processing (SOLAP) cube that can enable military operators to use multi-dimensional views to analyze geo-data objects, from which the operators can gain a more complete decision-making insight. However, since this spatial OLAP solution does not model real-world objects' properties and operations based on the object-oriented paradigm, it is hardly suitable for modeling objects' interactions.

In addition to the data modeling problem, an analytical approach on geo-data objects is another important issue that we need to address. Currently, a number of data analytics approaches has been proposed and developed to analyze geo-data objects. Data mining techniques [4, 5] are computational algorithms that analyze raw data over a terrain to extract valuable information. Some algorithms include rule mining (e.g., the Apriori algorithm) [6], dimensionality reduction methods (e.g., principal components analysis) [7], supervised learning (e.g., decision trees, support vector machines, neural networks, etc.) [8], and unsupervised learning (e.g., cluster analysis) [9]. However, these mining techniques often require expert human interpretation and supervision on data to deliver the results, which are not suitable for a quick decision-making operation. To support military troops to effectively make a better decision, i.e., determining the viable shortest path for

rescue and recovery missions, researchers proposed and developed data visualization tools that expressively display the learned results from the mined geo-data objects to support the analysis. This technique is called visual analytics or visual data mining [5, 10] that combines the advantages of both visualization and data mining methodology.

However, due to the high volume and the large size of the available data, the visual analytics approach is still very cumbersome to be used and can even overwhelm military operators during their decision-making operations. To solve this heavy data problem, a number of operation research methods [11], i.e., Mathematical Programming, e.g., Mixed Integer Linear Programming, Mixed Integer Nonlinear Programming, etc., are revised and streamlined to play a key role in this endeavor. The operation research methods help military units formulate precise objectives, e.g., minimize the cost of traveling time and distance, as well as specific constraints, e.g., the vehicles' speed limitations, imposed on the solution [12]. Once optimal values of decision parameters (e.g., select/unselect a node and/or a route segment along a path) are learned by optimization algorithms, such as Dijkstra's [13], Simplex [11], and Branch-and-bound [11], military decision makers can use the learned parameters to determine the shortest path.

Recently, a number of more advanced shortest path learning algorithms has been proposed and developed. The Smoothest Path Algorithm (SPA) [14] is to find the shortest surface path over a terrain in terms of distance and slope rather than the Euclidean distance only. The Shortest Path Algorithm for a Fixed Start Time [15] is to compute the shortest path either for a given start time or to search the start time and the path that results in the least travelling time. However, these approaches do not consider taking advantages of one another to determine the shortest route based upon the distance, slope, time schedule, and other possible terrain obstacles together. In addition, the learned paths computed by those algorithms only reflect an optimal reality under some environmental constraints rather than address the entire phenomenon, especially after natural disasters. For instance, the shortest path delivered by those models and algorithms are not considered to include other crucial factors, such as vehicle types, weather severity, moving entities, and soldiers' specialties, during the learning computation. These neglected factors definitely impact the traveling time and distance that still require human perception, cognition, and knowledge for making a final decision.

Thus this paper focuses on bridging the above research gaps among (1) the geo-data modeling, (2) the viable shortest path determination, and (3) the visual analytics. Specifically, we propose a **V**isual D**E**cision-**G**uided **A**nalytics (VEGA) tool that is a unified decision-making application that combines both advantages of quantitative analysis (i.e., optimization programming in operation research) and qualitative methodologies (i.e., geo-data visualization in data analytics) to determine the viable shortest path for rescue and recovery missions. This application supports reporting, simulation, visualization, and decision optimization [16] over geospatiotemporal data. Technically, this tool enables military units to (1) collect geo-data objects from multiple data sources, (2) conflate the diverse objects into unified data sets, and (3) perform analysis on these sets. First, to support a better decision-making, we extend the object-oriented spatial-temporal data model as a multidimensional OLAP (star-schema) cube, i.e., a Star-based Geo-Object-Oriented SpatiotEmporal (S-GOOSE) data model, which combines the advantages of both OLTP and OLAP approaches. This S-GOOSE data model is an object-relational-based cube that enables military operators to analyze unified geo-data objects from multiple dimensions, such as time, space, and location, to help them make a better decision on routes. Second, we enhance the capability of the PostGIS query [17], named Geo-Query Language (GQL), to support the S-GOOSE data cube analysis, which enables military operators to simulate the occurrence of events, to visualize and report geo-data objects, as well as to solve decision optimization problems based upon their events of interest. The reason to select the PostGIS query to be extended is because the GIS objects supported by the PostGIS query are a superset of the "Simple Features" defined by the OpenGIS Consortium (OGC) [18]. The PostGIS query supports all the objects and functions specified in the OGC "Simple Features for SQL" specification [17]. Third, we integrate optimization programming into geo-data visualization to determine the viable shortest path for rescue and recovery missions. The idea is to enable military units to make a visual decision on routes based upon the time-distance optimality among all the possible paths. Specifically, we are developing the *Top-k* Objected-oriented Smoothest Paths (TOSP) model which captures the object dynamics of geospatial temporal network in a terrain over a time horizon. These objects include stationary entities (e.g., buildings, roads, trees, etc.), mobile objects (e.g., vehicles, people, etc.), and route segments (e.g., steep slopes, mud roads, etc.). We are also extending the SPA to be a dynamic learning algorithm, i.e., the Time-varying Smoothest Path (TSP) algorithm, which integrates the object dynamics to learn the *top-k* smoothest paths at each instance of time. The main advantage offered by the SPA extension is its lower logarithmic time complexity, i.e., $O(NlogN)$, where N is the number of nodes in a terrain. Finally, we develop a new design of visual displays that enable military operators to analyze other crucial factors, such as vehicle types, weather severity, and soldiers' specialties, which are required to be interpreted by human perception, cognition, and knowledge to select the best path among the *top-k* smoothest routes at each instance of time for rescue and recovery missions.

The rest of the paper is organized as follows. In Section II, we use the above military operation, i.e., determining the viable shortest path for the rescue and recovery mission, to provide an overview on our VEGA Tool. In Section III, we describe the S-GOOSE data model and GQL, as well as illustrate the model and language based upon the military case. In Section IV, we present the implementation architecture for the viable shortest-path learning process. In Section V, we explain the initial design of our high-level 3D visual display for determining the viable shortest path among the *top-k* smoothest routes. In Section VI, we conclude our paper and briefly outline our future work.

II. Visual Decision-Guided Analytics Tool

Fig. 1 shows the VEGA tool and its components. The VEGA tool consists of seven main components: Geo-Data Source Collector (GDSC), Geo-Data Fusion Integrator (GDFI), Geo-Data Warehouse (GDW), Geo-Query Language (GQL), Data Visualizer (DV), TOSP Compiler, and TSP Solver. The GDSC allows military operators to directly interact with heterogeneous data sources, e.g., satellite imagery, vector/raster maps, temporal data, 3D objects, etc., and gather those geo-data objects from multiple units, including Army, Navy, Marine, and Air Force.

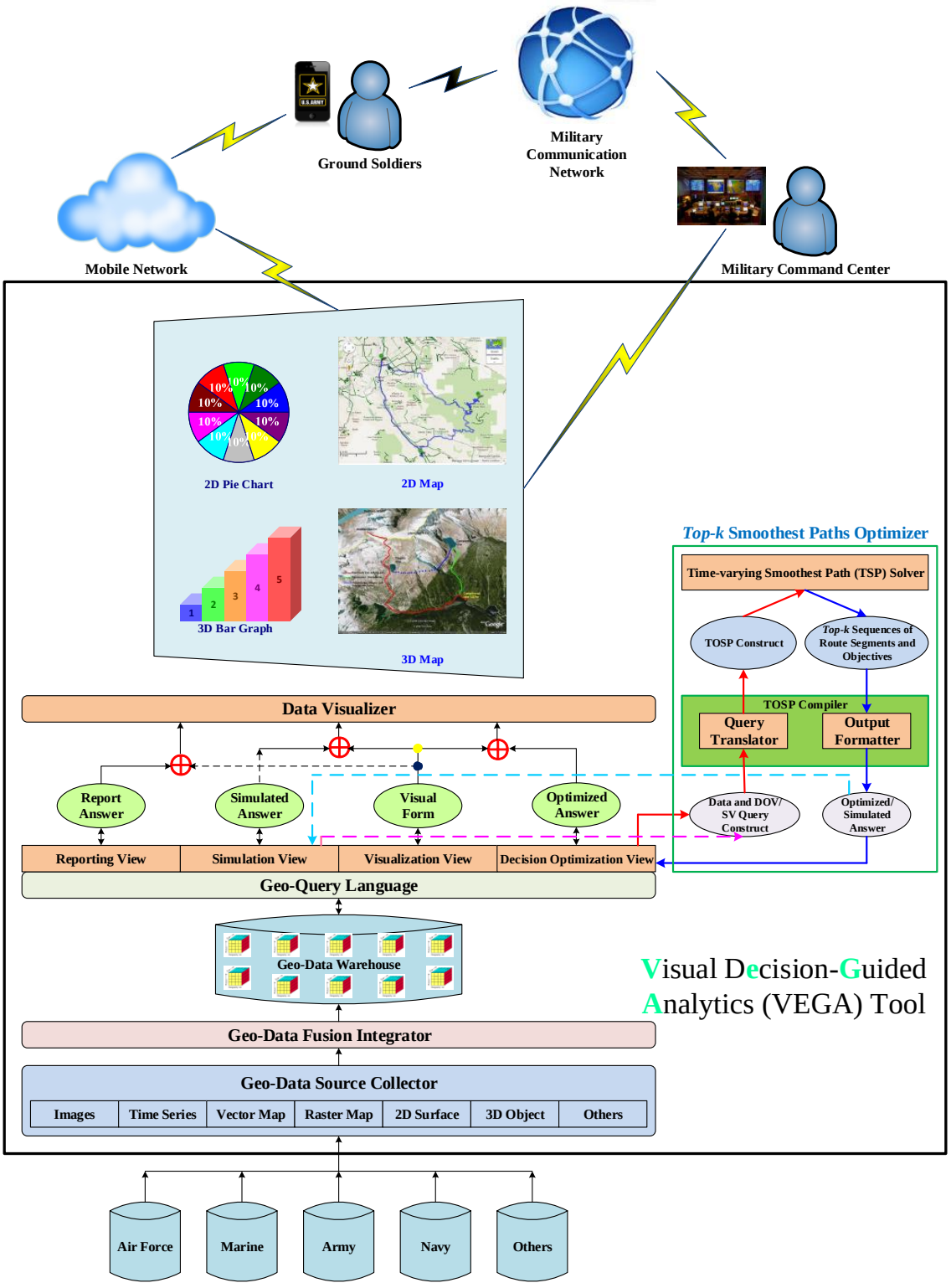


Fig. 1. Visual Decision-Guided Analytics Tool

After geo-data objects are collected, military forces can operate the GDFI to clean the objects and then integrate them into a S-GOOSE data cube, which provides and maintains a concentric and coherent view of the collected data. These cubes are then archived in the GDW, that is, the extended S-GOOSE data warehouse. The unified cubes can also be decomposed into different terrain layers according to their themes, e.g., trees, buildings, mountains, roads, etc., which are also stored in the GDW as well for future analysis.

The GQL enables military operators to (1) develop and implement the extended S-GOOSE data cubes and (2) construct different views, including reporting, simulation, visualization, and decision optimization, based upon the integrated geo-data objects to support determining the viable shortest path for rescue and recovery missions. More specifically, the GQL enables military operators to (1) retrieve data from the GDW and (2) construct reporting (RV), simulation (SV), visualization (VV), and decision optimization (DOV) views based on the unified geo-data objects. Once military operators initiate the data and DOV query construct to the TOSP compiler, the query translator transforms the DOV query into the TOSP format, which is then sent to the TSP solver with the TSP algorithm to learn the *top-k* smoothest paths at each instance of time. After the compiler receives the *top-k* route segments and objectives, the output formatter renders the results as the optimized answers to the DOV query. In order to provide the future insight, military operators can also construct the SV query to simulate the *top-k* smoothest paths at the next instance of time. The answers from both DOV and SV query, as well as the visual form from the VV query are then integrated by the $\oplus$ aggregator, which delivers the aggregated results to the DV. If it is needed, military operators can also formulate the RV and VV query for constructing a report, e.g., the total number of buildings in each city per state at a specific time after a natural disaster.

The DV displays analytical diagrams and figures, e.g., 2D pie charts and maps, 3D bar graphs and maps, etc., to military operators. Military operators, e.g., the ground soldiers, can use the pull or push services provided by the mobile network and devices to receive the latest visual information about the routes. The military command center can also base on the visual information to guide the soldiers' next movements via the communication network. Collaboratively, using our designed VEGA visuals, both parties are able to make a final decision on the viable shortest path based on the optimal and simulated results of the *top-k* smoothest paths, as well as the other crucial factors, e.g., vehicle types, weather severity, and soldiers' specialties.

III. S-GOOSE Data Model and Geo-Query Language By Example

In this section, we discuss in detail the S-GOOSE data model and GQL, which are used for reporting, simulation, visualization, and decision optimization. Again, we use the military operation, i.e., determining the viable shortest path for rescue and recovery missions, to illustrate the ideas.

A. S-GOOSE Data Model

The S-GOOSE data model is an extension of the object-relational database model with a specialized schema, i.e., a S-GOOSE OLAP cube with a number of dimension schemas. The dimension schemas include Time, Location, Visualization, and Composite Object in a terrain over a time horizon. To demonstrate the concepts, we assume that there is a building layer and a road layer in a disaster region and the military operational unit passes through a sequence of buildings along some road segments in an area to rescue a group of victims, who are trapped inside the target building. The extended S-GOOSE OLAP cubes of the building and the road layers are shown in Fig. 2 and 3 respectively.

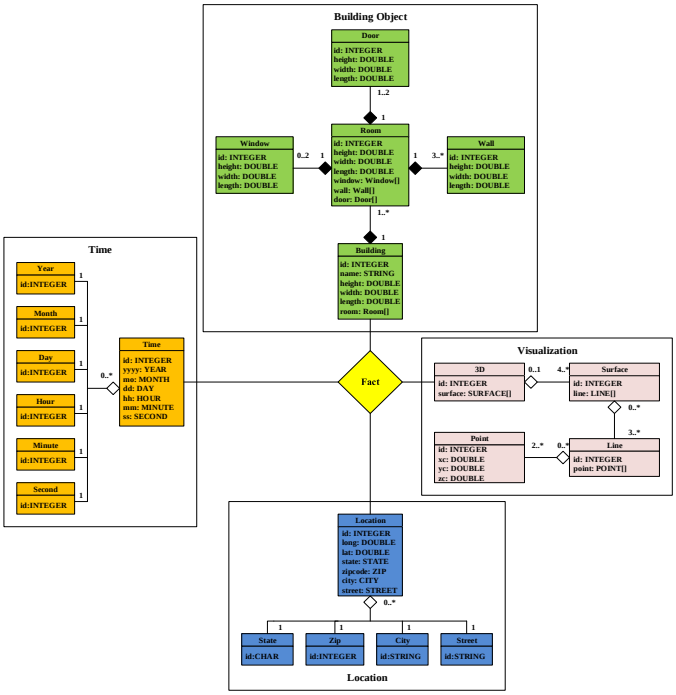


Fig. 2. S-GOOSE OLAP Cube for the Building Layer.

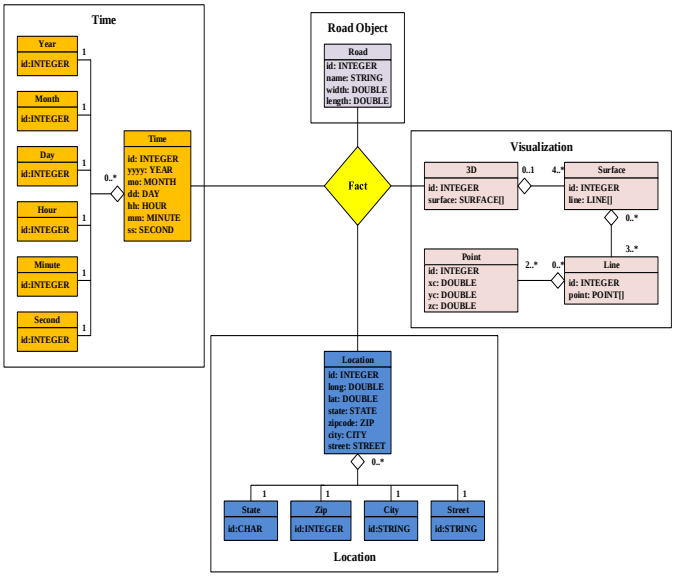


Fig. 3. S-GOOSE OLAP Cube for the Road Layer.

Time Dimension

A time dimension is of the form Time(id:INTEGER, yyyy:YEAR, mo:MONTH, dd:DAY, hh:HOUR, mm:MINUTE, ss:SECOND), where YEAR, MONTH, DAY, HOUR, MINUTE, and SECOND are the data types of yyyy, mo, dd, hh, mm, and ss respectively. Each data type is an object-relational table that has an attribute id to define a property of that type. For example, the attribute id '12' in the MONTH table represents December. The attribute id '15' in the MINUTE table represents the first quarter of an hour, i.e., 15 min.

A tuple over a Time schema is an object-relational tuple over that schema, i.e., a mapping m: {id, yyyy, mo, dd, hh, mm, ss} → INTEGER x YEAR x MONTH x DAY x HOUR x MINUTE x SECOND, such that m(id) ∈ INTEGER, m(yyyy) ∈ YEAR, m(mo) ∈ MONTH, m(dd) ∈ DAY, m(hh) ∈ HOUR, m(mm) ∈ MINUTE, and m(ss) ∈ SECOND.

Location Dimension

A location dimension is of the form Location(id:INTEGER, long:DOUBLE, lat:DOUBLE, state:STATE, zipcode:ZIP, city:CITY, street:STREET), where STATE, ZIP, CITY, and STREET are data types of state, zipcode, city, and street respectively. Each above data type is an object-relational table that has an attribute id to define a property of that type. For example, the attribute id '22032' in the ZIP table represents a zipcode of the city. The attribute id 'VA' in the STATE table represents a state, i.e., Virginia. Note that long and lat are the longitude and the latitude of an object on the map.

A tuple over a Location schema is an object-relational tuple over that schema, i.e., a mapping m: {id, long, lat, state, zipcode, city, street} → INTEGER x DOUBLE x DOUBLE x STATE x ZIP x CITY x STREET, such that m(id) ∈ INTEGER, m(long) ∈ DOUBLE, m(lat) ∈ DOUBLE, m(state) ∈ STATE, m(zipcode) ∈ ZIP, m(city) ∈ CITY, and m(street) ∈ STREET.

Visualization Dimension

A visualization dimension has four layers of schemas: 3D(id:INTEGER, surface:SURFACE[]), Surface(id:INTEGER, line:LINE[]), Line(id:INTEGER, point:POINT[]), and Point(id:INTEGER, xc:DOUBLE, yc:DOUBLE, zc:DOUBLE), where SURFACE[], LINE[], and POINT[] are the array types of surface, line, and point attributes respectively. Each above data type is an object-relational array that stores a set of its component objects to construct a composite object. For example, the array attribute 'surface' in the 3D table stores all the ids of the surfaces defined in the Surface table to construct a 3D object. The array attribute 'line' in the Surface table stores all the ids of the lines defined in the Line table to construct a 2D object. The (xc, yc, zc) in the Point schema is the actual coordinate (latitude, longitude, elevation) on a contour line of an object, e.g., a tree, a house, a road, a building, etc.

A tuple over a 3D schema is an object-relational tuple over that schema, i.e., a mapping m: {id, surface} → INTEGER x SURFACE[], such that m(id) ∈ INTEGER and m(surface) ∈ SURFACE[].

A tuple over a Surface schema is an object-relational tuple over that schema, i.e., a mapping m: {id, line} → INTEGER x LINE[], such that m(id) ∈ INTEGER and m(line) ∈ LINE[].

A tuple over a Line schema is an object-relational tuple over that schema, i.e., a mapping m: {id, point} → INTEGER x POINT[], such that m(id) ∈ INTEGER and m(point) ∈ POINT[].

A tuple over a Point schema is a relational tuple over that schema, i.e., a mapping m: {id, xc, yc, zc} → INTEGER x DOUBLE x DOUBLE x DOUBLE, such that m(id) ∈ INTEGER, m(xc) ∈ DOUBLE, m(yc) ∈ DOUBLE, and m(zc) ∈ DOUBLE.

Composite Object Dimension

A composite object dimension is of the form CompObj(id:INTEGER, [name:STRING], [height:DOUBLE], [width:DOUBLE], [length:DOUBLE], [compObjName:COMPOBJTYPE]). A CompObj is an object that can be constructed by a set of its component objects. The optional attributes, 'height', 'width', and 'length', are the dimension of an object with the optional 'name'. [compObjName:COMPOBJTYPE] is a set of optional object-relational arrays that stores a set of their component objects to construct a composite object. Each component object can be constructed from another set of component objects.

A tuple over a CompObj schema is an object-relational tuple over that schema, i.e., a mapping m: {id, [name], [height], [width], [length], [compObjName]} → INTEGER x [STRING] x [DOUBLE] x [DOUBLE] x [DOUBLE] x [COMPOBJTYPE], such that m(id) ∈ INTEGER, m(name) ∈ STRING, m(height) ∈ DOUBLE, m(width) ∈ DOUBLE, m(length) ∈ DOUBLE, and m(compObjName) ∈ COMPOBJTYPE.

S-GOOSE Database Schema

The S-GOOSE database schema is a set of object-relational schemas, which include a number of S-GOOSE OLAP cubes.

Using the extended S-GOOSE OLAP cubes of the building layer and the road layer shown in Fig. 2 and 3 respectively, the military forces can (1) create the S-GOOSE tables and views, including reporting, simulation, visualization, and decision optimization, (2) store the tables and views with the data in the GDW, and (3) execute the views to perform the analysis in different combinations of dimensions.

B. Reporting View

Using the Time, Location, and Composite Object dimensions, the military force can generate a report to display

the total number of buildings in each city, e.g., Fairfax, in a state, e.g., VA, at a specific time, e.g., March 15, 2014, after the natural disaster. This reporting view, shown in Box 1, can be constructed by using the conventional PostGIS query.

Box 1

```
CREATE VIEW BuildingReport AS (
  SELECT L.state, L.city,
       COUNT(DISTINCT B.id) AS "# of Cities"
  FROM Time T, Location L, Building B, Fact F
  WHERE T.id = F.tid AND L.id = F.lid AND
    B.id = F.bid AND T.yyyy[T.id] = '2014'
    AND T.mo[T.id] = '3' AND
    T.dd[T.id] = '15'
  GROUP BY L.state, L.city
  ORDER BY L.state, L.city
)
```

C. Decision Optimization View

The decision optimization query helps the military troop compute and learn the *top-k* smoothest paths to reach the target building at each instance of time for the rescue and recovery mission. The procedures for constructing the view are shown in the following steps:

STEP 1: Create a `BuildingCoordinate` view shown in Box 2 to retrieve all the buildings' coordinates in terms of their latitudes (lat) and longitudes (long) within the building layer in a particular city, e.g., Fairfax, at a specific time, e.g., March 15, 2014.

Box 2

```
CREATE VIEW BuildingCoordinate AS (
  SELECT B.id, L.long, L.lat
  FROM Time T, Location L, Building B, Fact F
  WHERE T.id = F.tid AND L.id = F.lid AND
    B.id = F.bid AND L.city = 'Fairfax' AND
    T.yyyy[T.id] = '2014' AND T.mo[T.id] =
    '3' AND T.dd[T.id] = '15'
)
```

STEP 2: Create a `RoadCoordinate` view shown in Box 3 to retrieve all the road segments' coordinates in terms of their lat and long within the road layer in the same city, i.e., Fairfax, on March 15, 2014.

Box 3

```
CREATE VIEW RoadCoordinate AS (
  SELECT R.id, L.long, L.lat
  FROM Time T, Location L, Road R, Fact F
  WHERE T.id = F.tid AND L.id = F.lid AND
    R.id = F.rid AND L.city = 'Fairfax' AND
    T.yyyy[T.id] = '2014' AND T.mo[T.id] =
    '3' AND T.dd[T.id] = '15'
)
```

STEP 3: Create a `BuildingRoadMap` view shown in Box 4 to retrieve all the buildings' coordinates within a distance of, e.g., 0.1 mile (5280 feet x 0.1), of their road segment's coordinate in that area. Note that `ST_DWithin()` [19] and `ST_MakePoint()` [20] are the spatial relationship functions of the PostGIS query [15].

Box 4

```
CREATE VIEW BuildingRoadMap AS (
  SELECT RC.long AS rlong, RC.lat AS rlat,
    BC.long AS blong, BC.lat AS blat
  FROM BuildingCoordinate BC,
       RoadCoordinate RC
  WHERE
    ST_DWithin(ST_MakePoint(BC.long, BC.lat),
    ST_MakePoint(RC.long, RC.lat), 5280 *
    0.1)
)
```

STEP 4: Create a decision optimization view shown in Box 5 and then execute the view `DetermineTopKSmoothestPaths` to determine the *top-k* sequences of all the buildings (`BRM.blong, BRM.blat`) within a distance of 0.1 mile of their road segments (`BRM.rlong, BRM.rlat`) such that the total distance in each sequence between the military force's present location and the target building location is minimal in order.

Box 5

```
CREATE VIEW DetermineTopKSmoothestPaths AS (
  LEARN SEQ(BRM.rlong, BRM.rlat, BRM.blong,
            BRM.blat)
  FOR MINIMIZE
  ST_TopK_Smoothest_Paths(BRM.rlong,
    BRM.rlat,      BRM.blong,      BRM.blat,
    presentLong,   presentLat,   targetLong,
    targetLat)
  FROM BuildingRoadMap BRM
)

EXECUTE DetermineTopKSmoothestPaths;
```

Once the above query construct is ***EXECUTED*** and initiated to the GDW, the VEGA tool invokes the *Top-k* Smoothest Paths Optimizer to ***LEARN*** the *top-k* sequences ***SEQ*** of the buildings' coordinates (`BRM.blong, BRM.blat`) along their paths (`BRM.rlong, BRM.rlat`) on the `BRM`. Note that ***ST_TopK_Smoothest_Pahts*** is a new function being developed, which accepts the four sets of input parameters, including the roads' coordinates (`BRM.rlong, BRM.rlat`), their buildings' coordinates (`BRM.blong, BRM.blat`), the troop's present location (`presentLong, presentLat`), and the target building location (`targetLong, targetLat`), to compute and learn the *top-k* smoothest paths based upon the `BRM`. The ***SEQ*** is another new function to return the *top-k* sequences of all the buildings' coordinates (`BRM.blong, BRM.blat`) along their roads' coordinates (`BRM.rlong, BRM.rlat`) delivered by the ***ST_TopK_Smoothest_Pahts*** function.

D. Simulation View

Similarly, following the above **STEP 1 ~ 4**, we can also construct the simulation view to predict the *top-k* smoothest paths at the next instance of time. Some existing methodologies, e.g., random-walk, random-trend, autoregressive, exponential smoothing, etc., can be used to generate those forecasting data based on the real-time dataset to learn the *top-k* smoothest paths.

E. Visualization View

To support military troops to make a better decision on the viable shortest path for their rescue and recovery missions, various data visualization approaches are proposed and developed to assist military units in performing visual analytics over object-oriented spatial-temporal data. Those techniques present and deliver visual objects that require human interpretation and supervision on data. To support human interpretation and supervision on those visual objects, military units can construct the visualization view, e.g., the distribution view of all the building objects along their paths in a terrain at an instance of time. Box 6 shows an example of a visual query that the military troops can construct to display the building layer in the terrain after a natural disaster.

Box 6

```
CREATE VIEW 3DBuildingShape AS (
  SELECT (SELECT P.xc, P.yc, P.zc
          FROM Point P
          WHERE P.id COMPOSE
            (SELECT 1D.point[1D.id]
             FROM Line 1D
             WHERE 1D.id COMPOSE
               (SELECT 2D.line[2D.id]
                FROM Surface 2D
                WHERE 2D.id COMPOSE
                  3D.surface[3D.id])
  FROM Time T, Location L, Building B, 3D,
    Fact F
  WHERE T.id = F.tid AND L.id = F.lid AND
    B.id = F.bid AND 3D.id = F.did AND L.city
    = 'Fairfax' AND T.yyyy[T.id] = '2014' AND
    T.mo[T.id] = '3' AND T.dd[T.id] = '15'
)

VISUALIZE 3DBuildingShape;
```

Please note that ***COMPOSE*** is a new keyword to evaluate whether an object constructs another object. For instance, the syntax *P.id* ***COMPOSE*** 1D.point[1D.id] means that an actual point coordinate, which is one of the components, constructs a contour line of an object, that is, a building. Likewise, the syntax *2D.id* ***COMPOSE*** 3D.surface[3D.id] means that a 2D object, which is one of the components, constructs a 3D object. ***VISUALIZE*** is another new keyword to execute the *3DBuildingShape* view to display a 3D building layer in a terrain at an instance of time.

IV. Implementation of a High-Level Architecture for the Viable Shortest-Path Learning Process

Fig. 4 illustrates the viable shortest-path learning process. As this figure shows, the military operators can use the GQL interface to construct the DOV and SV query for the learning event, e.g., *DetermineTopKSmoothestPaths*. Once this learning event aggregated with the S-GOOSE OLAP cube is initiated to the GDW, the TSP optimizer invokes the TOSP compiler, which calls the query translator to transform the learning event into the TOSP construct. This TOSP construct is then sent to the TSP solver to learn the *top-k* sequences of the buildings' coordinates along their route segments and objectives. These *top-k* sequences and objectives are then processed by the output formatter associated with the query translator to return the optimized and/or simulated answers back to the GQL interface, which presents the results to the operators. Finally, the operators can formulate the visualization view aggregated with the optimized and/or simulated answers to be displayed on the screen by the Data Visualizer.

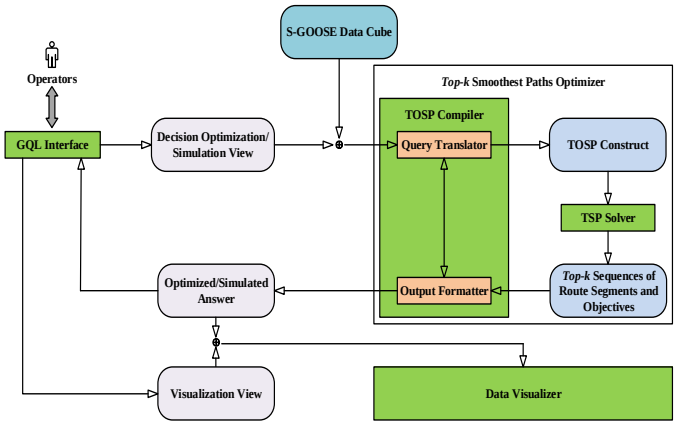


Fig. 4. The Viable Shortest-Path Learning Architecture.

V. Initial Design of 3D Visual Display

The initial design of our high-level 3D visual display for the optimized and the simulated *top-k* smoothest paths is shown on the mobile devices as an example (see Fig. 5 and 6 respectively). These visual displays are updated and refreshed for every instance of time. Fig. 5 screen display is divided into two portions. The right-hand portion shows the *top-k* optimal paths on the map, where the red line is the first optimal path, and the blue line is the second at the current time point *t*. Both of the paths are learned by our TSP algorithm. The left-hand portion displays the road characteristics, such as grades (e.g., HILL), conditions (e.g., BUMP), and speeds (e.g., 35 MPH), using the standard road signs, as well as shows the current weather (e.g., Sunny), the soldier's specialties (e.g., Corporal), and their vehicle types (e.g., a Four-wheel Truck). Fig. 6, which has the same visual layout as Fig. 5, shows the *top-k* simulated paths on the map, where the green line is the first optimal, simulated path, and the pink line is the second at the future time point $t + \Delta t$. Due to the dynamics of geospatial temporal network in a terrain over a time horizon, the simulated results render different *top-k* paths learned by our TSP algorithm. Using the both optimal and simulated paths

with other crucial factors shown on the visuals, the military command center and the soldiers can determine the best path among the fours for their rescue and recovery missions.

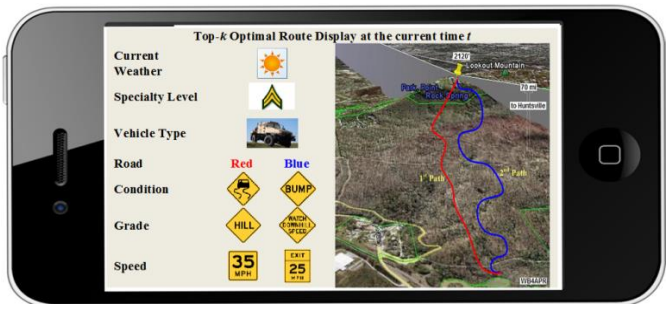


Fig. 5. The *Top-k* Optimal Paths Display.

Fig. 6. The *Top-k* Simulated Paths Display.

VI. Conclusions and Future Work

In this paper, we propose a VEGA tool that is a unified decision-making application that combines both advantages of quantitative analysis (i.e., optimization programming in operation research) and qualitative methodologies (i.e., geo-data visualization in data analytics) to determine the viable shortest path for rescue and recovery missions. This application supports reporting, simulation, visualization, and decision optimization over geospatiotemporal data. Technically, this tool enables military units to (1) collect geo-data objects from multiple data sources, (2) conflate the diverse objects into unified data sets, and (3) perform analysis on these sets. First, to support a better decision-making, we extend the object-oriented spatial-temporal data model as a multidimensional OLAP (star-schema) cube, i.e., a Star-based Geo-Object-Oriented SpatiotEmporal (S-GOOSE) data model, which combines the advantages of both OLTP and OLAP approaches. This S-GOOSE data model is an object-relational-based cube that enables military operators to analyze unified geo-data objects from multiple dimensions, such as time, space, and location, to help them make a better decision on routes. Second, we enhance the capability of the PostGIS query, named Geo-Query Language (GQL), to support the S-GOOSE data cube analysis, which enables military operators to simulate the occurrence of events, to visualize and report geo-data objects, as well as to solve decision optimization problems based upon their events of interest. Third, we integrate optimization programming into geo-data visualization to determine the viable shortest path for rescue and recovery missions. The idea is to enable military units to make a visual decision on routes based upon the time-distance optimality among all the possible paths. Finally, we develop a new design of visual displays that enable military operators to analyze other crucial factors, such as vehicle types, weather severity, and soldiers' specialties, which are required to be interpreted by human perception, cognition, and knowledge to select the best path among the *top-k* smoothest routes at each instance of time for rescue and recovery missions. There are still many open research questions, particularly conflating multiple geo-data objects, e.g., satellite imagery, vector/raster maps, temporal data, 3D objects, etc., into an integrated data unit.

References

[1] US Forest Service. (2000). http://www.fs.fed.us/fire/doctrine/genesis_and_evolution/source_materials/joint_vision_2020.pdf

[2] Li, B. & Cai, G. (2002). *A General Object-Oriented Spatial Temporal Data Model*. Geospatial Theory, Processing and Applications. ISPRS Commission IV, Symposium 2002. Volume XXXIV Part 4.

[3] Vaisman, A. & Zimanyi, E. (2009). *A Multidimensional Model Representing Continuous Fields in Spatial Data Warehouses*. Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems. New York, NY, USA.

[4] Maimon, O.Z. & Rokach, L. (2005). Data Mining and Knowledge Discovery Handbook. Springer New York, Inc.

[5] Bertini, E. & Lalanne, D. (2009). Investigating and Reflecting on the Integration of Automatic Data Analysis and Visualization in Knowledge Discovery. ACM SIGKDD Explorations Newsletter. NY, USA.

[6] Agrawal, R. & Srikant, R. (1994). Fast Algorithms for Mining Association Rules in Large Databases. Proceedings of the 20th International Conference on Very Large Data Bases. CA, USA.

[7] Jolliffe, I.T. (2002). Principal Component Analysis. Springer-Verlag.

[8] Mohri, M, Rostamizadeh, A, & Talwalkar, A. (2012). Foundations of Machine Learning. The MIT Press.

[9] Duda, R, Hart, P, & Stork, D. (2001). Unsupervised Learning and Clustering. John Wiley & Sons, Inc.

[10] Keim, D.A., Kohlhammer, J., Ellis, G., & Mansmann, F. (2010). Mastering the Information Age - Solving Problems with Visual Analytics. http://www.vismaster.eu/wp-content/uploads/2010/11/VisMaster-book-lowres.pdf.

[11] Winston, W.L. (2003). Operations Research: Applications and Algorithms (The 4th Edition). Cengage Learning.

[12] Pardalos, P.M. & Hansen, P. (2008). Data Mining and Mathematical Programming. American Mathematical Society.

[13] Skiena, S.S. (2008). The Algorithm Design Manual (The 2nd Edition). Springer-Verlag London Limited.

[14] Roles, J.A. & ElAarag, H. (2013). A Smoothest Path Algorithm and its Visualization Tool. Proceedings of the IEEE SoutheastCon Conference. Florida, U.S.A.

[15] George, B. & Kim, S. (2013). Shortest Path Algorithms for a Fixed Start Time. SpringerBriefs in Computer Science. Springer New York.

[16] Burstein, F., Silva, D., Jelinek, H., & Stranieri, A. (2013). *Multivariate Data-Driven Decision Guidance for Clinical Scientists*. Proceedings of the IEEE 29th International Conference on Data Engineering Workshops. Brisbane, Australia.

[17] PostGIS. (2013). http://postgis.net/.

[18] Open Geospatial Consortium. (2014). http://www.opengeospatial.org/.

[19] PostGIS. (2013). http://postgis.org/docs/ST_DWithin.html.

[20] PostGIS. (2013). http://postgis.net/docs/manual-1.4/ST_MakePoint.html.

Using Influence for Navigation in Online Social Networks

Bastien Lebayle[1], Mehran Asadi[2] and Afrand Agah[1]
(Corresponding author: Afrand Agah)
(Email: aagah@wcupa.edu)
Department of Computer Science, West Chester University[1]
West Chester, PA 19383
Department of Business and Entrepreneurial Studies, The Lincoln University[2]
Lincoln University, PA 19352

Abstract

If influence is the capacity to have an effect on someone, then who are the influential people in an Online Social Network?

In this paper, we investigate the role of influential people in an Online Social Network. Then we present an algorithm that take advantage of influential people to reach a target in the network. Our navigation algorithm returns a path between two nodes in an average of 10% less iterations, with a maximum of 83% less iterations, and only relies on public attributes of a node in the network.

1 Introduction

Over the course of human history, the collections of social ties among friends have grown steadily in complexity. When people live in neighborhoods or attend schools, the social environment already favors opportunities to form friendships with others like oneself.

In the most basic sense, a network is any collection of objects in which some pairs of these objects are connected by links. In a network of objects, objects can be people or computers, which we refer to them as nodes of the network. Have the people in the network adapted their behaviors to become more like their friends, or have they sought out people who were already like them [4]?

Here we are interested in connectedness at the level of behavior - the fact that each individual's actions have implicit consequences on the outcomes of everyone in the system [4]. We investigate prediction of peoples behavior and influences in online social networks. Our focus is on how different nodes can play distinct roles in information flow through an online social network.

This paper is organized as follows. Section 2 reports the related work. In Section 3 we define three different types of crawlers for navigation in Online Social Networks. Section 4 evaluates the performance of the proposed protocol, and Section 5 concludes the paper.

2 Related Work

Targeted crawler algorithms [5] allow a crawler to find a path from a source node to a target node assuming: (i) the crawler controls at least one node in the Online Social Networks (OSN), which is attainable by simply registering on most OSN, and (ii) the crawler knows when its target is reached, either by comparing attributes or unique ids. The targeted crawler algorithms can be seen as a frontier [5] that is expanding toward the target node using a distance function. The distance function represents how different two nodes are: the more they have in common, the less distant they are.

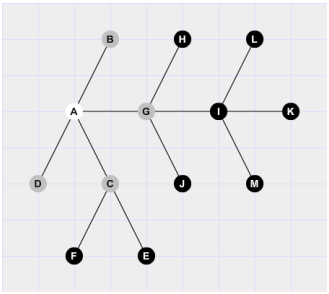


Figure 1: TargetedCrawler at first iteration

For each step in the algorithm, the frontier spreads toward the node that has the least remaining distance to the target. The frontier expansion can be seen in Figure 1 and Figure 2. Assume we want to find a path from node A to node K. At the first step (Figure 1), the frontier consists of all A's friends. Then using the distance function, we find G is the closer to the target (Figure 2). Hence G goes to the explored set represented in white, and the frontier represented in grey is extended to G's friends.

Two procedures to find key players by (i) the identification of key players for the purpose of optimally diffusing a property through the network by using the key players as seeds and (ii) the identification of key players for the purpose of disrupting or fragmenting the network by removing the key nodes, are discussed in [2]. Our work will try to find a procedure to find key players locally for a path between two nodes while [2] focuses on key players for the network as a whole.

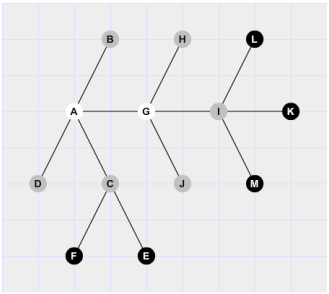


Figure 2: TargetedCrawler at second iteration

The definition of influential nodes changes with application and the type of commodity flowing through a network. While our work focuses on navigation paths, identifying influential nodes in OSN using principal component centrality are developed in [6].

3 Prediction in Online Social Network

Our focus is on how different nodes can play distinct roles in information flow through an OSN. If two people in an OSN have a friend in common, then there is an increased likelihood that they will become friends themselves at some point in the future [4]. The terminology of OSN reflects a largely similar views, through its emphasis on the connections one forms with friends, fans, followers, and so forth.

An Online Social Network can be viewed as a graph G = (V, E). Vertices (nodes) are representing people and edges are representing social links. Social links can be undirected (e.g. friends on Facebook) or directed (e.g. followers on Twitter). An OSN allows people to have attributes that are included in the Social Network list of attributes. For each attribute, people can define a privacy policy: visible to all, visible to friends, or private.

Our work is based on the following assumption: from a graph G with privacy policies on attributes and links, we can deduce a public sub-graph G' based on public attributes and public links [1] and [7] . Because of the default behavior of OSN is to share the most and hide the less, we can deduce G', thanks to users that do not change this default behavior. Each user that is changing the default settings contributes to shrink our graph G'.

We need a targeted crawler algorithm for our influence prediction in order to be able to navigate efficiently in an OSN. This is for the following two main reasons: (i) Online Social Networks are huge (billions of users for Facebook as of today) and should not been seen as random graphs. (ii) If people are connected, there is a high chance they share something in common, either a friendship, a location, a job, etc. We should not use algorithm like breadth-first search in Online Social Networks because they do not take advantage of the probability that people are connected.

3.1 Monte Carlo method

The Monte Carlo method is used when a distribution of an unknown probabilistic entity is close to impossible to determine in a deterministic way. Instead, we compute a non-deterministic algorithm a certain number of times until enough numerical results are collected to generalize the distribution of the unknown probabilistic entity.

3.2 Predicting Influence

We want to be able to locate those people, who are more influential. People are considered influential if when they do an action, their friends, friends of their friends, etc, replicate their actions.

The goal here is to understand influence not just as a property of nodes in a network, but in social interactions as the roles people play in groups of friends in communities or in organizations. Influence is not so much a property of an individual as it is a property of a relation between two individuals. Influence may be almost entirely the result of the personalities of the two people involved. But it may also be a function of the larger social network in which the two people are embedded. One person may be more powerful in a relationship because he occupies a more dominant position in the social network with greater access to social opportunities outside this single relationship.

One way to define important people is as follow: the more paths are going through a node, the more important this node is. This is different from the degree of a node as it also takes into consideration nodes that are not directly linked to it. In figure 3, node A and E have the same degree: 3. However because E's friends are more connected, there are more paths that go through E than paths that go through A.

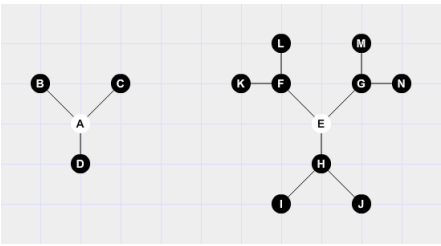


Figure 3: Degree vs. influence

If for 100 computed paths a particular node N_1 is part of 50 paths and another node N_2 is part of 10 paths, then we say N_1 is more important than N_2 because it connects more people together. The influence of a node N in the network is defined in equation 1:

$$Influence(N_i) = \#ofPathsGoingThrough(N_i) \quad (1)$$

Computing (1) is highly time consuming. Once we have the influence associated to each node, a navigation path can be defined by hoping from influent node to influent node until the target is reached.

4 Performance Evaluation

The dataset used for following experimentation comes from Stanford university [3]. Data was collected from users who had manually shared their circles in the Google+ Social Network. The original dataset consisted in 107 thousands nodes and 13.7 millions edges. In order to reduce running time of the experimentation, the dataset has been reduced to 700 nodes and 35 thousands edges. Moreover, bidirectional links between 2 nodes have been made unidirectional. This allows us to enlighten the different behaviors of our crawlers with a smaller data set. Each node of our sample networks is part of at least one link, either the head or the tail of the link. Each node has between 0 and 452 friends. Nodes that have at least one friend have in average 66 friends.

4.1 Simple Crawler

Assume the online network has N nodes. The first way to compute our influence is to compute all possible paths as defined in equation (1). The TargetedCrawler algorithm, which corresponds to the navigation between 2 nodes, is executed $O(N^2)$ times. If no path is found between source and target node, the influence table is not updated. If a path is found between source and target node, for each node N_i being part of the path, $Influence(N_i)$ is incremented by one.

Assuming the TargetedCrawler algorithm runs in $O(1)$ time, computing the full set of paths will be done in $O(N^2)$ time. For a 1 ms computation time

per iteration and 10^6 nodes (which is not unrealistic for OSN, and even far from the truth for some), the computation would take approximately 10^{12} ms or 31 years. We can easily realize how this is infeasible.

4.2 Monte Carlo Crawler

The first approach showed that it is possible to define and influence value for each node in the network. The method described in 4.1 can return very precise results. However the method is also highly time consuming and infeasible for large OSN. Can we reduced the computation time and still have correct results? second approach aims to use the Monte

Node Id	Simple Crawler	Monte Carlo Crawler
206	16.3 (1)	17.2 (1)
422	9.8 (2)	7.87 (4)
170	9.6 (3)	8.91 (2)
938	8.4 (4)	4.48 (5)
675	7.1 (5)	8.91 (2)

Table 1: Monte Carlo Crawler vs. Simple Crawler

Carlo method to approximate the influence of a node. Instead of running the Targeted Algorithm on all possible N^2 paths, we are going to restrict the execution time of the influence calculation to $O(N)$. To do that, our second approach selects two randoms nodes and then runs the Targeted Algorithm N times. With this procedure, the time spent to compute our influence defined in (1) is much shorter. For $N = 10^6$, this approach would end in less than 10^6 ms or 20 minutes which is very feasible. Table 1 is extracted by running the two approaches on our sample network [3]. For the top 5 nodes, it shows the influence defined in equation1 normalized by the number of path computed and the overall ranking of the nodes are in parentheses. We can see that after only N iterations of the Monte Carlo Crawler and instead of the N^2 of our Simple Crawler, the influence of a node is already close to what it will be after N^2 iterations.

4.3 Our approach

Simple Crawler and Monte Carlo Crawler are based on the Targeted Crawler algorithm [5]. The issue with this algorithm is that sometimes the frontier spread in the wrong direction and this can lead to unnecessary time consumption to find the path from a source to a target. Simple Crawler showed that it is possible to compute an influence value for a node. Monte Carlo Crawler showed that it is possible to compute this value in a reasonable amount of time. From the two previous approaches, we can conclude the goal for our third approach should be finding a navigation algorithm that (i) should not go back or expand in the wrong direction, (ii) should be computable in a reasonable amount of time and (iii) should hop using influential nodes of the network. Authors in [9] showed that mobility measures alone yield surprising predictive power, comparable to traditional network and similarity between two individuals' movements strongly correlates with their proximity in the social network. We use some quantities which have been proven to perform reasonably well in previous studies [9].

Computing the Influence Value (IV) of a node N_i in a path P_i from source node S to target node T is defined as a function with multiple parameters:

$$IV(N_i) = \lambda_1 a + \lambda_2 b + \lambda_3 c + \lambda_4 d + \lambda_5 e + \lambda_6 f \quad (2)$$

where each λ_i is a predefined weight parameter and (a) The number of direct friends (DF) of N_i: with V the set of vertices and E the set of edges of the network, direct friends of a node N_i are the friends that are reachable with a path of length one. They are defined by equation (3):

$$DF(N_i) = \{f | f \in V \text{ and } (N_i, f) \in E\} \quad (3)$$

(b) Shared neighbors (SN): If N_i and T share a direct friend, then there is a path of length 2 going from N_i to T. Common neighbors are defined in equation (4):

$$SN(N_i, T) = \{f | f \in DF(N_i) \text{ and } f \in DF(T)\} \quad (4)$$

(c) The number of attributes in common.

(d) The number of unique attributes.

(e) Distance to T: The distance between N_i and T is computed using attributes of the nodes. The more attribute they share, the smaller their distance is. See equation 5. Here J is the set of available attributes in OSN and $A_j(N_i)$ is the attribute j of

node N_i. The distance function can be simplified as in equation (5):

$$dist(N_i, T) = |J| - \sum_{j \in J} bool(A_j(N_i) == A_j(T_i)) \quad (5)$$

(f) The already crawled path: this is a list of nodes that have already been visited by the algorithm.

Our Influence Crawler algorithm is defined in Algorithm 1. Each time from the source node, we choose the highest influential friend and then consider this friend as the next hop until the target is reached.

Algorithm 1 Influence Crawler

```
Input: N_i, S, T
path = ∅
current = S
while current ∉ DF(T) do
    f* = friend of current with maximum influence
    if IV(f*) > 0 then
        add f* to path
        current = f*
    else                                  ▷ Dead end case
        Failure
    end if
end while
```

As our Influence Crawler is executed, multiple issues have to be resolved. In the early stage of routing, the distance between N_i and T decreases rapidly until we are in an virtual area where nodes are highly similar. People that share a lot are usually connected. Because the diameter of the Facebook graph is around 6 [8], we can consider that N_i has a high probability of having friends in common with T once our algorithm has run for 4 to 5 iterations. At this point, we can use the set of shared neighbors between N_i and T to help our algorithm to select the next hop more efficiently. The similarity between N_i and T defined in (c) is also a measure of relative proximity. In Online Social Networks, we are generally connected to people that look like us. Hence the more similar N_i and T are, the higher chance there is a short path between them.

Our influence values should be higher for nodes that are in the direction of the target. The already crawled path is passed as an argument of our influence function defined in equation (2). This allows our algorithm to select those nodes that are close to the source. The crawler can now return the next hop more efficiently by knowing what are the previous hops.

The correct direction of spreading can be defined as the direction that returns one of the shortest path between S and T, but not necessarily the shortest as it is impossible to know the shortest path without exploring the whole graph. The next hop should be chosen carefully because with our "can't go back" feature, we can't risk our algorithm to go in a dead-end direction. This can be avoided by making a compromise between two factors: (i) the next hop should be closer to the target and (ii) the next hop should be as connected as possible. If we favor (i) we are at risk to go straight and found ourselves in an impasse. If we favor (ii), we risk to found ourselves with an inefficient algorithm that is running in circle.

Algorithm 2 Influence Value

```
function IV(N_i)
    if N_i ∈ path then                          ▷ uses (f)
        influ = 0
    else if N_i ∈ DF(T) then                    ▷ uses (b)
        influ = K
    else if N_i ∈ DF(DF(T)) then                ▷ uses (b)
        influ = L
    else                                        ▷ uses (a)(c)(d)(e)
        influ = M − α*Dist(N_i, T) + β*|DF(N_i)|
    end if
    return influ
end function
```

with $K > L > M$

4.4 Simulation Results

The following table compares our Influence Crawler with the Targeted Crawler. Data have been extracted by running the two algorithms on the same random source and target nodes. We can see that our Influence Crawler founds the target node in a less number of iterations. However it is also possible that our Influence Crawler doesn't find a path when one exists. Our Influence Crawler stops in two cases:

(i) it reaches a dead-end in the graph because the algorithm made bad choices during its execution or (ii) we arbitrarily stop it because it takes too long comparing to the Targeted Crawler and our goal is to make an algorithm that is more efficient.

	Targ. Craw.	Influ. Craw.	Diff.
# iterations	17.746	2.92	-83.5%
Path length	2.495	2.436	-2.4%
Time (ms)	1.303	0.393	-69.8%
Success rate	80.4%	51.6 %	-28.8%

Table 2: Average results for 1000 iterations of the crawlers.

	Targ. Craw.	Influ. Craw.	Diff.
# iterations	19.327	2.754	-85.8%
Path length	3.103	2.754	-11.3%

Table 3: Success only

	Targ. Craw. + Infl. Craw.	Diff.
# iterations	14.243	-10.2%
Time (ms)	1.249	-4%
Success rate	80.4%	0%

Table 4: Average results for 1000 iterations of the Influence Crawler followed by the Targeted Crawler if failure.

Influence Crawler is constructed such as it adds a node to the general path at each iteration. Therefore the length of the returned path will always be equal to the number of iterations of the algorithm whereas in the Targeted Crawler, the length of the path is usually much smaller than the number of iterations.

Considering results in tables 2, 3 and 4we have a success rate of 50% with our Influence Crawler for 2.4 iterations in average. This means our Crawler fails 50% of the time. The number of iteration for a fail case is at most 8. In the case our Influence Crawler fails, we decided to run the Targeted Crawler to find a path instead. With this in mind, we can compute the average number of iteration needed to find a path between two nodes. This means the combination of the Targeted Crawler and our Influence Crawler runs in average 10% less iterations and has 50% chance of finding a path in 83% less iterations.

5 Conclusion and Future Work

We hope to develop a network perspective as a powerful way of looking at complex systems in general and a way of thinking about social dynamics, internal structure and feedback effects of the social networks.

References

[1] Lada A. Adamic and Eytan Adar, "Friends and neighbors on the web," *ScienceDirect*, 2003.

[2] Stephen Borgatti, "Identifying sets of key players in a social network," *Computational and Mathematical Organization Theory*, 2006.

[3] Stanford Large Network Dataset Collection.

[4] D. Easley and J. Kleinberg, *Networks Crowds and Markets*. Cambridge, 2010.

[5] Mathias Humbert, Theophile Studer, Matthias Grossglauser, and Jean Pierre Hubaux, "Nowhere to hide: Navigation around privacy in online in social networks," *The 18th European Symposium on Research in Computer Security (ESORICS)*, 2013.

[6] Muhammad U. Ilyas and Hayder Radha, "Identifying influential nodes in online social networks using principal component centrality," *IEEE International Conference*, 2011.

[7] Pedram Pedarsani, "Privacy and dynamics of social networks," *École Polytechnique Fédérale de Lausanne*, 2013.

[8] Johan Ugander, Brian Karrer, Lars Backstrom, and Cameron Marlow, "The anatomy of the facebook social graph," *Cornell University*, 2011.

[9] Dashun Wang, Dino Pedreschi, Chaoming Song, Fosca Giannotti, and Albert-Laszlo Barabasi, "Human mobility, social ties and link prediction," *KDD Conference*, 2011.

Sentiment Analysis for Mobile SNS Data

SeonHwan Kim, Il-Kyu Ha, Bong-Hyun Back and Byoungchul Ahn
Department of Computer Engineering, Yeungnam University, Gyeongsan, Gyeongbuk, Korea

Abstract – *Everyday a lot of diverse data have been generated every day regarding individual opinions and preferences on the contents of Social Network Service* (*SNS*). *These data could affect greatly to various fields of our society such as politics, public opinions, economics, services and entertainments. It is necessary to extract new information from SNS data and to understand the true intention of users or customers. To extract important information, it is required to several techniques to analyze a large amount of SNS data, extract meaningful data from them, and generate new information. This paper presents an efficient method that can process various unstructured big data on social networks, and extract the information for sentiment and generate preferences of users from sentiment information. The proposed method shows O*(*n*) *processing time as the number of data increases.*

Keywords: Big data, SNS, Sentiment

1 Introduction

Social Networking Service (SNS) is widely serviced by smart phones and their users are increased very rapidly in recent years. In addition, a lot of data for a variety of personal opinions and interests are generated exponentially. Some critical information from SNS data might generate a great impact to public opinion formation in various fields such as politics, economy, service, and entertainment. It is necessary to develop methods or algorithms which extract and process meaningful information from a large amount of data generated by the SNS. Also it is required to capture opinions in real time and to utilize this information for various application fields and to represent them with visualization.

We propose a big data processing method that can efficiently handle various unstructured data that collected from a lot of SNS data. Further, we suggest sentiment analysis algorithms, which can extract the sentiment information and classify preferences and changes of customers about a particular issues as time passes.

2 Related work

Most data generated on SNS service are unstructured data because data have not been standardized and its structure and shape are so complex unlike video image data and document data [1]. In order to extract meaningful information from a number of unstructured data on SNS, the process of unstructured data is needed. Various technologies for processing the unstructured data are studied focusing on morphological analysis. However, barriers to data analysis such as symbol word and new buzzword from the young people could exist. For this reason, big data processing and sensitive analysis using the computer has become more difficult.

Thus, researches on text mining extract information in the semi-structured or atypical text data based on the natural language processing techniques have been developed[5-7]. They are using statistical, periodic algorithm based on machine learning to extract meaningful information and to purify the information from the text data of the mass. In addition, research on opinion mining to determine the evaluation of positive, negative, neutral preference in the text has also been carried out [8-9].

Currently, a variety of open source projects for processing big data are in progress by naming ecosystem of Hadoop (Hadoop ECO system) [10]. Database that is used to process the big data, use NoSQL (Not-Only SQL) for storage and retrieval of data using the consistency model less restrictive than traditional relational databases [11]. As relational databases such as RDBMS, NoSQL uses a database depending on the situation. Many studies on the NoSQL database is underway in academia and industry current. Typically, Google BigTable, Amazon DynamoDB, Apache HBase of open source projects, Cassandra, MongoDB are representative [10][12][13][14][16]. In particular, MongoDB that are used in this study is classified to a CP type database with the Partition tolerance and Consistency based on the theory (Consistency, Availability, Partition tolerance) of CAP. It has been promoted as a source project.

The sentiment is emotion which we feel in mind and happen to some works or phenomena [16]. Sentiment Analysis is a process that discovers and extracts subjective information from the original data by utilizing computational linguistics, natural language processing and text analytics [16]. Studies that analyze the sentiment from big data have been developed[17-19]. Work to analyze the type of sentiment and classification, can be divided into three stages significantly. In the first step, the sentence in which sensibility information is included to express thoughts and feelings subjective is extracted. In the next step, the polarity of the sentence or document is classified like as positive,

negative or neutral. In final step, a classification of intensity determines subjectivity strength of text documents [20-21].

3 Sentiment Analysis of Unstructured SNS Data

3.1 System Model

We propose a big data processing system that can efficiently handle various unstructured SNS data. The proposed system is comprised of parallel HDFS(Hadoop Distributed File System) and MapReduce. Parallel HDFS that is based on the ecosystem of Hadoop is used to collect and save data reliably from a large variety SNS data. And MapReduce[22] is used to analyze large amounts of unstructured data for sentiment of user effectively. Configuration of the proposed system is shown in Figure 1.

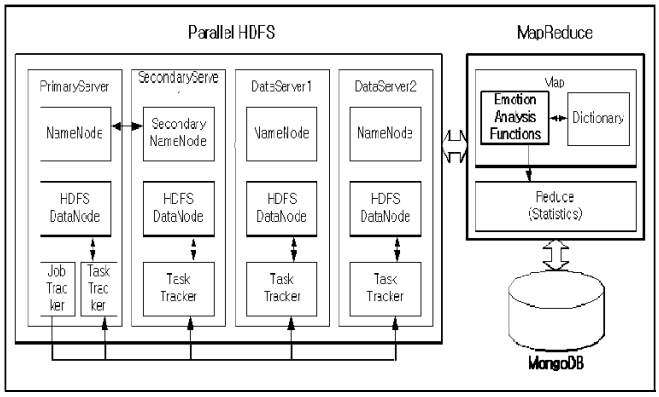


Figure 1. The proposed system

3.2 Composition of HDFS

HDFS is a file processing system which has the structure of distributed processing. It has been configured as a parallel server shown in Figure 1. The system is connected in parallel using four servers based on Linux and each chunk node to store data is set to 64MB. It duplicates the name server using the NFS for disaster recovery. Functions of the proposed servers are described in Table 1.

Table 1. HDFS Servers

Server	Components	Functions
PrimaryServer (Master Node)	Namenode, DataNode MapReduce, Crawler	Main server for parallel distribution process Name node(controlling other servers) Data node, Data loading
SecondaryServer (Slave Node 1)	Secondary NameNode DataNode	Backup server of main server Data node, Data loading
DataServer1 (Slave Node 2)	DataNode	Data node, Data loading
DataServer2 (Slave Node 3)	DataNode	Data node, Data loading

3.3 MapReduce Functions

MapReduce is a software framework developed by Google to support distributed computing and parallel programming using the concept of function called map. In this paper, it is classified into four special map functions. They perform positive/negative context analysis, morphological analysis, token analysis, prohibitive word analysis respectively. Table 2 shows 4 proposed functions and their operations.

Table 2. Functions of the proposed sentiment analysis

Sentiment analysis function	Operations	Referenced dictionary
Positive/negative context analysis function	context analysis using sentence pattern matching	positive/negative context dictionary
Morphological analysis function	elimination of needless elements, calculation of the result count	positive/negative word dictionary
Token analysis function	creation of tokens calculation of the result count	“
Prohibited words analysis function	calculation of the prohibited word score	prohibited word dictionary

First, it performs a positive/negative contextual analysis function. It examines the context by each sentence to enhance accuracy and is subjected to matching pattern with the negative context dictionary or the positive context dictionary. And it counts the number of positive and negative context, if the number of positive word is equal to the number of negative words, the sentence is treated as positive and it is transferred to the morphological analysis if the contextual analysis does not classify context. Algorithm for contextual analysis is shown as Figure 2.

Second, it performs a morphological analysis function. This function removes an unnecessary component such as special symbols in the analysis by using the morphological analyzer. And it counts by comparing the sentence to positive and negative clause dictionaries. If the value of positive counter is equal to that of negative counter, the sentence is treated as positive. If the morphological analyzer does not classify the polarity, the sentence is passed to the token analysis.

Third, it performs the token analysis. After separating tokens by space from the source sentence, the function counts positive word and negative word by comparing the negative and positive dictionaries. If the value of the positive counter is equal to that of negative counter, the sentence is treated as positive. If the token analysis does not classify the polarity, the sentence is passed to the prohibition word analysis.

Fourth, it performs prohibitive analysis. It calculates the prohibition score based on prohibition dictionary. Algorithm

for morphological analysis, token analysis and prohibition word analysis is described as Figure 3.

```
//Context Analysis
//input keyword, source
   //keyword: target word for decision of positive or negative sentiment
   //source: source data of text form that is processed by HDFS
   Input keyword and source
   Initialize result // a criteria for sentiment decision
//pre-processing
   Change the keyword to lower-case
   Change the source to lower-case
   Eliminate the needless characters in source text
   Initialize positive_count and negative_count
//Context Analysis
   Get the minimum sentence unit from the source
 //Computation of the positive  count and negative  count
   if (minimum sentence unit == positive) then positive  count++
   if( minimum sentence unit == negative) then negative  count++
   Repeat this step until there is no minimum sentence unit
//Computation of the result by positive  count and negative_count
   if (positive  count == 0 and negative_count == 0) then
       result = 0 //undecidable
   if (positive  count == negative_count) then
       result = 1 //positive
   else
       result = positive_count - negative_count
```

Figure 2. Context analysis function

3.4 Dictionaries for Sentiment Analysis

The proposed dictionaries use five MapReduce functions. They are a positive context Dictionary, a negative context dictionary, a positive word dictionary, a negative word dictionary and a prohibited word dictionary. In prohibition word dictionary, it is composed of polarity and score. The role of each dictionary is shown as Table 3.

```
//Morphological Analysis – if (result == 0) in previous stage
   Input source
   Initialize result-s //a criteria for sentiment decision
   //pre-processing source
   Eliminate the needless characters in source text
   Initialize positive  count  s and negative  count  s
   //Computation of the positive  count  s and negative_count_s using
   //the positive/negative word dictionary
   Compute positive  count  s, negative  count  s
   Repeat this step until there is no morpheme unit
   if (positive  count  s == 0 and negative  count  s == 0) then result-s=0
   if (positive  count  s == negative_count_s) then
      result-s = positive_count_s
   else
      result-s = positive  count  s - negative  count  s
//Token Analysis – if (result-s == 0) in previous stage
   Create tokens
   Initialize positive  count  s and negative  count  s
   //Computation of the positive  count  s and negative_count_s using
   // the positive/negative word dictionary
   Compute positive  count  s, negative  count_s
   Repeat this step until there is no token
   if (positive  count  s == 0 and negative  count  s == 0) then result-s=0
   if (positive  count  s == negative_count_s) then
      result-s = positive_count_s
   else
      result-s = positive  count  s - negative  count  s
//Prohibited word Analysis –if (result-s == 0) in previous stage
   //Computation of the positive  count_s and negative_count_s using
   // the prohibited word dictionary
   Compute positive  count  s, negative  count  s
   result-s = positive_count_s - negative_count_s
```

Figure 3. Analysis of Morphological, token and prohibited word

Table 3. Dictionaries for sentiment analysis

Dictionary	Role	application
Positive Context Dictionary	compute the number of positive context in source sentence / set of positive context patterns	Context Analysis
Negative Context Dictionary	compute the number of negative context in source sentence / set of negative context patterns	"
Positive Word Dictionary	compute the number of positive word in source sentence / set of positive word patterns	Morphological/Token Analysis
Negative Word Dictionary	compute the number of negative word in source sentence / set of negative word patterns	"
Prohibited Word Dictionary	compute the number of prohibited word in source sentence / set of prohibited words	Prohibited Word Analysis

4 Experiment and Results

4.1 Data Collection and Experimental Environment

Data collection performance of the proposed system is analyzed through the Twitter and Topsy. Topsy analyzes the activity of users in the SNS services such as Google Plus and Twitter. Topsy provides the analyzed data by analyzing about 500 millions of data per day. After the acquisition of the historical data, Twitter4j is used to collect data for continuous incremental data. Twitter provides one week data only and the key that may be used to query 450 for 15 minutes. In this study, a data collection module is to run every 4 hours using the crawler.

Experimental environment of the proposed system for performance analysis is described at Table 4. The proposed system consists of four Hadoop-based parallel servers and uses the 6.3 x64 CentOS as an operating system.

Table 4. Experimental environment

Components	Roles
OS, RE	Use of Hadoop for distributed storage Supporting Java environment for processing some business logic
Crawler, HDFS Layer	Crawler: Gathering the source data from various SNSs HDFS: Distribution File system, Data storage
MapReduce Layer	Sentence Analysis, Text Mining, Sentiment Analysis
MongoDB	Storing analyzed results by MapReduce in MongoDB
WAS, Web Server	Supporting Web applications using analyzed results

4.2 Analysis and Evaluation

The following four tests have been carried out to analyze the performance of the proposed system. First, it is an experiment of the system performance according to the number of data. The test of system load and acquisition time is performed using seven Twitter data sets at Table 5. Each data are collected using Topsy API.

Table 5. Data sets for experiment and analysis

Data Set number	Number of data	Extraction period (day)	API
1	2,106	1	Topsys API
2	11,672	6	"
3	20,000	10	"
4	40,788	20	"
5	79,080	36	"
6	90,014	44	"
7	100,497	52	"

Figure 4 shows a comparison of HDFS loading time and crawling time for each data set. Figure 5 and 6 shows the CPU load and memory load of each node in HDFS when each dataset has stacked and crawled. In the case of 2,106 dataset, crawl time is 6 seconds and HDFS loading time is 1 second. In the case of 100,497 dataset, crawl time is 70 seconds and HDFS loading time is 10 seconds as shown in Figure 4. The processing time is increased in HDFS loading time and crawl time in proportion to the number of data. Thus, the network load and the system load by collecting and stacking data show very close to the proposed system, the stable data collection and the data loading are processed in a few seconds.

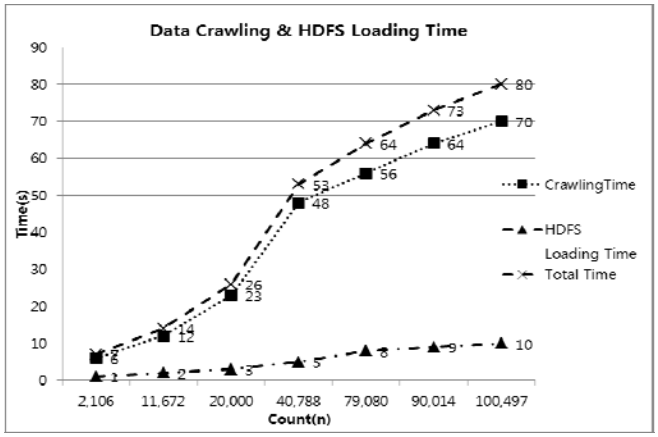


Figure 4. Crawling Time and HDFS Loading Time

The memory usage from slave node SN1 to slave node SN3 has used maximum 3.93% and minimum 0.03%. The master node M, has used from maximum 7.31% to minimum 0.6% as shown in Figure 5. Slave nodes use small memory resource by distributing loading the data. The master node uses more memory resource than the slave nodes.

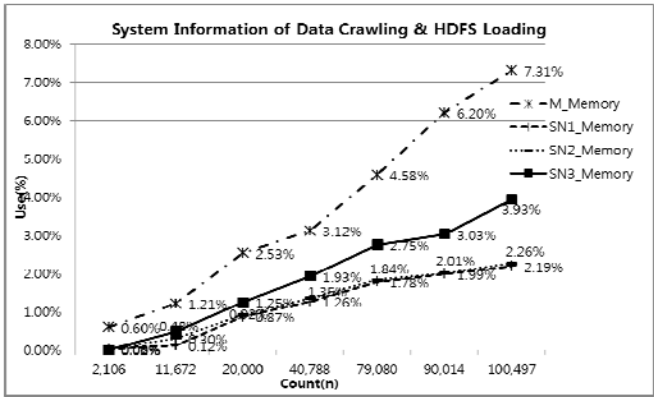


Figure 5. Memory Usage for Data Crawling and HDFS Loading

In Figure 6, slave node SN1 and slave node SN2 show that CPU usages are from maximum 2.8% to minimum 0.0%. But the slave node SN3 shows the CPU usage is from minimum 0.0% up to 11.4%. The reason is that the slave node SN3 loads data in parallel and distributed processing. The master node shows the CPU usage is from 5.0% up to 7.9%. Therefore, the proposed system provides a stable environment when it collects and loads data.

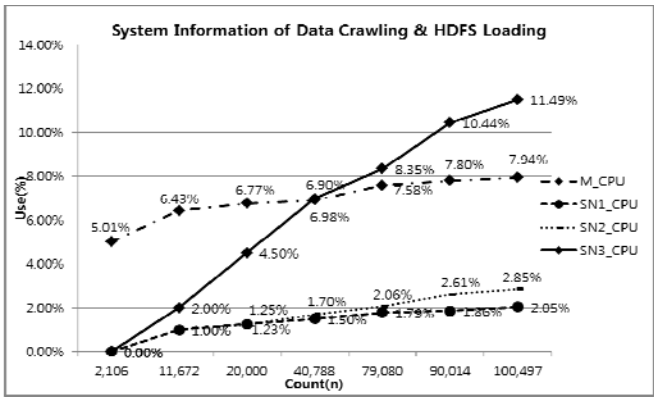


Figure 6. CPU Usage for Data Crawling and HDFS Loading

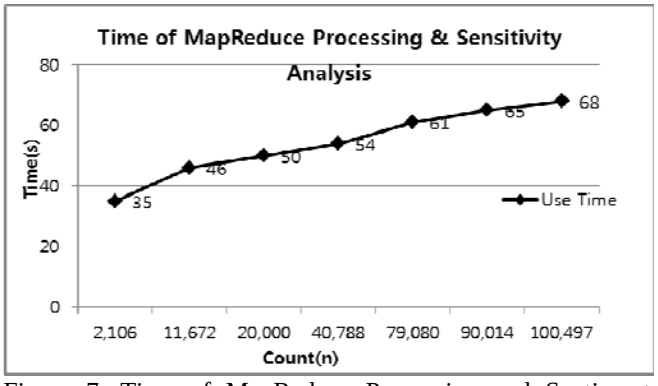


Figure 7. Time of MapReduce Processing and Sentiment Analysis

Sentiment analysis time and system load are tested by increasing the number of data. The experiment is executed in the degree of the system load and analysis time for sentiment analysis. Figure 7 shows the comparison of the sentiment analysis time for each data set. Figure 8 and 9 show memory load and CPU load for each node. The sentiment analysis time takes from 68 seconds to 35 seconds for each 7 data sets. The analysis time increases linearly to the number of data as shown in Figure 7.

In Figure 8, the master node does not process actual analysis but manage slave nodes. Its CPU usage is low when the slave nodes use most of the CPU resource. When the number of data set is less than 40,000, each slave node processes data in parallel. When the number of data set is greater than 40,000, all slave nodes utilize to maximize CPU resources according to the number of data. Therefore, the proposed system is performed stably as the number of data is increased. This is because the proposed system engages in parallel mode if CPU loads are increased. In Figure 9, the memory usage of the master node is low, but the load of the memory usage of slave nodes is distributed to each slave node and all slaves have balanced for the analysis. Therefore the proposed system distributes work load to slave nodes equally and maintains the load balancing. The system and algorithm of the propose method shows O(*n*) processing time. It provides a stable distributed analysis environment without processing by a single node.

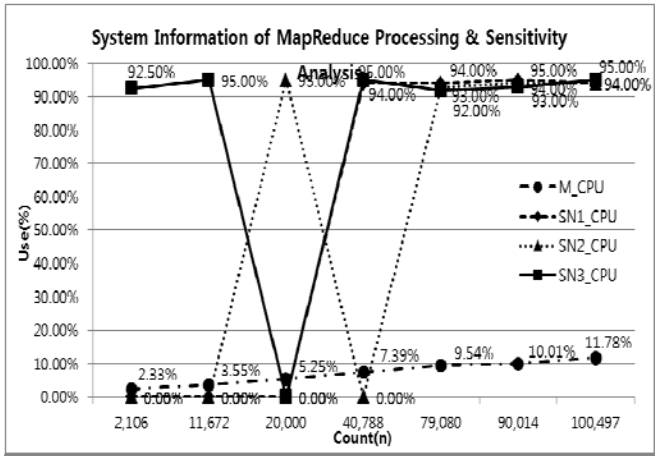


Figure 8. CPU Consumption of MapReduce Processing and Sensitivity Analysis

The accuracy of the sentiment analysis is measured. "Happy" word is used to analyze the sentiment. Figure 10 shows the comparison results of the proposed system and manual works. In Figure 10, error ratio of neutral sentiment is relatively high and the error rate for positive and negative sentiment is relatively small. The sentiment analysis results of the proposed system are very close to those of manual works.

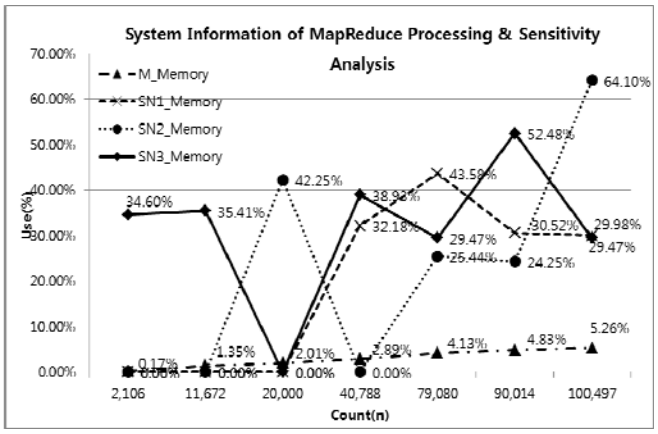


Figure 9. Memory Consumption of MapReduce Processing and Sentiment Analysis

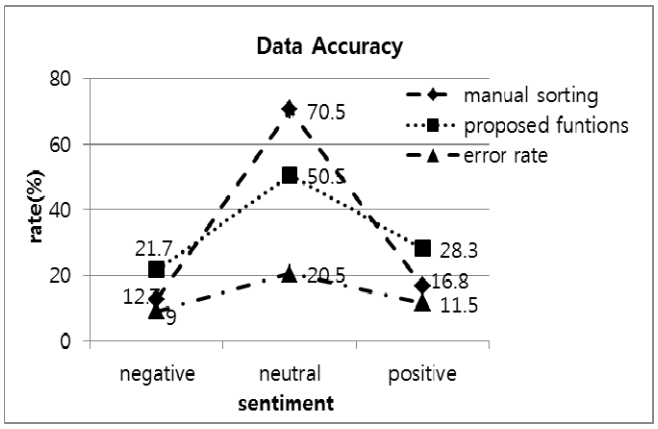


Figure 10. Comparison between the results of the proposed functions and the results of manual sorting

5 Conclusions

A big data processing system and algorithms are proposed to analyze the sentiment of users from the large amounts of unstructured data generated by SNS. The proposed system is composed of a parallel HDFS system based Hadoop Ecosystem and four primary special functions for the MapReduce. In addition, it uses the five types of data dictionary for sentiment analysis. The proposed system processes data with small loading time as the number of data increases. The analyzing works are not processed by one node, but distributed to all nodes for load balancing. When the proposed sentiment analysis functions have processed the data, the load of the system is distributed to all slave nodes equally. The sentiment analysis results of the proposed system are very close to those of manual works. Therefore the proposed system distributes work load to slave nodes equally and maintains the load balancing. Please address any questions of this paper to Byoungchul Ahn by Email (b.ahn@yu.ac.kr).

6 Acknowledgement

This work (Grants No. C0146250) was supported by Business for Cooperative R&D between Industry, Academy, and Research Institute funded Korea Small and Medium Business Administration in 2013.

7 References

[1] McKinsey, 2011, "Big Data: The Next Frontier for Innovation, Competition, and Productivity", [Online. McKinsey & Company, http://www.mckinsey.com/

[2] Chang-Shing Lee, Mei-Hui Wang, "Automated ontology construction for unstructured text documents", Data & Knowledge Engineering, Vol.60, Iss.3, pp.547-566, 2007

[3] B. Lee, J. Lim, J. Yoo, "Utilization of Social Media Analysis using Big Data", Jour. of the Korea Contents Association, Vol.13, No.2, pp.211-219, 2013

[4] M. Song, S. Kim, "A Study of improving on prediction model by analyzing method Big data", The Journal of Digital Policy & Management, Vol.11, No.6, pp.103-112, 2013

[5] Ah Tan, "Text mining: The state of the art and the challenges", Proc. of the PAKDD 1999, 1999

[6] Q. Mei, C. Xhai, "Discovering evolutionary theme patterns from text: an exploration of temporal text mining", Proc. of the 11th ACM SIGKDD international conference on knowledge discovery in data mining, pp.198-207, 2005

[7] K. Park, K. Hwang, "A Bio-Text Mining System Based on Natural Language Processing", Jour. of KISS: computing practices, Vol.17, No.4, pp.205-213, 2011

[8] B. Pang, L. Lee, "Opinion Mining and Sentiment Analysis", Foundations and Trends in Information Retrieval, Vol.2, No.1-2, pp.1-135, 2008

[9] B. Kang, M. Song, "A Study on Opinion Mining of Newspaper Texts based on Topic Modeling", Jour. of the Korean Library and Information Science Society, Vol.47, No.4, pp.315-334, 2013

[10] http://hadoop.apache.org/

[11] Jing Han, Kian Du, "Survey on NoSQL database", Proc. of 6th International Conference on Pervasive Computing and Applications(ICPCA), pp.363-366, 2011

[12] Fay Chang, R.E. Gruber, "Bigtable: A Distributed Storage System for Structured Data", ACM Transations on Computer System, Vol.26, Iss.2, 2008

[13] S. Sivasubramanian, "Amazon dynamoDB: a seamlessly scalable non-relational database service", Proc. of the 2012 ACM SIGMOD'12, pp.729-730, 2012

[14] Lars George, "HBase: The Definitive Guide", O'REILLY, 2011

[15] Kristina Chodorow, "MongoDB: The Definitive Guide 2nd Edition", O'REILLY, 2013

[16] B. Pang, ,L. Lee, "Opinion Mining and Sentiment Analysis," Foundations and Trends in Information Retrieval: Vol.2, No.1-2,pp.1-135, 2008

[17] S. Mukherjee, P. Bhattacharyya, "Sentiment Analysis in Twitter with Lightweight Discourse Analysis", Proc. of COLING 2012, pp.1847-1864, 2012

[18] N. Godbole, S. Skiena, "Large-Scale Sentiment Analysis for News and Blogs", Proc. of the ICWSM'2007, 2007

[19] A. Pak, P. Paroubek, "Twitter as a Corpus for Sentiment Analysis and Opinion Mining", Proc. of the LREC'2010, 2010

[20] H. Tang, S. Tan, X. Cheng, " A survey on sentiment detection of reviews," Expert Systems with Applications, Vol.36, pp.10760-10773, 2009

[21] Seth Gilbert, Nancy Lynch, Brewer's conjecture and the feasibility of consistent, available, partition-tolerant web services, ACM SIGACT New 33(2), pp. 51-59, 2002.

[22] J. Dean, S. Ghemawat, "MapReduce; Simplified Data Processing on Large Clusters", Communications of the ACM, Vol.51, No.1, pp.107-113, 2008

E-commerce evolution based on the perspective of network effects: A multi-case study in China

Peng Shao, Ping Hu, Jie Qi
School of management, Xi'an Jiaotong University, Xi'an, China

Abstract—*This paper, which is based on network effect theory, focused on the key factors of two-sided network construction and e-commerce ecosystem formation in a big data environment. It takes six e-commerce companies as a research case and suggests how to improve the timing and spacing of service precision and business credit, reduce the experience gap between online and offline commerce, increase the members in e-commerce networks, and ameliorate the form of e-commerce ecosystems. Based on the several propositions, we discuss e-commerce evolution in four stages and propose new perspectives on two-sided network effect improvement in the big data era.*

Keywords: big data; e-commerce; two-sided networks; network effects

1 Introductions

With the wide application of a new generation of information technology, such as social networking, cloud computing, and mobile commerce, global data is growing at an unprecedented speed [1]. *Nature* published a special "Big Data" issue in September 2008, and *Science* launched its own special issue in February 2011 to explain the importance of big data in scientific research. Meanwhile, the academics of China have also become interested in big data related areas, such as business management, technological innovation, and interdisciplinary research. There are large amounts of data in e-commerce that can be used to segment the market and meet customer needs with customized products and services. In a network situation, customer behavioral data in all channels and all stages of growth can be recorded or collected by enterprises, and accurate and quantifiable marketing strategies can be designed [2].

E-commerce is a two-sided market; the research on two-sided markets usually takes pricing strategy as a starting point; it analyzes the impact of supply and demand characteristics on the profits of platforms, for which on optimal pricing strategy is developed. E-commerce platforms have a two-sided market effect and the interdependence of the two sides play a driving role in development of platforms. Unstructured data, such as relational, location, video data, and images, emerge in large numbers in the big data era and function as the driving resource in the e-commerce model transformation. The existence of a two-sided network effect is based on the demand to resolve big data, and new information technology can facilitate the matching of supply and demand in platforms. Based on the two-sided network theory, research propositions are made in this paper through a case study of six e-commerce businesses; further, the evolution of an e-commerce model is discussed in the four stages of the big data era, and a new perspective on the improvement of the two-sided network effect is developed.

2 Theoretical Review

Network effects are largely influenced by the number of users. Katz and Shapiro (1985) [3] point out that there are direct and indirect network effects. The term direct network effect refers to increases in product value with increases in the number of users, including new users; this increase is known as a demand-side network effect. Indirect network effect refers to the product value derived from the number of complementary products. Indirect network effects will be strong when there are various kinds of complementary products, and the consumption of

core products and the value of other products will increase. In addition, network effect can be divided into local network effects and global network effects. Local network effects are caused by family and friends and global network effects by the installed base. As a consumer's utility depends on the number of interactions with his or her friends, rather than the size of the overall network [4], in markets with network effects, companies seek to remove features of its original products and sell degraded versions of them at low prices or for free [5].

There are two kinds of users in a two-sided market, and value is created by the interaction between consumers and businesses on the platform [6]. The revenue of participants on the platform depends on the number of participants. Cross-platform network effects should be considered when business strategies are formulated so that these can be rational. Generally, network product markets have the following characteristics in two-sided network effects [7-9]: First, user groups are often very concerned about the scale of the other side of the platform because their needs are interdependent. Second, each member of a group on one side is willing to join the network only when the group members of the other side are expected to join it. Third, the platform sponsor is often quite inclined to a price structure that matches the needs of the two-sides, since the demands of the users on both sides have an asymmetrical interdependence.

Proposition 1: In a two-sided market, the utility of players on one side is dependent on the number of players on the other side.

Proposition 1a: In a two-sided market, the utility of business is dependent on the number of consumers.

Proposition 1b: In a two-sided market, the utility of consumers is dependent on the number of businesses.

3 Study Design

TABLE1 SAMPLE DESCRIPTION

type	Sample Description	Differential & advantage	Data acquisition
Group-buying & mobile commerce	Meituan.com, which was founded in March 2010, is the No.1 website in the group-buying industry in China.	Services accuracy: location based service, information push	Interview with Marketing PR Director on August 10, 2012; secondhand data collection
	Komovie.cn, which was founded in July 2011, is a startup in the mobile commerce industry; it's main business is movie tickets with dynamic prices.	Services accuracy: based on the remaining time, dynamic pricing on the remaining number of seats	Interview with CEO on August 10, 2012; secondhand data collection
Foreign trade e-commerce	Dhgate.com , which was founded in 2004，is a foreign trade platform for small businesses, based on online trading	Credit: mechanism of honesty and guarantee, seller management, credit files, sunshine plans	Interview with vice-president on August 10, 2012; secondhand data collection
	Globalmarket.com, which was founded in 2000， is a international e-commerce platform with self-developed audit standards.	Credit: eight items of global manufacturer certificate	Interview with CEO on July 27, 2012; secondhand data collection
Combining of online & offline	500ccc.com, which was founded in 2011，is based on the model of "one city onenetwork andtransaction and distribution services offline."	Minimize the experience gap between online and offline: "one city one network" to solve the last mile problem of large appliances	Interview with vice-president on August 9, 2012; secondhand data collection
E-commerce ecosystem	Alibaba.com, which was founded in 1999，announced the launch of seven business groups in 2012, it acquired shares of Sina Weibo in 2013.	Ecosystem:resource integration Extensive,third-party service providers settled in the ecosystem	Interview with director of Taobao President Office on August 19, 2013; secondhand data collection

Based on the network theory, research propositions are made in this paper through the case study of six e-commerce businesses, and a new perspective on the improvement of the two-sided network effect is developed. The case study is the elementary method in management research [10], and it is suitable for observation and research on the series changes of businesses [11]; it coincides with the study of the evolution of e-business models in this paper. The criterion of sample selection in the case study is particularity rather than the generality; four to eight is the appropriate number of cases in a multi-case study [12]. Semi-structured interviews are proposed, based on theory and literature review after the cases are selected. There are four interviewers for each interview, one of whom is the local contact person; one is responsible for questioning and clarification, and the remaining two are responsible for photographing and recording. After the interview, cases were saved in Word format, based on written notes and audio recordings. The findings of interview recording are combined with those of published studies, such as internal reports, journal articles, and speeches in order to ensure the reliability and validity of the collected data.

4 Analysis on the e-commerce revolution

4.1. Services accuracy improved by data mining

With the emergence of new technology, such as blogs, social networks, location based services, and the rise of cloud computing, networking, and other technologies, data is growing and accumulating at an unprecedented rate. There are many kinds of data; the most common are unstructured data for fairly long periods, such as relational data, location data, images, and videos.

> The founder of komovie.cn: We know the positions, phone models, general consumption periods, and the consumption capacities of consumers. And we also know the time they enter and come out cinemas, and what kinds of movie they like. All this data can be used to provide precise marketing services.
>
> The Marketing Director of Meituan.com: The two-sided market will appear gradually . Omn the consumer side, Meituan.com want users to be able to enjoy good services at anytime. on the business side, Meituan.com is establishing a long-term relationships between businessesed and consumers.

A large amount of data is generated every day by e-commerce, not only those that serve as records of objective phenomena or numerous unordered values, but also data with special meaning and value. Marketing management combines art and science; the scientific part depends on various data collection methods and marketing databases. Depending on the characteristics of consumption and behavior, modern technologies, methods, and strategies are adopted by enterprises in order to achieve the goal of marketing communication among consumer groups. The two-sides of an e-commerce platform involve consumers and businesses; further, an important issue concerns the precise matching of supply and demand. Information mining by mass businesses to meet consumer needs requires not only various types of data analysis, but convenient channels for information transfer between consumers and businesses.

Proposition 2: The impact of consumer numbers on business utility can be enhanced by improvement of service accuracy in the two dimensions of time and space.

4.2. Transaction risk reduced by business credit rating

Big data means high value, which may result from the mass convergence of data with minimum value and the useful data needed for management decision-making. E-commerce transactions are conducted between suppliers and consumers who do not know each other; information asymmetry is an important factor that affects the development of e-commerce. Credibility identification is particularly vital in foreign trade e-commerce because of asymmetric information.

> The Vice president of dhgate.com: Sellers om the management department will review business licenses, registered capital, products, and other information to build up credibility files. Various phenomena of false credit are listed in the "unshine" project, which was set up in 2012; the project asks businesses to conduct self-examinations. If the same complaint is received again, a more severe punishment is applied the businesses.
>
> The CEO of globalmarket.com: It is hard to distinguish sellers with credit; the credit system (GMC) was launched to solve this problem. A survey is conducted and the result shows that there are 100,000 Chinese enterprises with GMC standards, accounting to 90% of exports in China. Globalmarket.com uses their energy to service valuable customers, including Dell, IKEA, and so on.

Online service providers create better services by the online information that they provide, but they can also use the information to despoil users; thus, the key issue is to establish ethics on the Internet and to ensure that these ethics do not inhibit innovation [13]. Buyers are not only concerned about the number of sellers but also by comments on goods. The more positive comments, the better the products and services provided by sellers; in this way, direct network effects become greater. An online reputation is an important factor that affects e-commerce development; credit conditions should be enhanced in e-commerce transactions between buyers and sellers in order to increase the rate of online trading. More traders will join a platform through active participation when fair credit environments exist on them.

Proposition 3: The impact of business numbers on consumer utility can be enhanced by the improvement of business credit.

4.3. The experience gap minimized by the integration of online and offline commerce

The relationship between online commence and traditional commence is the hot issue in e-commerce research [14]. The level of e-commerce development varies by area, as does the e-commerce model and the perspective on the relationship between online and

traditional commerce. Overall, existing research findings can be divided into four perspectives: alternative, promotional, complementary, and independent [15]. With the rapid development of e-commerce, customer migration from stores is unprecedented; the consumer transformation has become a reality. Small bits of data are merged into big data. E-commerce can more easily accumulate and mine data than offline channels and should provide services based on this data processing.

> The founder of 500ccc.com: There is a big difference between online and offline in appliance selling. 1:10 is the difference between online and offline in the selling of clothes and 1:100 is the maximum in appliances. Order online and service offline, the one city-one network model can solve the last mile problem of large appliances.

In the e-commerce era, the dual-channel strategy will be the best choice for manufacturers to improve the market competitiveness of enterprises and increase customer demand. The integration of online and offline commerce enhances the ability to penetrate markets and creates a new value transfer mode [16]. The integration of online and offline commerce depends on shortening the experience gap shorten these two sectors. The experience gap will shorten by the improvement of customer experiences and service levels, such as through online bookings, online payments, online counseling, online customization, online complaints, and other service experiences. By shortening the gap between experiences, more consumers will purchase offline products or service online, thus increasing the network effect of the impact of the number of businesses on consumer utility.

Proposition 4: The impact of business numbers on consumer utility can be enhanced by minimizing the experience gap between online and offline commerce.

4.4 Mutualism promoted by the ecosystem of e-commerce

The characteristics and attributes of biological ecosystems are helpful to understand business ecosystems [17]. Every player has its own task in an e-commerce ecosystem, which are intertwined to form a complete network. After several decades of development, the Alibaba Group has accumulated hundreds of millions of consumers, sellers, and numerous third-party service providers; all these players promote logistics and online payment development. In January 2013, the Alibaba Group divided the e-commerce business into three categories and 25 business units; the first is vertical businesses BU, including Taobao, Tmall, and Juhuasuan; the second is the infrastructure platform, including cloud computing, payments, logistics, and so on; the third category is the sharing platform, which is responsible for information sharing between vertical businesses and the platform. The coordination between vertical business and the platform is achieved through internal open source implementation; the individual needs of vertical businesses and the common needs of the foundation platform can be fully met.

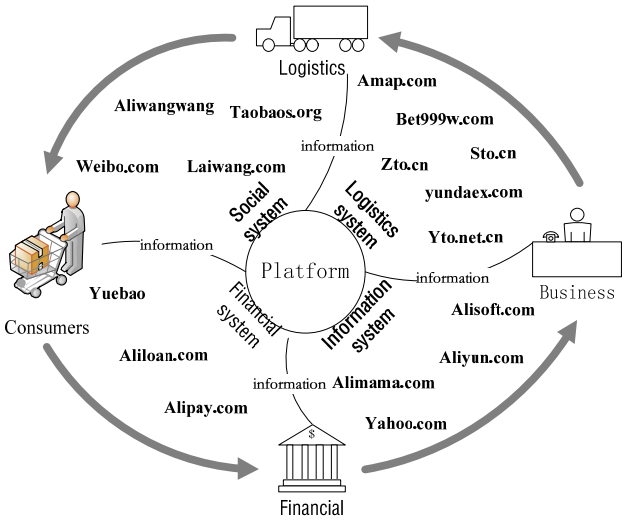


Figure1. Alibaba's ecosystem

Alibaba and third-party service providers settled in the platform carry out their duties for e-commerce services, and a complete ecosystem is created. (1) The social ecosystem: Alibaba acquired the preferred shares of Sina Weibo in 2013; this deal, which marks a milestone in the history of e-commerce, extends to online social networks. Sina Weibo has hundreds of millions of users; information can be quickly spread on the Weibo platform, which is a weak-tie social network; a strong-tie social network, named Laiwang, was also launched by Alibaba. (2) The financial ecosystem: Alibaba uses the credit and behavioral data accumulated by B2B, Taobao, Alipay, and other e-commerce platforms, network models, and videos; a credit investigation mode was introduced to confirm the authenticity of customer information. Customer behavior data is transformed into personal credit evaluation, and then those who are usually unable to obtain loans in the traditional banking channels can achieve small loans

online. (3) The logistics ecosystem: Cainiao was jointly established by Alibaba and other logistics companies in 2013; an open social storage facilities network was formed all over China through self-build cooperation reconstruction. In addition, Alibaba become a shareholder of amap.com; it opened up location information and user data in order to promote the development of the O2O business. (4) Information services: Alisoft provides personalized software services for SMEs, based on the SaaS model. Alimama built a platform for small websites and e-commerce for advertising cooperation. Aliyun and Yahoo facilitate the obtaining of information for users; thus, consumers are able to search for thousands of products, and search cost is reduced.

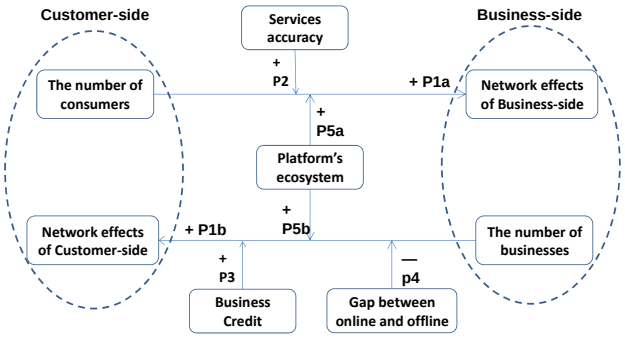


Figure 2. The two-sided network effect model of e-commerce

Proposition 5: The two-sided network effect can be enhanced by the formation of an e-commerce ecosystem.

Proposition 5a: The impact of business numbers on consumer utility can be enhanced by the formation of an e-commerce ecosystem.

Proposition 5b: The impact of business numbers on consumer utility can be enhanced by the formation of an e-commerce ecosystem.

5 Discussion and Prospect

The formation of an e-commerce ecosystem has impacts on business; enterprises specialize in different areas to offer products that reflect their capabilities. Enterprises need communication and collaboration with other enterprise in the system to improve its value. The propositions developed in the case study emphasize the different stages of e-commerce. The discussion conducted on e-commerce evolution is based on four stages.

1) Introduction stage: New technology, such as mobile commerce and location based services, record consumer behavior in the dimensions of time and space. Marketing accuracy can be improved by data mining during this stage in order to attract businesses and consumers and boost sales. Big data is able to provide sufficient space for personalized business applications, based on individual consumer behavior and preferences data, the future enterprise may provide personalized products and services according to different interests and preferences for each consumer [18].

2) Growing stage: Unexpected advantage is achieved by technology,， but only the most fit survive. It is hard to avoid the growing stage as mixed participants join the plantfrom. This is also the transition phase of platfrom, when it moves from barbaric growth to brand reputation building. Negetive comments are easier to spread than positive ones. Rigorous review mechanisms are semt to the market to avoid adverse selection problems in the market for lemons.

3) Adjustment stage: With the developmnt of e-commerce, online purchases are acheiced by PC and mobile commerce. Offline purchases become bottlenecks in the development of e-commerce. The traditional market faces changes in the interacion of online and offline commerce. In this suitation, business systems open up. Online and offline system integration become an important issue in order to solve O2O problems.

4) Mature stage: There are contestable assumptions in market theory: freedom in entering, no technogical inferiority of later entrants，and no sunk costs for those who leave. As the barriers to entry in the e-commerce ecosystem are almost zero, many business are involved in the e-commerce ecosystem. The existence of the scope economic effect promotes the opening up of the platform, as, for example, in the open API interface; this encourages third-party developers to create applications and to provide privacy processed data to research institutions and corporations.

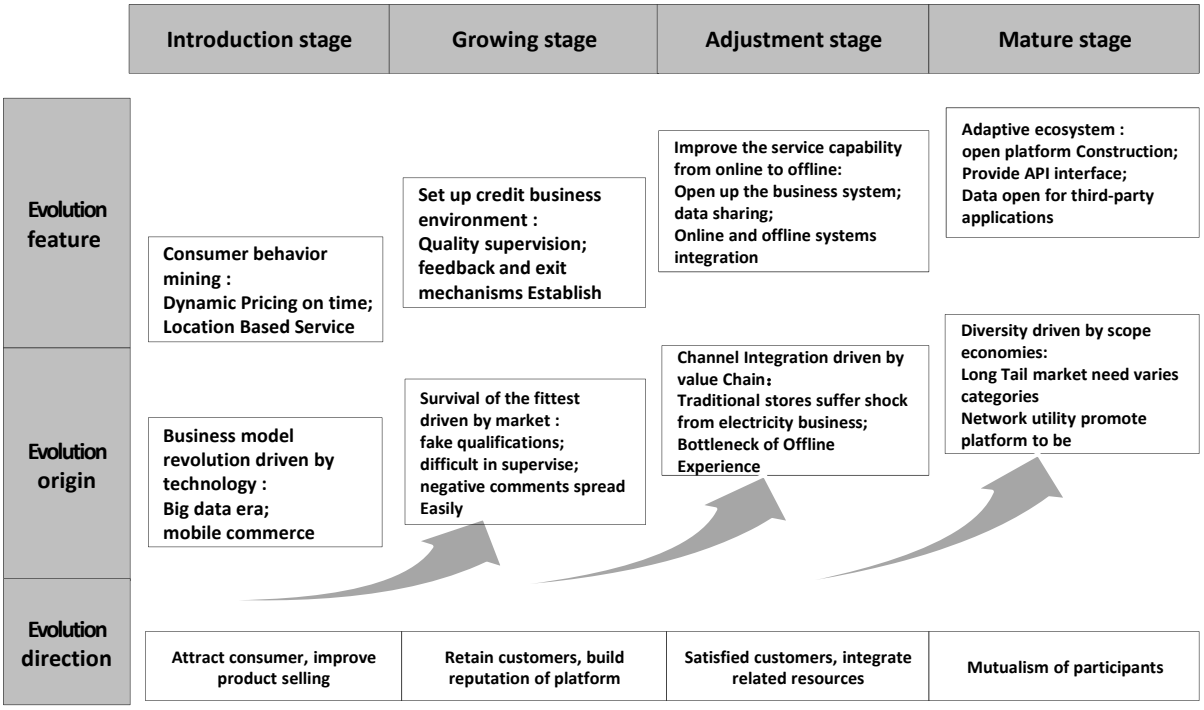


Figure3 revolution of e-commerce model

6 Conclusion

E-commerce is a business model with network effects; thus, how to utilize network effects is the important issue in e-commerce development. Based on network effect theory, a case study was conducted on six e-commerce corporations and two-sided network effects in e-commerce were explored. The proposed propositions included the following: the impact of consumer numbers on business utility can be enhanced by improvement of service accuracy in the two dimensions of time and space. The impact of business numbers on consumer utility can be enhanced by the improvement of business credit and by the minimization of the experience gap between online and offline commerce. The two-sided network effect can be enhanced by the formation of an e-commerce ecosystem. The research model was based on the above propositions, and e-commerce model evolution was discussed during the four stages of big data era.

With the applications of web2.0 and the emergence of big data, there are many areas still worth exploring and improving in research and practice: redundancy in personalized recommendations and visualization in retrieval; fake credit and credit evaluation mechanisms based on network relationships; the emergence of network behaviors and supply chain coordination between online and offline; herd behavior in networks and coordination of the e-commerce ecosystem.

ACKNOWLEDGMENT

This work was supported by the Electronic and Information Department of the Chinese Academy of Engineering in 2013 (2013-XX-03）

7 References

[1] Lynch C. Big Data: How do your data grow? [J]. Nature, 2008, 455 (7209): 28-29.

[2] Hui S K, Fader P S, Bradlow E T. Path data in marketing: An integrative framework and prospectus for model building [J]. Marketing Science, 2009, 28(2) : 320-335.

[3] Katz M, Shapiro C. Network externalities, competition, and compatibility [J]. The American Economic Review, 1985,75(3): 424-440

[4] Corrocher N, Zirulia L. Me and you and everyone we know: An empirical analysis of local network effects in mobile communication [J].Telecommunications Policy, 2009, 33(1): 68-79

[5] Csorba G, Hahn J H. Functional degradation and asymmetric network effects [J].The Journal of Industrial Economics, 2006, 54(2) :253-268

[6] Wright J. The determinants of optimal interchange fees in payment systems[J]. Journal of Industrial Economics, 2004, 52:1- 26.

[7] Hagiu A. Two- sided platforms: product variety and pricing structures [J]. Journal of Economics & Management Strategy, 2009, 18(4):1011-1143.

[8] Weyl E G. A Price theory of multi-sided platforms [J]. American Economic Review, 2009, 100(4):1642-1672.

[9] Bolt W, Tieman A F. Heavily skewed pricing in two-Sided markets [J] .International Journal of Industrial Organization, 2008, 26(5):1250-1255.

[10] Yin R K. The case study crisis: Some answers [J] .Administrative Science Quarterly, 1981, 26:58-56.

[11] Pettigrew A. Longitudinal Field Research on Change: Theory and practice [J] .Organization Science, 1990, 1(3):267-292

[12] Eisenhardt K M. Build theories from case study research [J] .Academy of Management Review, 1989,14(4):532-550.

[13] Marc B. Interview with Hal Varian Chief Economist at Google [J]. Communications & Strategies, 2012, 88(4):127-128

[14] Welteveden J W J. Substitution or complementarity? How the Internet changes city centre shopping [J]. Journal of Retailing and Consumer Services, 2007,14(3): 192-207.

[15] Mokhtarian P L. Telecommunications and travel: the case for complementarity [J]. Journal of Industrial Ecology, 2002, 6(2):43-57.

[16] Tedeschi B. Compressed Data: Big companies go slowly in devising net strategy [N]. New York Times, 2000, 3-27.

[17] Iansiti M, Levien R. Strategy as ecology [J]. Harvard business review, 2004, 3:68-78.

[18] Gobble M M. Big Data: The Next Big Thing in Innovation[J]. Research-Technology Management , 2013 :64-66

Estimating the Visit and Content Association of Websites with their Weblogs

Hyun-Ho Lee,
Yeonsung University
South Korea,
hhlee@yeonsung.ac.kr

Jong-Min Lee,
Hanyang University
South Korea,
jmlee@visionlab.or.kr

Nam-Hun Park,
Anyang University
South Korea,
nmhnpark@gmail.com

Won-Suk Lee,
Yonsei University
South Korea,
leewoo@gmail.com

***Abstract*—Recently, big data has been spotlighted as an important emerging issue. As a case study of big data analysis, this paper estimates websites by analyzing their weblogs and content meta-database (meta-DB). As one of the representative types of big data, a weblog keeps track of users' click streams. The analysis goal is to derive the possible integration (or cooperative utilization) implications of websites by measuring their association aspects, such as content association and visit association. In particular, while estimating websites through the available practical records of their users, in order to extract statistical information, the proposed technique will incorporate data mining (association rule) methods to extract statistical information.**

***Keywords*—Big data; weblog; meta-DB; visit association; content association; data mining**.

I. INTRODUCTION

People today receive a flood of information provided through the numerous websites. However, due to the breakdown of information and inconsistent way in which it overlaps, inefficiencies occur in terms of the delivery of organized and systematic information [1]. In particular, for public information services in some specific areas, it is important to ensure consistency and efficiency through collaboration between institutions, while avoiding excessive competition among information service providers.

Recently, the topic of big data has been emphasized as an important emerging issue. As one of the representative forms of big data, weblogs keep track of users' click streams (i.e., where they go on a website and how long they stay). In this paper, in order to estimate the popularity of websites that offer cultural information services for a large number of public institutions related to Korean culture management, weblogs and a web content meta-database (meta-DB) are analyzed. The goal of this analysis is to derive the integration (or cooperative utilization) implications between websites by measuring associative aspects such as their *visit association* and *content association*. A *visit* is defined as the collected click activities of a single user on the website. If particular websites are visited by the same user in the same time period, it can be said that there is *visit association* between them. A high visit association between websites implies meaningful integration (or cooperation utilization). From this point, it can be inferred that they would able to generate synergy through their integration (or cooperative utilization). Meanwhile, *content association* can be measured by estimating the degree of similarity among the keywords representing the individual content. Since a high level of content association between websites means that their content is redundant, their integration (or cooperative utilization) can be considered. As a case study of these ideas, this paper examines websites that provide Korean public cultural information services. Based on measuring their visit and content association, the implications of effective integration (or cooperative utilization) strategies for related websites are derived.

Contributions The contributions of this paper can be summarized as follows:

- It provides a case study analysis for a weblog, one of the representative types of big data;
- Rather than methods such as questionnaire surveys, it suggests a method of analysis based on the actual use records of website users; and
- It applies data mining techniques to the analysis of website utilization.

Paper Outline Section II mentions related works, while Section III illustrates how to define the use association of a website. In Section IV, as a case study, an example of use association analysis is introduced, together with the implications of the results of the analysis. Finally, Section V presents our conclusions.

II. Related Works

Various methods have been proposed to estimate the use-quality of a website. Critical factors associated with website success were identified in [2], in the context of electronic commerce (EC). The research framework was derived from information systems and marketing literature; webmasters from Fortune 1000 companies were used as the target group in a survey. Four factors that were found to be critical to website success in EC were identified, as follows: 1) information and service quality, 2) system use, 3) playfulness, and 4) system design quality. In addition, [3] presented a case study on the use of benchmarking to determine how one organization's website compares to the websites of related schools and similar professional organizations. The results of the benchmarking study provided a measure of how a given website compared to the sites of related organizations, as well as ideas on how this may be further enhanced and evaluated regularly. Moreover, [5] introduced a knowledge base (KB) consisting of a DB-type repository for maintaining patterns and rules as an independent program that consults the pattern repository. Using the proposed architecture, either an artificial system or a human user can consult the KB in order to improve the relationship between a website and its visitors.

In [7], intelligent web caching algorithms were introduced that employed predictive models of web requests; the general idea was to extend the least recently used (LRU) policy of web and proxy servers by making them sensitive to web access models that had been extracted from weblog data using data mining techniques. Two approaches have been studied in particular, namely frequent patterns and decision trees. The experimental results of the new algorithms show substantial improvement over existing LRU-based caching techniques in terms of *hit rate*. Web optimization in a fog computing context focusing on a user's web page rendering performance was proposed in [8]. In addition, [9] implemented an MKD-webserver rating application in order to analyze web performance through the SSL aimed at secure communications over the Internet. However, in contrast to the present paper, [7,8,9] focused on developing the algorithms that analyze *or* improve the performance of a web-server rather than use quality.

Estimation of the popularity of a website through the available practical records of its users continues to be performed at the level of extracting statistical information, such as the number of visits and number of visitors. In particular, the case for the utilization analysis of websites taking advantage of data mining techniques such as the association rule [10,11], representing the approach proposed here, can scarcely be found in the literature.

III. Definitions of Website Association

A. Visit association

Visit association is determined by analyzing all the websites that a user visits in a single period of time. A high simultaneous-visit rate among websites implies that there would be a benefit to their joint utilization. In order to analyze the visit association of websites, some formulas are defined as follows: For websites X and Y, the term $X \rightarrow Y$ would indicate that Y is also visited when a user visits X. A sequence of visits by a single user within a certain period of time is defined as a *visit transaction*. In this paper, only visit transactions that comprise visits to more than one website are targeted, as a one-site visit transaction cannot provide information about visit association. Let T_x and T_y be the visit respective transactions of the sites X and Y, and let $T_{x,y}$ be the visit transaction that includes both sites. The visit ratio for the transaction T_x, denoted by $p(X)$, is defined as follows:

$$p(X) = |T_x| / |T_{tot}|, \quad (1)$$

where $|T_x|$ is the number of T_x visit transactions and $|T_{tot}|$ is the total number of visit transactions.

In the same way, the visit ratio for the transaction $T_{x,y}$, denoted by $p(X,Y)$, is

$$p(X,Y) = |T_{x,y}| / |Ttot|, \quad (2)$$

where $|T_{x,y}|$ is the number of $T_{x,y}$ visit transactions and $|T_{tot}|$ is the total number of visit transactions.

The measurement of the visit association of site Y in relation to site X—which is to say, the degree of visit association between them—as denoted by $assoc_v(X,Y)$, is defined as follows:

$$assoc_v (X \rightarrow Y) = p(X, Y) / p(X). \quad (3)$$

When measuring the visit association between two websites, it is necessary to analyze it in both directions. In other words, both $assoc_v(X \rightarrow Y)$ and $assoc_v(Y \rightarrow X)$ should both be considered, because either $assoc_v(X \rightarrow Y)$ or $assoc_v(Y \rightarrow X)$ can be high even when the visit association between sites X and Y is low. Such a phenomenon mainly occurs when there is a large difference between $|Tx|$ and $|Ty|$. For example, if site Y is a large, well-known site and site X is a small, unknown site, $assoc_v(X \rightarrow Y)$ can be high even though the visit association between X and Y is low because the high $assoc_v(X \rightarrow Y)$ value can be solely attributable to a high $p(Y)$ value. This is similar to the causes of confidence distortion in the association rule generation methods of data mining. It can be resolved by measuring the mutual visit association between the websites. The degree of cross-visit association of sites X and Y, denoted by $cross\_ assoc_v(X, Y)$, is defined as follows:

$cross\_assoc_v(X, Y) = (assoc_v(X \rightarrow Y), assoc_v(Y \rightarrow X))$. (4)

To summarize, both of the degrees that make up the cross-visit association between the sites are high if and only if the cross-visit association degree is high.

B. Content association

In this paper, content association analysis was conducted for the meta-DB content of the culture portal site that was investigated. In order to promote itself as a reference point for the Culture and Information Service, the culture portal site finds the content of each cultural information site that can strengthen its search function, then matches the keyword entered by the user to provide a link that can direct him or her to that site. In order to provide these features, the information from each cultural site is listed on the culture portal site, where the meta-information content of each site is located. Therefore, the content of the culture portal meta-DB may be viewed as a collection of the meta-information on the content included in each culture website.

The method of content association defined in this paper measures the similarity between different contents in the context of what each includes. To accomplish this, we have focused on a set of keywords that are defined for each item of content. In other words, the more common keywords there are between the contents, the higher their similarity is judged to be. For content A and B, let $keyword(A)$ be a set of keywords in content A. The degree to which the content of B is associated with A, denoted by $assoc_c(A \rightarrow B)$, is defined as follows:

$$assoc_c(A \rightarrow B) = | keyword(A) \cap keyword(B) | / |keyword(A)|, \quad (5)$$

where $|keyword(A)|$ is the number of keywords included in content A.

Based on the definition of content association, the degree of content association of site Y with site X, denoted by $assoc_c(X \rightarrow Y)$, is estimated by measuring the similarity between the two respective complete sets of keywords extracted from the total contents of sites X and Y. Accordingly, it is defined as follows:

$$assoc_c(X \rightarrow Y) = \sum_{A \in X, B \in Y} |keyword(A) \cap keyword(B)| \; / \; \sum_{A \in X} |keyword(A)|. \quad (6)$$

Like visit association, content association should be considered mutually. In other words, from sites X and Y, both $assoc_c(X \rightarrow Y)$ and $assoc_c(Y \rightarrow X)$ should be considered together. The reason for this is the same as for visit association. Accordingly, the cross-content association between sites X and Y, denoted by $cross\_assoc_c(X, Y)$, is defined as follows:

$$cross\_assoc_c(X, Y) = (assoc_c(X \rightarrow Y), assoc_c(Y \rightarrow X)). \quad (7)$$

To summarize, both of the degrees that make up cross-content association between the sites are high if and only if the actual cross-content association degree is high.

IV. Analysis of Website Association

As mentioned in Section I, visit association between websites can be measured by specifying a series of websites that have been visited by a specific user in a single period of time. In this paper, in order to estimate the associations of websites in the field of cultural information services, we have analyzed the weblogs of a culture portal site (http://www.culture.go.kr). As a representative Korean culture portal, this is a typical search-service site. Due to the nature of search-service sites, its logs have many visit records for several culture-oriented websites that can be searched through it. Accordingly, it is possible to estimate the visit associations for various websites through its weblogs. In addition, since it possesses a meta-DB for the content of culture-oriented websites, it is possible to analyze the content associations for various websites through its content meta-DB. In this paper, we have analyzed the associations of the culture-oriented websites with the weblogs of the culture portal site from three viewpoints, as follows:

- Estimating the visit association of the culture-oriented websites and deriving implications by analyzing the weblogs of the culture portal site;
- Estimating the content association of the culture-oriented websites and deriving implications by analyzing the meta-DB content of the culture portal site; and
- Deriving implications for the integration (or cooperative utilization) of the culture-oriented websites by analyzing the correlations of both their visit associations and content associations.

A. Visit association

Analysis results In order to analyze the visit association, we examined the weblog of the culture portal site. The period of collection is one year. Table 1 provides an overview of the weblog. As shown in this table, for the *59* frequently visited websites, there are *2,022,508* visit transactions extracted from *19,535,333* effective logs. For each visit transaction, the average number of different sites visited is *2.7*. This indicates that, on average, a user visits *2.7* websites at the same time. Table 2 gives the analysis results for visit association, displaying the top 20 website pairings with the highest degree of cross-visit association.

TABLE 1. AN OVERVIEW OF THE ANALYZED WEBLOG

Websites (#)	Collection period	Log records(#)	Visit transactions(#)	Websites(#) per a transaction
59	1 year	19,535,333	2,022,508	2.7

Implications The implications of the results of analysis are as described below.

- In terms of the cultural information field, the cultural industry field accounts for the topmost rankings. From the perspective of website type, the content-providing type is ranked highest, meaning that many users visit such websites, based on the search results of the culture portal site.
- The visit association between online bookstore sites (e.g., Hotaruen Runisu, Kyobo, YongPung, and YES24) is ranked very high.
- If the websites of Table 2 are grouped based on their cross-visit association degree, *36* of the *59* websites are separated into two groups when the degree threshold is not less than *(0.1, 0.1)*. In particular, four online bookstore sites show very high visit association. None of the cross-visit association degrees between them were less than *(0.9, 0.9)*. In the case where the degree was no less than *(0.45, 0.45)*, *20* of the sites were separated into four groups. Overall, the websites that shared a group had similar characteristics in their content with respect to cultural information fields such as books, cultural people and arts, cultural heritage, and tourism.

TABLE 2. ANALYSIS RESULTS OF CROSS-VISIT ASSOCIATION (TOP 20)

Rank-ing	Site *X*	Site *Y*	*p(X,Y)*	$assoc_v$ *(X→Y)*	$assoc_v$ *(Y→X)*
1	Bandi&Luni's Bookstore	Kyobo Bookstore	*1.9*	*100*	*100*
2	Bandi&Luni's Bookstore	YoungPung Bookstore	*1.9*	*100*	*100*
3	Bandi&Luni's Bookstore	YES24 Bookstore	*1.9*	*100*	*93.3*
4	Kyobo Bookstore	YES24 Bookstore	*1.9*	*100*	*93.3*
5	YES24 Bookstore	YoungPung Bookstore	*1.5*	*90.1*	*90.6*
6	Kyobo Bookstore	YoungPung Bookstore	*1.5*	*92.7*	*90.6*
7	Arts Road	Standard Portrait	*21.8*	*70.6*	*80*
8	National Culture Symbol 100	Standard Portrait	*19.9*	*70.7*	*89.5*
9	Korean Cultural Person	Standard Portrait	*20*	*61.1*	*73.3*
10	Korean Movie Database	WeCon	*26.3*	*60.2*	*70.5*
11	National Culture Symbol 100	Korean Cultural Person	*15.8*	*63.7*	*58.5*
12	Jeju Customs Tourism Unabridged	WeCon	*23.4*	*57.4*	*54.4*
13	Jeju Customs Tourism Unabridged	Korean Movie Database	*26.8*	*66*	*53.9*
14	Jeju Customs Tourism Unabridged	Tourism Knowledge & Information System	*29.3*	*52.9*	*64.1*
15	Arts Road	National Culture Symbol 100	*16.9*	*54.6*	*75.2*
16	e-Museum	National Memory Heritage Service	*14.6*	*53.3*	*72.7*
17	National Memory Heritage Service	National Folk Museum of Korea	*10*	*50*	*57.9*
18	Asiart Jeonju Tour	Seoul Arts Center	*25.9*	*49.6*	*50.7*
19	Arts Road	Korean Cultural Person	*15.3*	*48.4*	*57.4*
20	Korean Pattern	National Memory Heritage Service	*8.3*	*54.6*	*48.2*

- The cross-visit association among the websites (standard portrait, a hundred ethnic cultures, Culture People site of Korea) that the same institution (Ministry of Culture, Sports and Tourism) manages is very high. The cross-visit association between WeCon (managed by the Korea Creative Content Agency) and Korean Movie Databases (managed by the Korea Institute Visual Materials) was high. The cross-visit association between iTourSeoul (managed by Seoul Tourism Marketing) and VisitKorea (managed by the Korea Tourism Organization) was also high. This means that the website pairs were closely associated with each other in terms of their utilization, even though they are managed by different organizations. In particular, the case of iTourSeoul and VisitKorea showed a high utilization association between a private and public website.

B. Content association

Analysis results In order to analyze the content association between websites, we used a list of keywords defined in the meta-DB for each content. Table 3 shows an overview of the meta-DB content. The total number of websites that have more than one type of content in their entry in the meta-DB of the culture portal website was *1,226*. The number of websites with more than *1,000* keywords was *27*. The total number of websites was *4,709,194*. However, since no keywords were defined for *88.5%* of the websites, the number of analyzed sites with actual keywords was *541,557*. The average number of keywords per website was *88.2*. Table 4 shows the top 20 website pairs in terms of the cross-content association.

TABLE 3. AN OVERVIEW OF THE ANALYZED META-DB

Websites (#)	Collection period	Contents (#)	Keywords(#) per content	Null ratio of content keywords
27 (1,226)	1 year	4,709,194	88.2	88.5

Implications The implications of analysis result are as described below.

- From the perspective of the cumulative number of content keywords per website, e-Museum (managed by the National Museum), the National Folk Museum, art, VisitKorea (managed by the Korean Tourism Organization) have a great number of content keywords in their mentioned order.
- The content association degree between websites was ranked highly between the Seoul Arts Center and the Gyeonggi Cultural Foundation, between Arts Load and the Seoul Arts Center, and between the National Theatre and the Seoul Arts Center. Overall, the degrees of content association between websites that were classified into the same cultural information fields were relatively high.
- If the websites in Table 4 were grouped based on their cross-content association degrees, *22* of the *27* websites were surrounded by a single group when the degree was no less than *(0.5, 0.5)*. If the degree threshold was higher, the websites that had a relatively low degree were eliminated. If the degree threshold was no less than *(0.75, 0.75)*, seven websites were separated into two groups. One consisted of five websites (the Seoul Arts Center, Gyeonggi Cultural Foundation, National Theatre, Culture and Art Information Service, and Arts Load). The other consisted of two websites (e-Museum and the National Folk Museum). In particular, the pairing of the Seoul Arts Center and Gyeonggi Cultural Foundation showed a very high degree of cross-content association, at more than *(0.8, 0.8)*.

TABLE 4. ANALYSIS RESULTS OF CROSS-CONTENT ASSOCIATION (TOP 20)

Ranking	Site *X*	Site *Y*	$assoc_c$ *(X→Y)*	$assoc_c$ *(Y→X)*
1	Seoul Arts Center	Gyeonggi Cultural Foundation	*83.3*	*81.4*
2	Arts Road	Seoul Arts Center	*83.2*	*78.7*
3	National Theatre	Seoul Arts Center	*81.5*	*77.4*
4	National Folk Museum of Korea	e-Museum	*85.8*	*78.0*
5	Culture & Art Information Service	Seoul Arts Center	*80.1*	*76.4*
6	Arts Road	Gyeonggi Cultural Foundation	*80.2*	*76.2*
7	Culture & Art Information Service	Gyeonggi Cultural Foundation	*76.9*	*75.3*
8	Arts Road	Culture & Art Information Service	*78.7*	*75.3*
9	Arts Council Korea	National Gugak Center	*74.8*	*74.4*
10	National Theatre	Arts Road	*79.6*	*75.2*
11	National Theatre	Culture & Art Information Service	*77.0*	*73.0*
12	Seoul Arts Center	e-Museum	*74.3*	*72.4*
13	Arts Council Korea	National Theatre	*76.5*	*72.6*
14	Korean Film Council	Korea Culture Information Service Agency	*71.2*	*71.1*
15	Gyeonggi Cultural Foundation	e-Museum	*79.9*	*72.9*
16	Arts Council Korea	Seoul Arts Center	*84.9*	*74.2*
17	Seoul Cultural Foundation	Arts Council Korea	*72.0*	*71.2*
18	Korea Culture Information Service Agency	Seoul Cultural Foundation	*70.3*	*70.0*
19	National Theatre	Gyeonggi Cultural Foundation	*79.3*	*71.4*
20	Korea Culture Information Service Agency	Arts Council Korea	*70.8*	*69.5*

- For the content of the websites, since the proportion that defined their keywords was very low, the reliability of the content association analysis results could be relatively low as well. However, if accurate and systematic meta-information (keywords, classification information, etc.) had been supported, it would have been possible to derive highly reliable analysis results.

TABLE 5. ANALYSIS RESULTS OF CROSS-VISIT ASSOCIATION TO INVESTIGATE THE CORRELATION BETWEEN VISIT AND CONTENT ASSOCIATION (TOP 20)

Ranking	Site *X*	Site *Y*	$p(X,Y)$	$assoc_v$ *(X→Y)*	$assoc_v$ *(Y→X)*
1	Arts Council Korea	Press Arbitration Commission	*4.7*	*50.9*	*48.9*
2	Seoul Arts Center	Tourism Knowledge & Information System	*10.7*	*45.8*	*48.3*
3	Arts Council Korea	National Theatre	*4.4*	*47.6*	*41.9*
4	National Theatre	Press Arbitration Commission	*4.2*	*40.0*	*43.6*
5	Arts Council Korea	National Folk Museum of Korea	*4.7*	*51.0*	*38.7*
6	Bucheon Cultural Foundation	Korea Publication Ethics Commission	*1.7*	*36.6*	*50.5*
7	e-Museum	Seoul Arts Center	*8.4*	*36.2*	*35.7*
8	Arts Council Korea	Bucheon Cultural Foundation	*3.2*	*34.5*	*68.5*
9	Bucheon Cultural Foundation	Press Arbitration Commission	*3.2*	*69.8*	*33.7*
10	e-Museum	Tourism Knowledge & Information System	*7.5*	*32.6*	*34.0*
11	National Folk Museum of Korea	Press Arbitration Commission	*3.8*	*31.6*	*40.1*
12	National Theatre	Tourism Knowledge & Information System	*6.2*	*59.7*	*28.1*
13	National Folk Museum of Korea	National Theatre	*3.5*	*29.0*	*33.7*
14	e-Museum	National Folk Museum of Korea	*6.3*	*27.2*	*52.0*
15	Bucheon Cultural Foundation	National Theatre	*3.0*	*64.4*	*28.5*
16	Press Arbitration Commission	Seoul Arts Center	*6.7*	*70.3*	*28.8*
17	Press Arbitration Commission	Tourism Knowledge & Information System	*6.0*	*62.2*	*26.9*

18	Arts Council Korea	Tourism Knowledge & Information System	*6.1*	*66.8*	*27.7*
19	National Theatre	Seoul Arts Center	*6.2*	*58.9*	*26.3*
20	National Folk Museum of Korea	Seoul Arts Center	*6.0*	*49.6*	*25.7*

C. Correlation between visit and content associations

Analysis results In order to analyze the correlation between the visit and content associations of websites, we examined the cross-visit associations among the only 25 websites that were analyzed in both Section IV.A (59 websites) and Section IV.B (27 websites). Table 5 shows the top 20 website pairs with high degrees of cross-visit association. Based on the cross-content association analysis results in Table 4 and the cross-visit association analysis results in Table 5, the analyzed websites were separated into four groups according to the distribution of cross-visit and cross-content association degrees with each other. Based on the analysis results, the threshold of the degree of cross-visit association was set to *(0.1, 0.1)* and the threshold of the degree of cross-content association was set to *(0.7, 0.7)*. Table 6 shows the correlation between the analysis results for both association degrees.

TABLE 6. CORRELATION BETWEEN CONTENT AND VISIT ASSOCIATION

(A) LIST OF WEBSITES

Content association \ Visit association	High	Low
High	Seoul Arts Center, e-Museum, Arts Council Korea, National Theatre, National Folk Museum of Korea, Seoul Cultural Foundation, Korea Culture Information Service Agency, Gyeonggi Cultural Foundation	Culture & Art Information Service, Arts Road, WeCon, National Museum of Korea, Korean Film Council, National Gugak Center
Low	Tourism Knowledge & Information System, Press Arbitration Commission, Bucheon Cultural Foundation, Korea Publication Ethics Commission, Cultural Heritage Administration, Korea Copyright Commission, VisitKorea	

(B) PROPER STRATEGY FOR INTEGRATION (OR COOPERATIVE UTILIZATION)

Content Association \ Visit association	High	Low
High	Integration of websites	Cooperative utilization
Low	Integration of content-DB	No strategy

Implications The implications of the results of the analysis are as described below.

- As shown in Table 6, of the websites that belonged to either the group that had a high cross-visit association (*15* websites) or the group that had a high content-visit association (*14* websites), *eight* websites belonged to both groups. This meant that there was no absolute correlation between the visit and content associations, even though the content association had some impact on the visit association.
- *Eight* websites in Table 6(a) with both a *high* cross-content association degree and *high* cross-visit association degree can be assumed to be used simultaneously by many users, as they have a similar type of content. Accordingly, strategies for their integration could be considered, with expectations of the synergistic effect of integration (*see* Table 6(b)).
- *Seven* websites in Table 6(a) with *low* cross-content association degrees and *high* cross-visit association degrees can be interpreted to be used simultaneously by many users, even though their content differs. Accordingly, a cooperative utilization strategy can be considered for them rather than integration, with the anticipation of a synergistic effect of cooperative utilization (*see* Table 6(b)).
- *Six* websites in Table 6(a) with *high* cross-content association degree and *low* cross-visit association degree can be interpreted to be rarely used simultaneously by many users, even though their contents are similar. Accordingly, an integration strategy for their content can be considered rather than the integration (or cooperative utilization) of the websites, with the expectation that the efficient common use of resources and the augmentation of consistency in the content (*see* Table 6(b)).

V. CONCLUSIONS

The present paper examined websites by analyzing their weblogs and content meta-DBs. The websites analyzed were ones that provide information related to aspects of Korean culture and are managed by public institutions; weblogs and web-content meta-DBs were considered. By investigating the visit and content associations of the websites defined in this paper, the integration (or cooperative utilization) implications for these websites were derived. It is particularly meaningful

that data-mining techniques were applied to the estimation of websites through the available practical records of their users.

In future work, more detailed and systematic methodologies should be utilized in order to enhance the accuracy and variety of the website analyses, including the standardization of the analytical methods and the effective design of the website-utilization knowledge repositories [6]. In addition, it will be important to design the framework and workflow in such a way that real-time analysis is enabled through distributed computing environments such as Hadoop.

ACKNOWLEDGEMENTS

This research was supported by Basic Science Research Programs (NRF-2012R1A1A2009170) through the National Research Foundation of Korea(NRF) funded by the Ministry of Science, ICT and Future Planning.

REFERENCES

[1] Madu, Christian N., and Madu, Assunpta A., "Dimension of e-Quality, International Journal of Quality & Reliability Management", Vol.19(3), pp.246-258, 2002.

[2] Liu, C., and Arnett, K. P., "Exploring the Factors Associated with Web Site Success in the Context of Electronic Commerce", Information and Management, Vol.38(1), pp.23-33, 2000.

[3] Misic, M. M. and Johnson, K., "A Tool for Web Site Evaluation and Improvement", Internet Research, Vol.9(5), pp.383-392, 1999.

[4] J.D.Velasquez, H.Yasuda, T.Aoki, and R.Weber, "A genetic data mart architecture to support web mining", Proceedings of the 4th International Conference on Data Mining, pp.389-399, 2003.

[5] Juan D. Velasquez, and Vasile Palade, "A knowledge base for the maintenance of knowledge extracted from web data", Knowledge-Based System, Vol.20, pp.238-248, 2007.

[6] J. Debenham, "Knowldege base maintenance through knowledge representation", Proceedings of the 21th International Conference on Database and Expert Systems Applications, pp.599-608, 2001.

[7] F. Bonchi, F.Giannotti, C.Gozzi, G.Manco, M.Nanni, D.Pedreschi, C.Renso, and S.Ruggieri, "Web log data warehousing and mining for intelligent web caching", Data & Knowledge Engineering, Vol.39(2), pp.165-189, 2001.

[8] Jiang Zhu, Chan, D.S., Prabhu, M.S., Hao Hu, and Bonomi, F., "Improving Web Sites Performance Using Edge Servers in Fog Computing Architecture", IEEE 7th International Symposium on Service Oriented System Engineering(SOSE), pp.320-323, 2013.

[9] Veereshkumar M Kolli, Vinaykumar M Kolli, and Vaishakh B., "Implementation of MKD-WebServer Rating Application for Analysis of Web performance through SSL", International Journal of Advanced Research in Computer Engineering & Technology (IJARCET), pp.314-318, 2013.

[10] Chengqi Zhang, Shichao Zhang, "Association rule mining : models and algorithms", Lecture notes in computer science(2307), pp.229-236, 2002.

[11] Adamo, Jean-Marc, "Data mining for assciation rules and sequential patterns", New York : Springer, 2000.

SESSION

ALGORITHMS FOR BIG DATA

Chair(s)

TBA

NOISE BENEFITS IN CONVOLUTIONAL NEURAL NETWORKS

Kartik Audhkhasi, Osonde Osoba, Bart Kosko

Signal and Information Processing Institute
Electrical Engineering Department
University of Southern California, Los Angeles, CA
Email: kosko@sipi.usc.edu

ABSTRACT

We prove that noise speeds convergence in the back-propagation (BP) training of a convolutional neural network (CNN). CNNs are a popular model for large-scale image recognition. The proof builds on two recent theorems. The first result shows that the BP algorithm is a special case of the Expectation-Maximization (EM) algorithm. The second result states when adding noise to the data will speed convergence of the EM algorithm. Then this noisy EM algorithm (NEM) algorithm gives a simple geometrical condition for a noise speed up in the BP training of a CNN. Noise added to the output neurons speeds up CNN training if the noise lies above a hyperplane that passes through the origin. Simulations on the MNIST digit recognition data set show that the noisy BP algorithm reduces the mean per-iteration training-set cross entropy by 39% compared with the noiseless BP. The noisy BP algorithm also reduces the mean per-iteration training-set classification error by 47%. The noise benefit is more pronounced for small data sets.

Index Terms— Convolutional neural network, back-propagation algorithm, Expectation-Maximization (EM) algorithm, Noisy EM algorithm, noise benefit, stochastic resonance.

1. NOISE INJECTION IN CONVOLUTIONAL NEURAL NETWORKS

We prove that noise speeds up convergence in the popular back-propagation (BP) training algorithm [1] of a convolutional neural network (CNN) [2, 3]. CNNs are standard neural network systems for large-scale image recognition [4–9]. Figure 1 shows a 3-layer CNN with one hidden convolutional layer and 3 convolution masks. The proof builds on our previous result [10] that the backpropagation algorithm is a special case of the Expectation-Maximization (EM) algorithm [11]. We then use the recent noisy EM (NEM) algorithm [12, 13] that gives a sufficient condition to speed up convergence in the EM algorithm. Then the NEM algorithm's sufficient condition gives a simple geometrical sufficient condition for a noise speed up in the BP training of a CNN.

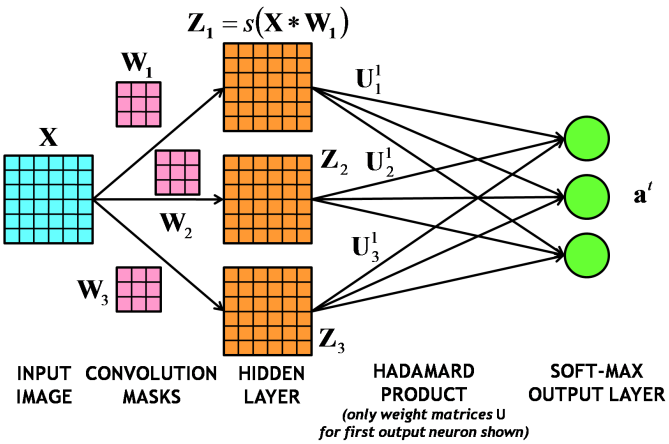


Fig. 1. Convolutional Neural Network (CNN): The figure shows a CNN with just one hidden layer. The input image $\mathbf{X}$ convolves with 3 masks $\mathbf{W}_1, \mathbf{W}_2$, and $\mathbf{W}_3$. These masks act as receptive fields in the retina. The resulting images pass pixel-wise through logistic sigmoid functions s that give the hidden neuron activations. Then the CNN computes element-wise Hadamard products between the hidden neuron activation matrices $\mathbf{Z}_1, \mathbf{Z}_2$, and $\mathbf{Z}_3$ with weight matrices $\mathbf{U}_j^k$ where $j = 1, 2, 3$ and $k = 1, 2, 3$. The soft-max Gibbs signal function gives the activations of the output layer neurons.

Figure 2 shows the noise-benefit region for a CNN with three output neurons. Noise added to the output neurons speeds up the BP training of a CNN if the noise lies above a hyperplane that passes through the origin of the noise space. This is a simple linear condition that the noise must satisfy. The output layer activation vector $\mathbf{a}^t$ determines the normal to the hyperplane.

Figure 3 shows the training-set cross entropy of a CNN using standard noiseless BP, BP with blind noise (Blind-BP), and BP with NEM noise (NEM-BP). Noisy back-propagation reduces the average training-set cross entropy by 39.26% compared with noiseless back-propagation. Figure 4 plots the training-set classification error rates as the system trains. The testing-set classification error rate is essentially the same at

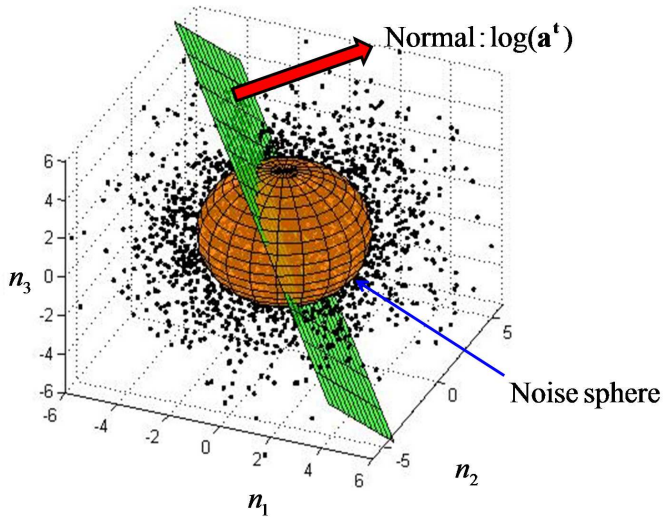


Fig. 2. Noise-benefit region for a CNN with soft-max output neurons: Noise speeds up maximum-likelihood parameter estimation of the CNN with soft-max output neurons if the noise lies above a CNN-based hyperplane that passes through the origin of the noise space. The activation signal $\mathbf{a}^t$ of the output layer controls the normal to the hyperplane. The hyperplane changes as learning proceeds because the parameters and hidden-layer neuron activations change. We used independent and identically distributed (i.i.d.) Gaussian noise with mean 0, variance 3, and $(3, 1, 1)$ as the normal to the hyperplane.

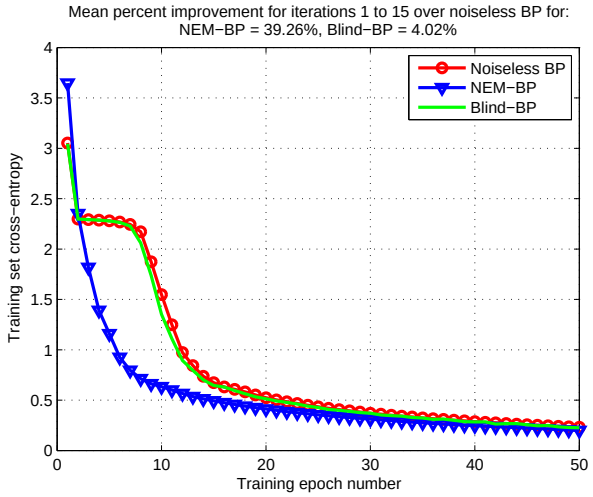


Fig. 3. NEM noise-benefit in BP training of a CNN using MNIST data: The figure shows that the NEM-BP training reduces the average training-set cross entropy of the MNIST data set compared with standard noiseless BP training. We obtain a 39.26% average reduction in cross entropy for the NEM-BP case when compared with the standard BP over the first 15 training iterations. Adding blind noise gives a minor average reduction of 4.02% in cross entropy. Training used 1000 images from the MNIST data for a CNN with one convolution hidden layer. The convolutional layer used three 3×3 masks or filters. Factor-2 downsampling followed the convolutional layer by removing all even index rows and columns of the hidden neuron images. The hidden layer fully connects to 10 output neurons that predict the class label of the input digit. We used uniform noise over $[-0.5/\sqrt{t^5}, 0.5/\sqrt{t^5}]$ where t is the training iteration number for both NEM and blind noise.

convergence. NEM-BP gives a 47.43% reduction in training-set error rate averaged over the first 15 iterations compared with noiseless BP. Adding blind noise only slightly improves cross entropy and classification accuracy. Figure 5 shows a noise-benefit inverted U-curve for NEM-BP training of a CNN on the MNIST data set. This inverted U-curve is the signature of a nonlinear noise benefit or so-called *stochastic resonance* [14–23]. The optimal uniform noise scale occurs at 1. NEM noise hurts CNN training when the noise scale increases beyond 2.6.

The next section presents an overview of CNNs. Section 3 presents the back-propagation algorithm for CNN training. Theorem 1 shows that the BP algorithm is a special case of the generalized EM (GEM) algorithm. Section 4 reviews the NEM algorithm. Section 5 presents the NEM-BP algorithm for CNN training. Section 6 summarizes the simulations on the MNIST data set.

2. CONVOLUTIONAL NEURAL NETWORKS

A convolutional neural network (CNN) convolves the input data with a set of filters. This is a rough analogy to the use of receptive fields in the retina [24] as in the Neocognitron network [25]. Consider a CNN with one hidden layer for simplicity. The notation extends directly to allow multiple hidden layers. Let $\mathbf{X}$ denote the input 2-dimensional data of size $M_X \times N_X$ where M_X and N_X are positive integers. Consider 2D filters $\mathbf{W}_1, \ldots, \mathbf{W}_J$ each of size $M_W \times N_W$. The convolution of $\mathbf{X}$ with the filter $\mathbf{W}_j$ gives

$$\mathbf{C}_j = \mathbf{X} \circledast \mathbf{W}_j \tag{1}$$

where $\circledast$ denotes 2D convolution. The 2D data matrix $\mathbf{C}_j$ has size $(M_X + M_W - 1) \times (N_X + N_Y - 1)$ with (m, n)-th entry

$$\mathbf{C}_j(m, n) = \sum_{a=1}^{M_W} \sum_{b=1}^{N_W} \mathbf{X}(a - m, b - n)\mathbf{W}_j(a, b) \; . \tag{2}$$

Pad $\mathbf{X}$ with zeros to define it at all points in the above double sum. Then pass the J matrices $\mathbf{C}_1, \ldots, \mathbf{C}_J$ element-wise through logistic sigmoid function s to give the hidden-neuron

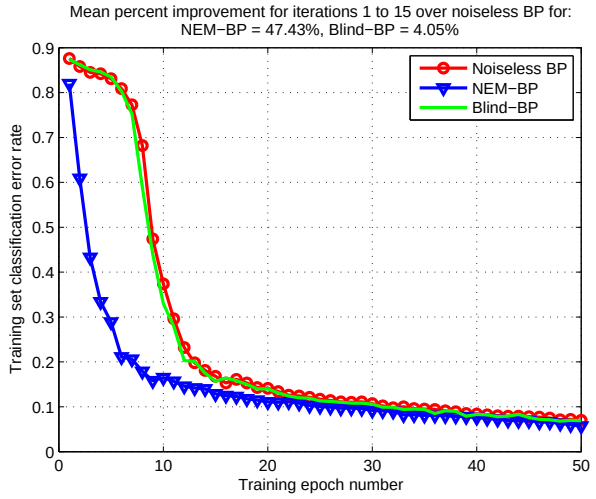


Fig. 4. NEM noise-benefit in BP training of a CNN using MNIST data: The figure shows that the NEM-BP training reduces the training-set classification error rate of the MNIST data set compared with standard noiseless BP training. We obtain a 47.43% average reduction in classification error rate for the NEM-BP case when compared with the standard BP over the first 15 training iterations. Adding blind noise gives a minor average reduction of 4.05% in classification error rate. Training used 1000 images from the MNIST data for a CNN with one convolution hidden layer. The convolutional layer used three 3×3 masks or filters. Factor-2 downsampling followed the convolutional layer by removing all even index rows and columns of the hidden neuron images. The hidden layer fully connects to 10 output neurons that predict the class label of the input digit. We used uniform noise over $[-0.5/\sqrt{t^5}, 0.5/\sqrt{t^5}]$ where t is the training iteration number for both NEM and blind noise.

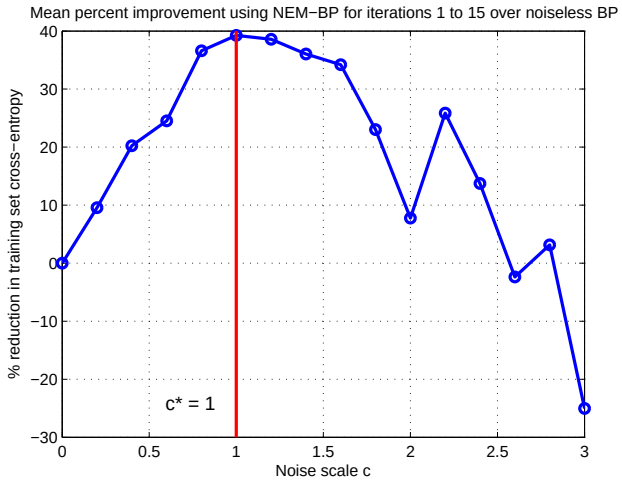


Fig. 5. NEM noise-benefit inverted U-curve for NEM-BP training of a CNN: The figure shows the mean percent reduction in per-iteration training-set cross entropy for NEM-BP training of a CNN with different uniform noise variances. We add zero mean uniform$(-0.5\sqrt{c/t^d}, 0.5\sqrt{c/t^d})$ noise where $c = 0, 0.2, \ldots, 2.8, 3$, t is the training epoch, and $d = 5$ is the noise annealing factor. The noise benefit increases when c increases from 0 to 1 and tends to decrease after 1. The optimal noise scale is $c^* = 1$. NEM noise addition hurts the training-set cross entropy when $c \geqslant 2.6$.

activations $\mathbf{Z}_j$:

$$\mathbf{Z}_j(m,n) = s(\mathbf{C}_j(m,n)) \tag{3}$$

$$= \frac{1}{1 + \exp(-\mathbf{C}_j(m,n))} \,. \tag{4}$$

Suppose the network has K output neurons. A $(M_X + M_W - 1) \times (N_X + N_Y - 1)$ weight matrix $\mathbf{U}_j^k$ multiplies the j-th hidden neuron matrix $\mathbf{Z}_j$ element-wise. The soft-max or Gibbs activation of the k-th output neuron is

$$a_k^t = \frac{\exp\left(\sum_{j=1}^{J} \mathbf{e}^T \mathbf{Z}_j \odot \mathbf{U}_j^k \mathbf{e}\right)}{\sum_{k_1=1}^{K} \exp\left(\sum_{j=1}^{J} \mathbf{e}^T \mathbf{Z}_j \odot \mathbf{U}_j^{k_1} \mathbf{e}\right)} \tag{5}$$

where $\odot$ denotes the element-wise Hadamard product between two matrices. $\mathbf{e}$ is a vector of all 1s of length $(M_X + M_W - 1)(N_X + N_W - 1)$. The JK matrices $\mathbf{U}_j^k$ ($j = 1, \ldots, J$ and $k = 1, \ldots, K$) are the weights of the connections between the hidden and output neurons. The next section presents the back-propagation training algorithm for a CNN.

3. BACK-PROPAGATION FOR CNN TRAINING

The back-propagation (BP) algorithm performs maximum likelihood (ML) estimation of the J convolution matrices $\mathbf{W}_1, \ldots, \mathbf{W}_J$ and the JK hidden-output weight matrices $\mathbf{U}_j^k$. Let $\mathbf{y}$ denote the 1-in-K encoding vector of the target label for a given input image $\mathbf{X}$. This means $y_k = 1$ when k corresponds to the correct class and 0 otherwise. BP computes the cross entropy between the soft-max activations of the output neurons and the target vector $\mathbf{y}$:

$$E(\Theta) = -\sum_{k_1=1}^{K} y_{k_1} \log(a_{k_1}^t) \tag{6}$$

where Θ denotes all the parameters of the CNN – the J convolution matrices $\mathbf{W}_1, \ldots, \mathbf{W}_J$ and the weight matrix $\mathbf{U}$. Note that $-E(\Theta)$ is the log-likelihood

$$L(\Theta) = \log(a_k^t) = -E(\Theta) \tag{7}$$

of the correct class label for the given input image. Hence the ML estimate of Θ is

$$\Theta^* = \arg\max_{\Theta} L(\Theta) \,. \tag{8}$$

BP performs gradient ascent on the log-likelihood surface $L(\Theta)$ to iteratively find the ML estimate of Θ. This also

holds when minimizing squared-error because BP is equivalent to ML estimation with a conditional Gaussian distribution [10, 26]. The estimate of Θ at the $(n+1)$-th iteration is

$$\Theta^{(n+1)} = \Theta^{(n)} - \eta \nabla_\Theta E(\Theta)\Big|_{\Theta=\Theta^{(n)}} \quad (9)$$

where η is a positive learning rate. A forward pass in BP computes the activations of all hidden and output neurons in the CNN. Back-propagating the output neuron activation errors through the network gives the gradient of the data log-likelihood function with respect to the CNN parameters. The gradient ascent in (9) updates these parameters.

The hidden neuron activations in a CNN are "latent" or unseen variables for the purposes of the EM algorithm. BP here performs ML estimation of a CNN's parameters. The EM algorithm is a popular iterative method for such ML estimation [11]. The EM algorithm uses the lower-bound Q of the log-likelihood function $L(\Theta)$:

$$Q(\Theta|\Theta^{(n)}) = \mathbb{E}_{p(\mathbf{Z}_1,\ldots,\mathbf{Z}_J|\mathbf{X},\mathbf{y},\Theta^{(n)})}\{\log p(\mathbf{Z}_1,\ldots,\mathbf{Z}_J,\mathbf{y}|\mathbf{X},\Theta)\} \quad (10)$$

The J matrices $\mathbf{Z}_1,\ldots,\mathbf{Z}_J$ are the latent variables in the algorithm's expectation (E) step. Then the Maximization (M) step maximizes the Q-function to find the next parameter estimate

$$\Theta^{(n+1)} = \arg\max_\Theta Q(\Theta|\Theta^{(n)}) \; . \quad (11)$$

The generalized EM (GEM) algorithm performs this optimization by stochastic gradient ascent. Theorem 1 from [10] shows that BP is a special case of the GEM algorithm.

Theorem 1. Backpropagation is a special case of the GEM Algorithm [10]
The backpropagation update equation for a differentiable likelihood function $p(y|\mathbf{x},\Theta)$ at epoch n

$$\Theta^{(n+1)} = \Theta^{(n)} + \eta \nabla_\Theta \log p(y|\mathbf{x},\Theta)\Big|_{\Theta=\Theta^{(n)}} \quad (12)$$

equals the GEM update equation at epoch n

$$\Theta^{(n+1)} = \Theta^{(n)} + \eta \nabla_\Theta Q(\Theta|\Theta^{(n)})\Big|_{\Theta=\Theta^{(n)}} \quad (13)$$

where GEM uses the differentiable Q-function $Q(\Theta|\Theta^{(n)})$ in (10).

This result lets us use the noisy EM algorithm to speed up BP training of a CNN. The next section details the noisy EM algorithm.

4. NOISY EXPECTATION-MAXIMIZATION (NEM) ALGORITHM

The Noisy Expectation-Maximization (NEM) algorithm [12, 13] provably speeds up the EM algorithm. It adds noise to the data at each EM iteration. The noise decays with the iteration count to ensure convergence to the optimal parameters of the original data model. The additive noise must also satisfy the NEM condition below that ensures that the NEM parameter estimates will climb faster up the likelihood surface on average.

4.1. NEM Theorem

The NEM Theorem [12, 13] states when noise speeds up the EM algorithm's convergence to a local optimum of the likelihood surface. The NEM Theorem uses the following notation. The noise random variable $\mathbf{N}$ has pdf $p(\mathbf{n}|\mathbf{x})$. So the noise $\mathbf{N}$ can depend on the data $\mathbf{x}$. Vector $\mathbf{h}$ denotes the latent variables in the model. $\{\Theta^{(n)}\}$ is a sequence of EM estimates for Θ. $\Theta_* = \lim_{n\to\infty} \Theta^{(n)}$ is the converged EM estimate for Θ. Define the noisy Q function $Q_{\mathbf{N}}(\Theta|\Theta^{(n)}) = \mathbb{E}_{\mathbf{h}|\mathbf{x},\Theta_k}[\ln p(\mathbf{x}+\mathbf{N},\mathbf{h}|\theta)]$. Assume that the differential entropy of all random variables is finite and that the additive noise keeps the data in the support of the likelihood function. Then we can state the general NEM theorem [12, 13].

Theorem 2. Noisy Expectation Maximization (NEM)
The EM estimation iteration noise benefit

$$Q(\Theta_*|\Theta_*) - Q(\Theta^{(n)}|\Theta_*) \geqslant Q(\Theta_*|\Theta_*) - Q_N(\Theta^{(n)}|\Theta_*) \quad (14)$$

or equivalently

$$Q_N(\Theta^{(n)}|\Theta_*) \geqslant Q(\Theta^{(n)}|\Theta_*) \quad (15)$$

holds on average if the following positivity *condition holds:*

$$\mathbb{E}_{\mathbf{x},\mathbf{h},\mathbf{N}|\Theta*}\left[\ln\left(\frac{p(\mathbf{x}+\mathbf{N},\mathbf{h}|\Theta_k)}{p(\mathbf{x},\mathbf{h}|\Theta_k)}\right)\right] \geqslant 0 \; . \quad (16)$$

The NEM Theorem states that each iteration of a properly noisy EM algorithm gives higher likelihood estimates on average than do the regular EM's estimates. So the NEM algorithm converges faster than EM for a given data model. The faster NEM convergence occurs both because the likelihood function has an upper bound and because the NEM algorithm takes larger average steps up the likelihood surface. NEM also speeds up the training of hidden Markov models [27] and the K-means clustering algorithm [28] used in big-data processing [29].

5. NOISY BACKPROPAGATION FOR CNN TRAINING

We add noise to the 1-in-K encoding vector $\mathbf{y}$ of the target class label. The next theorem states the noise-benefit sufficient condition for Gibbs activation output neurons used in CNN K-class classification.

Theorem 3. Forbidden Hyperplane Noise-Benefit Condition for CNN

The NEM positivity condition holds for ML training of a CNN with Gibbs activation output neurons if

$$\mathbb{E}_{\mathbf{y},\mathbf{Z}_1,\ldots,\mathbf{Z}_J,\mathbf{n}|\mathbf{X},\Theta*}\left\{\mathbf{n}^T\log(\mathbf{a}^t)\right\} \geqslant 0 \tag{17}$$

where the activation of the k-th output neuron is

$$a_k^t = \frac{\exp\left(\sum_{j=1}^J \mathbf{e}^T\mathbf{Z}_j \odot \mathbf{U}_j^k\mathbf{e}\right)}{\sum_{k_1=1}^K \exp\left(\sum_{j=1}^J \mathbf{e}^T\mathbf{Z}_j \odot \mathbf{U}_j^{k_1}\mathbf{e}\right)} \tag{18}$$

where $\odot$ denotes the element-wise Hadamard product between two matrices. $\mathbf{e}$ is a vector of all 1s of length $(M_X + M_W - 1)(N_X + N_W - 1)$.

Proof. Add noise to the target 1-in-K encoding vector $\mathbf{y}$ at the output neurons. Then the likelihood ratio in the NEM sufficient condition becomes

$$\frac{p(\mathbf{y}+\mathbf{n},\mathbf{Z}_1,\ldots,\mathbf{Z}_J|\mathbf{X},\Theta)}{p(\mathbf{y},\mathbf{Z}_1,\ldots,\mathbf{Z}_J|\mathbf{X},\Theta)} = \frac{p(\mathbf{y}+\mathbf{n}|\mathbf{Z}_1,\ldots,\mathbf{Z}_J,\Theta)}{p(\mathbf{y}|\mathbf{Z}_1,\ldots,\mathbf{Z}_J,\Theta)} \ . \tag{19}$$

The output soft-max activations $\mathbf{a}_k^t$ from (5) simplify the ratio on the right-hand side of the above equation. This gives

$$\frac{p(\mathbf{y}+\mathbf{n}|\mathbf{Z}_1,\ldots,\mathbf{Z}_J,\Theta)}{p(\mathbf{y}|\mathbf{Z}_1,\ldots,\mathbf{Z}_J,\Theta)} = \prod_{k=1}^K \frac{(a_k^t)^{t_k+n_k}}{(a_k^t)^{t_k}} = \prod_{k=1}^K (a_k^t)^{n_k} \ . \tag{20}$$

Substituting the above equation in (16) gives

$$\mathbb{E}_{\mathbf{y},\mathbf{Z}_1,\ldots,\mathbf{Z}_J,\mathbf{n}|\mathbf{X},\Theta*}\left\{\log\left(\prod_{k=1}^K (a_k^t)^{n_k}\right)\right\} \geqslant 0 \tag{21}$$

or

$$\mathbb{E}_{\mathbf{y},\mathbf{Z}_1,\ldots,\mathbf{Z}_J,\mathbf{n}|\mathbf{X},\Theta*}\left\{\sum_{k=1}^K n_k\log(a_k^t)\right\} \geqslant 0 \ . \tag{22}$$

□

Figure 2 illustrates the sufficient condition in (17) for a CNN with three output neurons. All noise $\mathbf{n}$ above the hyperplane $\{\mathbf{n} : \mathbf{n}^T\log(\mathbf{a}^t) = 0\}$ speeds CNN training on average.

6. SIMULATION RESULTS

All simulations used the MNIST data set of handwritten digits. The MNIST data set contains 28×28 gray-scale pixel images with pixel intensities between 0 and 1. Figure 6 shows 20 sample images from this data set. Figure 7 shows a schematic diagram of the BP training of a CNN using images from the MNIST data set. The simulations used at least 1000 images from the MNIST training-set. We modified an open-source Matlab toolbox [30] to add noise during CNN training. The CNN contained one convolution layer with three 3×3 pixel masks each. We followed the convolution layer with factor-2 down-sampling to increase system robustness and to reduce the number of CNN parameters [2].

Data: T input images $\{\mathbf{X}_1,\ldots,\mathbf{X}_T\}$, T target label 1-in-K vectors $\{\mathbf{y}_1,\ldots,\mathbf{y}_T\}$, number J of convolution masks, size $M_W \times N_W$ of each convolution mask, number of BP epochs R
Result: Trained CNN weight matrices
while *epoch* $r : 1 \to R$ **do**
  while *training image number* $t : 1 \to T$ **do**
    • Compute the J hidden activation matrices $\mathbf{Z}_1,\ldots,\mathbf{Z}_J$ using (2) and (3);
    • Downsample the J hidden activation matrices $\mathbf{Z}_1,\ldots,\mathbf{Z}_J$ by a factor of 2. • Compute the K-D output soft-max activation vector $\mathbf{a}$ using (5);
    • Generate noise vector $\mathbf{n}$;
    if $\mathbf{n}^T\log(\mathbf{a}) \geqslant 0$ **then**
      • Add NEM noise: $\mathbf{y}_t \leftarrow \mathbf{y}_t + \mathbf{n}$;
    else
      • Do nothing
    end
    • Compute error $\mathbf{y}_t - \mathbf{a}$;
    • Back-propagate error to compute cross entropy gradient $\nabla_\Theta E(\Theta)$;
    • Update network parameters Θ using gradient descent in (9);
    .
  end
end

Algorithm 1: The NEM-BP Algorithm for a CNN.

The output layer neurons used the soft-max or Gibbs activation function for 10-way classification. All hidden neurons used the logistic sigmoid function. We used uniform noise over $(-0.5\sqrt{c/t^d}, 0.5\sqrt{c/t^d})$ where $c = 0, 0.2, \ldots, 3$, $d = 1, 2, \ldots, 5$, and t is the training epoch. The noise variance thus decreased to 0 as training epochs proceed. Figure 3 shows the training-set cross entropy of a CNN for three algorithms: standard noiseless BP, BP with blind noise (Blind-BP), and BP with NEM noise (NEM-BP). We obtain a 39.26% average reduction in training-set cross entropy over the first 15 iterations using NEM-BP compared with the noiseless BP. Figure 4 plots the training-set classification error rates as the CNN learns. NEM-BP gives a 47.43% reduction in training-set error rate averaged over the first 15 iterations as compared with noiseless BP. Adding blind noise (Blind-BP) gave only a minor improvement of 4.05%.

We next plot the relative average reduction in cross entropy for NEM-BP as the noise scale c varies from 0 to 3 in

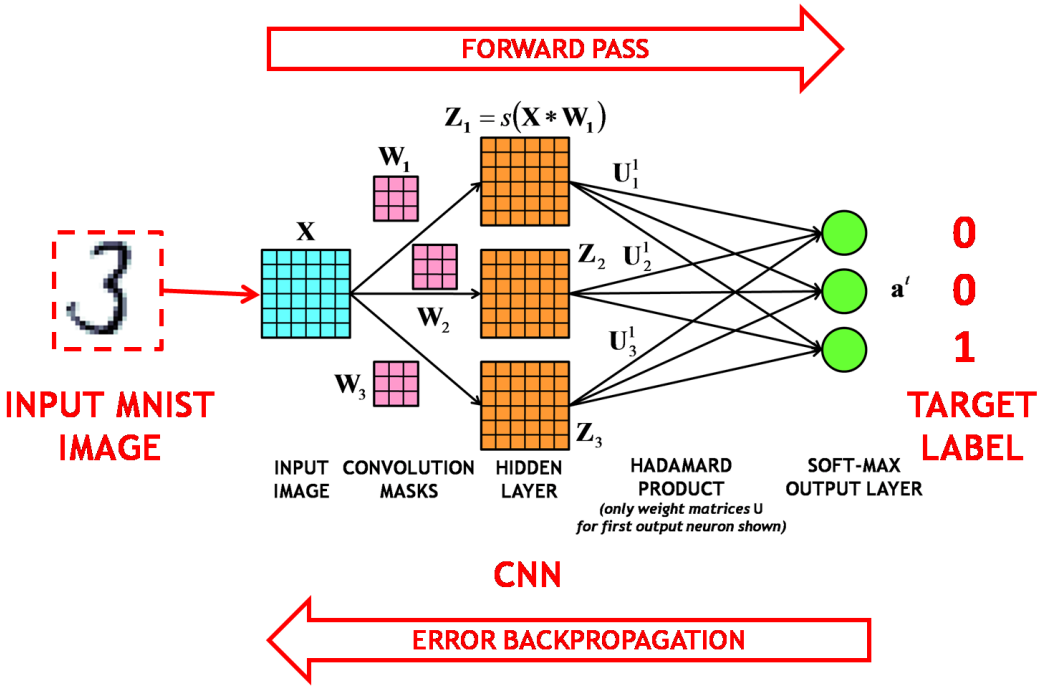


Fig. 7. CNN Training on MNIST: The figure shows a schematic diagram for training a CNN using an image from the MNIST data set. A forward pass through the CNN computes the hidden and output neuron activations. The error between the output activation vector $\mathbf{a}^t$ and the true target vector $(0, 0, 1)$ then propagates back through the CNN to compute the gradient of cross entropy. Then gradient descent updates the weights of the CNN using this gradient.

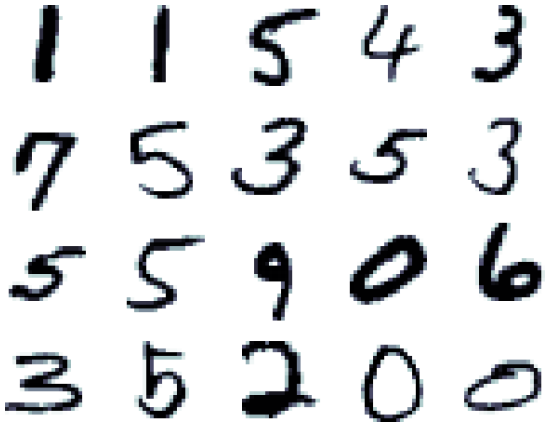


Fig. 6. MNIST Digits: The figure shows 20 sample images from the MNIST data set. Each digit is a 28×28 pixel grayscale image.

steps of 0.2. Figure 5 shows the resulting characteristic noise-benefit inverted U-curve. The optimal uniform noise scale occurs at $c^* = 1$ and NEM-BP gives a 39.26% improvement in average cross entropy. NEM noise hurts CNN training when the noise scale increases beyond 2.6.

We also explored how the training-data set size affects NEM performance. We varied the MNIST training-set size over $1000, 2000, \ldots, 5000$ and computed the relative average reduction in training cross entropy for NEM-BP using the optimal noise variance. Figure 8 shows the resulting decreasing bar chart: NEM-BP's performance falls as the number of training data samples increases. This shows that NEM-BP is especially useful when the number of training data samples is small relative to the number of estimated CNN parameters.

7. CONCLUSIONS

Proper noise injection speeds up the back-propagation (BP) training of a convolutional neural network (CNN). This follows because the BP algorithm is a special case of the EM algorithm and because the recent noisy EM (NEM) theorem gives a sufficient condition for speeding up the EM algorithm using noise. NEM noise injection experiments on the MNIST data set show substantial reduction in training-set cross entropy and classification error rate as compared with the noiseless BP algorithm. Blind noise gave at best a small noise ben-

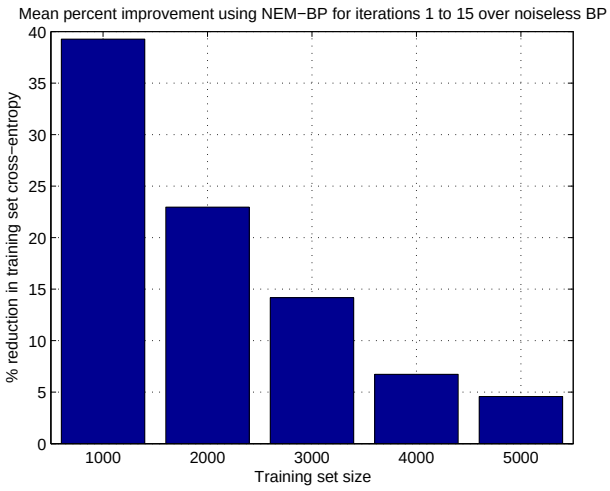


Fig. 8. Variation of NEM-BP performance benefit with increasing training-set size: The bar chart shows the relative average reduction in training-set cross entropy for NEM-BP as the training-set size increases. The noise benefit is greater for smaller training-data set sizes.

efit. Simulations show that the NEM noise benefit was largest for smaller data sets. Future work will explore adding noise to both the input data and hidden neurons.

8. REFERENCES

[1] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[2] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[3] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Handwritten digit recognition with a back-propagation network," in *Proc. NIPS*, 1990.

[4] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *NIPS*, 2012, vol. 1, p. 4.

[5] S. Lawrence, C. L. Giles, A. C. Tsoi, and A. D. Back, "Face recognition: A convolutional neural-network approach," *IEEE Transactions on Neural Networks*, vol. 8, no. 1, pp. 98–113, 1997.

[6] P. Simard, D. Steinkraus, and J. C. Platt, "Best practices for convolutional neural networks applied to visual document analysis.," in *ICDAR*, 2003, vol. 3, pp. 958–962.

[7] M. Szarvas, A. Yoshizawa, M. Yamamoto, and J. Ogata, "Pedestrian detection with convolutional neural networks," in *Proceedings of the IEEE Intelligent Vehicles Symposium*. IEEE, 2005, pp. 224–229.

[8] D. C. Cireşan, U. Meier, J. Masci, L. M. Gambardella, and J. Schmidhuber, "Flexible, high performance convolutional neural networks for image classification," in *Proceedings of the Twenty-Second international joint conference on Artificial Intelligence*. AAAI Press, 2011, vol. 2, pp. 1237–1242.

[9] M. Matsugu, K. Mori, Y. Mitari, and Y. Kaneda, "Subject independent facial expression recognition with robust face detection using a convolutional neural network," *Neural Networks*, vol. 16, no. 5, pp. 555–559, 2003.

[10] K. Audhkhasi, O. Osoba, and B. Kosko, "Noise benefits in backpropagation and deep bidirectional pre-training," in *Proc. IJCNN*, 2013, pp. 1–8.

[11] Arthur P Dempster, Nan M Laird, and Donald B Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 1–38, 1977.

[12] O. Osoba, S. Mitaim, and B. Kosko, "Noise Benefits in the Expectation-Maximization Algorithm: NEM theorems and Models," in *The International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2011, pp. 3178–3183.

[13] O. Osoba, S. Mitaim, and B. Kosko, "The noisy expectation–maximization algorithm," *Fluctuation and Noise Letters*, vol. 12, no. 03, 2013.

[14] B. Kosko, *Noise*, Viking, 2006.

[15] A. Patel and B. Kosko, "Levy Noise Benefits in Neural Signal Detection," in *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, 2007, vol. 3, pp. III–1413 –III–1416.

[16] A. Patel and B. Kosko, "Stochastic Resonance in Continuous and Spiking Neurons with Levy Noise," *IEEE Transactions on Neural Networks*, vol. 19, no. 12, pp. 1993–2008, December 2008.

[17] M. Wilde and B. Kosko, "Quantum forbidden-interval theorems for stochastic resonance," *Journal of Physical A: Mathematical Theory*, vol. 42, no. 46, 2009.

[18] A. Patel and B. Kosko, "Error-probability noise benefits in threshold neural signal detection," *Neural Networks*, vol. 22, no. 5, pp. 697–706, 2009.

[19] A. Patel and B. Kosko, "Optimal Mean-Square Noise Benefits in Quantizer-Array Linear Estimation," *IEEE Signal Processing Letters*, vol. 17, no. 12, pp. 1005 – 1009, Dec. 2010.

[20] A. Patel and B. Kosko, "Noise Benefits in Quantizer-Array Correlation Detection and Watermark Decoding," *IEEE Transactions on Signal Processing*, vol. 59, no. 2, pp. 488 –505, Feb. 2011.

[21] B. Franzke and B. Kosko, "Noise Can Speed Convergence in Markov Chains," *Physical Review E*, vol. 84, no. 4, pp. 041112, 2011.

[22] A. R. Bulsara and L. Gammaitoni, "Tuning in to Noise," *Physics Today*, pp. 39–45, March 1996.

[23] L. Gammaitoni, "Stochastic Resonance in Multi-Threshold Systems," *Physics Letters A*, vol. 208, pp. 315–322, December 1995.

[24] D. H. Hubel and T. N. Wiesel, "Receptive fields of single neurones in the cat's striate cortex," *The Journal of physiology*, vol. 148, no. 3, pp. 574–591, 1959.

[25] K. Fukushima, "Neocognitron: A hierarchical neural network capable of visual pattern recognition," *Neural networks*, vol. 1, no. 2, pp. 119–130, 1988.

[26] C. M. Bishop, *Pattern recognition and machine learning*, vol. 1, Springer, 2006.

[27] K. Audhkhasi, O. Osoba, and B. Kosko, "Noisy hidden Markov models for speech recognition," in *Proc. IJCNN*, 2013, pp. 1–8.

[28] O. Osoba and B. Kosko, "Noise-Enhanced Clustering and Competitive Learning Algorithms," *Neural Networks*, Jan. 2013.

[29] A. K. Jain, "Data clustering: 50 years beyond K-means," *Pattern Recognition Letters*, vol. 31, no. 8, pp. 651–666, 2010.

[30] Rasmus Berg Palm, "DeepLearn Toolbox," `https://github.com/rasmusbergpalm/DeepLearnToolbox`.

On Walsh-Fourier Bispectral Analysis

M. M. Gabr[1], M. El-Hashash[2], and G. S. Sabbah[2]
Department of Mathematics and Computer Science, Faculty of Science, Alexandria University, Egypt
Department of Mathematics, Bridgewater State University, Bridgewater, MA 02325, USA
Department of Mathematics and Computer Science, Faculty of Science, Alexandria University, Egypt

Abstract

In this paper we define the Walsh-Fourier bispectral density and establish a statistical methodology for bispectral analysis of integer valued time series. The theoretical properties and results pertaining to the definition and estimation of the Walsh-Fourier bispectral density are obtained. Many of the results we obtain have their analogues in the conventional Fourier frequency domain. We present simulation results of Walsh Fourier bispectral density for linear and bilinear integer valued time series models.

Keywords: Time Series, Spectral Analysis, Bilinear Models, Walsh-Fourier Analysis, Bispectral Density

1. Introduction

The orthogonal system of sine and cosine functions plays an important role in the analysis of time series. In communications engineering, especially in the theory of linear, time invariant networks, astronomy, and many other fields, is considered to be a very important mathematical tool. The second and third order Fourier spectral density functions of a stationary time series $\{X_t, t\varepsilon Z\}$ are defined as

$$g(\omega) = \frac{1}{2\pi}\sum_{-\infty}^{\infty}\gamma(s)\, e^{-i\omega s} \quad (1)$$

and

$$g(\omega_1,\omega_2) = \frac{1}{(2\pi)^2}\sum_{-\infty}^{\infty}\sum_{-\infty}^{\infty}C(s_1,s_2)e^{-i(\omega_1 s_1+\omega_2 s_2)} \quad (2)$$

where the second and third order autocovariance functions

$$\gamma(s) = E[(X_t-\mu)(X_{t+s}-\mu)] \quad (3)$$

and

$$C(s_1,s_2) = E[(X_t-\mu)(X_{t+s_1}-\mu)(X_{t+s_2}-\mu)] \quad (4)$$

are absolutely summable, i.e. satisfying the conditions

$$\sum_{-\infty}^{\infty}|j||\gamma(j)|<\infty \text{ and}$$

$$\sum_{-\infty}^{\infty}|j_r||C(j_1,j_2)|<\infty \ , r=1,2 \quad (5)$$

The lag or spectral window estimators of these functions are well established; appropriate references are Priestley (1981), Brillinger (1981), Subba Rao and Gabr (1984) and Brockwell and Davis (1991)).There are, however, many physical situations in which time series take values in a discrete (and possibly finite) set, so that it makes little statistical sense to correlate the data with sines and cosines.

A natural alternative to the usual trigonometric Fourier analysis is the Walsh-Fourier (WF) transform. This approach would enable investigators to analyze discrete (or integer)-valued time series (which we may think of as square wave forms) in terms of square waves and "sequency" (see e.g. Kohn (1980a,b) and Morettin (1981)) rather than sine waves and frequency. The Walsh functions are defined as products of the Rademacher functions. Walsh (1923), and others have developed a theory of Walsh-Fourier series and most of the results are parallel to those of classical trigonometric series theory. The original definition of Walsh was a recursive one and the functions were ordered by the number of sign changes in the interval [0,1). See Ahmed and Rao (1975), Beauchamp (1975) and Harmuth (1977) for further details.

There are two modes of developing Walsh spectral analysis in literature. The first mode is termed "*Walsh spectral analysis*" and is developed via the concept of *dyadic stationarity*. This mode is based on processes $\{\mathbf{X_t}, t=0, 1, 2, ...\}$ for which $\text{cov}(X_t, X_{t\oplus s}) = B(s)$ is a function only of the dyadic distance between t and $t\oplus s$ (cf. Morettin, 1974, 1981, for definitions, discussions and references). The other mode of development is termed "*Walsh-Fourier spectral analysis*" and is based on real-time stationarity. Kohn (1980a,b) laid the groundwork by showing that many of the results concerning the decomposition of stationary time series using trigonometric functions have their Walsh function analogs. Estimation of the Walsh-Fourier spectrum is established and discussed by Kohn (1980b), Stoffer (1987, 1990, 1991), and Ferryanto (1995).

The Walsh functions $\mathbf{\{W(n,\lambda), n = 0, 1, 2, ... ; 0\le\lambda\le1\}}$ form a complete orthonormal sequence on (0,1) and take on only two values +1 and -1. They are ordered by the number of zero crossing which is called *sequency*. Thus if $W(n, \lambda)$ is the n^{th} sequency-ordered Walsh functions, then W(n,.) makes n zero-crossing in [0,1). Two important properties are

$$\mathbf{W(n, \lambda_1\oplus\lambda_2)=W(n,\lambda_1)\ W(n,\lambda_2)} \quad \text{and}$$
$$\mathbf{W(n_1\oplus n_2,\lambda)=W(n_1,\lambda)\ W(n_2,\lambda)} \qquad \mathbf{(6)}$$

Here, $\oplus$ denotes modulo 2 or dyadic addition defined as,

$$\mathbf{m \oplus n = \sum_{j=0}^{k} |m_j - n_j| 2^j}$$

for $\mathbf{m = \sum_{j=0}^{k} m_j 2^j}$ and $\mathbf{n = \sum_{j=0}^{k} n_j 2^j}$

where $\mathbf{m_j}$ and $\mathbf{n_j}$ are either 1 or 0, so that $0\oplus0 = 1\oplus1 = 0$, $0\oplus1 = 1\oplus0 = 1$.

Let $\mathbf{X_0}, \mathbf{X_1}, ..., \mathbf{X_{N-1}}$ be a realization of length $N = 2^\nu$, $\nu > 0$ integer, from a stationary time series $\{X_t\}$ with zero mean and absolutely summable covariance function, then the logical covariance and the WF spectral density of $\{\mathbf{X_t}\}$ (see Kohn (1980a)) are defined to be respectively,

$$\mathbf{\tau(j) = \frac{1}{N}\sum_{k=0}^{N-1}\gamma(j\oplus k - k)}$$

and

$$\mathbf{f(\lambda) = \sum_{j=0}^{\infty}\tau(j)\ W(j,\lambda)} \qquad \mathbf{(7)}$$

Estimation of the WF spectrum is studied by Kohn (1980b) and Stoffer (1987, 1991).

In this paper, we introduce a WF bispectral density and establish its estimate.

2. WF Bispectral Density

Let $\mathbf{X_0}, \mathbf{X_1}, ..., \mathbf{X_{N-1}}$ be a realization of length $\mathbf{N=2^p}$, with a positive integer, from a discrete-time, zero-mean, stationary time series $\{X_t, t \varepsilon Z\}$ with absolutely summable covariance function, and let

$$\mathbf{d_N(\lambda) = \frac{1}{\sqrt{N}}\sum_{j=0}^{N-1} X_j W(j,\lambda)} \qquad \mathbf{(8)}$$

be the discrete WF transform of the data. For $\mathbf{\lambda_1\oplus\lambda_2\oplus\lambda_3 = 0}$, $\mathbf{0 \le \lambda_i < 1,\ i = 1, 2, 3}$ which implies that $\mathbf{\lambda_1\oplus\lambda_2 = \lambda_3}$ and using (7) and (6), we have

$$\mathbf{E\left[\sqrt{N}\, d_N(\lambda_1) d_N(\lambda_2)\ d_N(\lambda_1\oplus\lambda_2)\right]}$$

$$= \frac{1}{N}\sum_{j_1=0}^{N-1}\sum_{j_2=0}^{N-1}\sum_{j_3=0}^{N-1} E\left[X_{j_1} X_{j_2} X_{j_3}\right] W(j_1,\lambda_1)\, W(j_2,\lambda_2)\, W(j_3,\lambda_1\oplus\lambda_2)$$

$$= \frac{1}{N}\sum_{j_1=0}^{N-1}\sum_{j_2=0}^{N-1}\sum_{j_3=0}^{N-1} C(j_1 - j_3, j_2 - j_3) W(j_1,\lambda_1) W(j_2,\lambda_2) W(j_3,\lambda_1)\, W(j_3,\lambda_2)$$

$$= \frac{1}{N^{3/2}}\sum_{j_1=0}^{N-1}\sum_{j_2=0}^{N-1}\sum_{j_3=0}^{N-1} C(j_1 - j_3, j_2 - j_3) W(j_1\oplus j_3,\lambda_1)\, W(j_2\oplus j_3,\lambda_2)$$

$$= \frac{1}{N}\sum_{j_1=0}^{N-1}\sum_{j_2=0}^{N-1}\sum_{k=0}^{N-1} C(j_1\oplus k - k, j_2\oplus k - k) W(j_1,\lambda_1) W(j_2,\lambda_2)$$

Put $\mathbf{\tau(0,0) = C(0,0)}$, and

$$\mathbf{\tau(j_1, j_2) = \frac{1}{N}\sum_{k=0}^{N-1} C(j_1\oplus k - k, j_2\oplus k - k)} \qquad \mathbf{(9)}$$

where $\mathbf{\tau(j_1, j_2)}$ is the 3rd order logical autocovariance function. Thus,

$$\mathbf{E\left[\sqrt{N}\, d_N(\lambda_1) d_N(\lambda_2) d_N(\lambda_1\oplus\lambda_2)\right]} = \mathbf{\sum_{j_1=0}^{N-1}\sum_{j_2=0}^{N-1}\tau(j_1, j_2)\, W(j_1,\lambda_1)\, W(j_2,\lambda_2)} \qquad \mathbf{(10)}$$

and therefore $\lim_{N\to\infty} E[\sqrt{N} d_N(\lambda_1) d_N(\lambda_2) d_N(\lambda_1 \oplus \lambda_2)] = f(\lambda_1, \lambda_2)$

where

$$f(\lambda_1, \lambda_2) = \sum_{j_1=0}^{N-1} \sum_{j_2=0}^{N-1} \tau(j_1, j_2) W(j_1, \lambda_1) W(j_2, \lambda_2) \quad (11)$$

By analogy to the Fourier analysis, we will call $f(\lambda_1, \lambda_2)$ the Walsh -Fourier bispectral density function.

Lemma 2.1

If assumption (5) is satisfied, then the 3rd order logical autocovariance function $\tau(j_1, j_2)$ is absolutely summable and the Walsh-Fourier bispectral density function $f(\lambda_1, \lambda_2)$ exits.

Proof

$$|\tau(j_1, j_2)| \le \frac{1}{N} \sum_{k=0}^{N-1} |C(j_1 \oplus k - k, j_2 \oplus k - k)|$$

Hence, from the properties of the dyadic addition

$$\sum_{j_1, j_2=0}^{N-1} |\tau(j_1, j_2)| \le \frac{1}{N} \sum_{j_1, j_2, k=0}^{N-1} |C(j_1 \oplus k - k, j_2 \oplus k - k)|$$

$$= \frac{1}{N} \sum_{j_1, j_2, k=0}^{N-1} |C(j_1 - k, j_2 - k)|$$

$$= \sum_{j_1, j_2=-N+1}^{N-1} \left(1 - \frac{\max(|j_1|, |j_2|)}{N}\right) |C(j_1 - k, j_2 - k)|$$

$$\le \sum_{j_1, j_2=N-1}^{N-1} |C(j_1 - k, j_2 - k)|$$

and the result follows.

Now, We generalize the concept of the logical third order autocovariance function as follows. If cumulants of all orders for $\{X_t\}$ exist, then the logical joint cumulant of order k of $\{X_t\}$, with the same argument as in (9), is defined as

$$\tau_k(j_1, \ldots, j_{k-1}) = \frac{1}{N} \sum_{r=0}^{N-1} C_k(j_1 \oplus r - r, j_2 \oplus r - r, \ldots, j_k \oplus r - r) \quad (12)$$

where $0 \le j_1, \ldots, j_{k-1} < N-1$, $N = 2^\nu$, $\nu > 0$ integer and $C_k(.)$ is the joint cumulant of order k of $\{X_{j_1}, X_{j_2}, \ldots, X_{j_k}\}$. We define the k^{th} order Walsh-Fourier cumulant spectral density of $\{X_t\}$ as

$$f_k(\lambda_1, \ldots, \lambda_{k-1}) = \sum_{j_1, \ldots, j_{k-1}=0}^{\infty} \tau_k(j_1, \ldots, j_{k-1}) W(j_1, \lambda_1) \ldots W(j_{k-1}, \lambda_{k-1}) \quad (13)$$

Where both $\tau_k(j_1, \ldots, j_{k-1})$ and $f_k(\lambda_1, \ldots, \lambda_{k-1})$ exist if the following assumption is satisfied

$$\sum_{-\infty}^{\infty} |C_k(j_1, \ldots, j_{k-1})| < \infty \quad (14)$$

3. Estimation of the Walsh-Fourier Bispectral Density

Let $X_0, X_1, \ldots, X_{N-1}$ be a realization of length $N = 2^\nu$, $\nu > 0$ integer, from a stationary zero mean time series $\{X_t\}$, a natural estimator of $\tau(j_1, j_2)$ is given by

$$\hat{\tau}(j_1, j_2) = \frac{1}{N} \sum_{r=0}^{N-1} X_r X_{r \oplus j_1} X_{r \oplus j_2} \quad (15)$$

By analogy to the definition of the Fourier periodogram, we define the WF third order periodogram of $\{X_t\}$, using (8), as

$$I_N(\lambda_1, \lambda_2) = \sqrt{N} d_N(\lambda_1) d_N(\lambda_2) d_N(\lambda_1 \oplus \lambda_2)$$

$$= \frac{1}{N} \sum_{j_1, j_2, j_3=0}^{N-1} X_{j_1} X_{j_2} X_{j_3} W(j_1, \lambda_1) W(j_2, \lambda_2) W(j_3, \lambda_1 \oplus \lambda_2)$$

$$= \frac{1}{N} \sum_{j_1, j_2, j_3=0}^{N-1} X_{j_1} X_{j_2} X_{j_3} W(j_1 \oplus j_3, \lambda_1) W(j_2 \oplus j_3, \lambda_2)$$

$$= \frac{1}{N} \sum_{j_1, j_2, r=0}^{N-1} X_{j_1 \oplus r} X_{j_2 \oplus r} X_r W(j_1, \lambda_1) W(j_2, \lambda_2)$$

$$= \sum_{j_1, j_2 = 0}^{N-1} \hat{\tau}(j_1, j_2)\, W(j_1, \lambda_1)\, W(j_2, \lambda_2) \qquad (16)$$

We call (16) a 2-dimensional Walsh transform of the logical third order autocovariance function estimate. Now, taking the two dimensional Walsh transform of (16), we obtain

$$\frac{1}{N^2} \sum_{n_1, n_2 = 0}^{N-1} I_N\left(\frac{n_1}{N}, \frac{n_2}{N}\right) W\left(n_1; \frac{i_1}{N}\right) W\left(n_2; \frac{i_2}{N}\right)$$

$$= \frac{1}{N^2} \sum_{n_1, n_2 = 0}^{N-1} I_N\left(\frac{n_1}{N}, \frac{n_2}{N}\right) W\left(i_1; \frac{n_1}{N}\right) W\left(i_2; \frac{n_2}{N}\right)$$

$$= \frac{1}{N^2} \sum_{n_1, n_2, j_1, j_2 = 0}^{N-1} \hat{\tau}(j_1, j_2)\, W\left(i_1; \frac{n_1}{N}\right) W\left(i_2; \frac{n_2}{N}\right) W\left(j_1; \frac{n_1}{N}\right) W\left(j_2; \frac{n_2}{N}\right)$$

$$= \sum_{j_1, j_2 = 0}^{N-1} \hat{\tau}(j_1, j_2) \left[\frac{1}{N} \sum_{n_1 = 0}^{N-1} W\left(i_1 \oplus_1; \frac{N_1}{N}\right)\right] \left[\frac{1}{N} \sum_{n_2 = 0}^{N-1} W\left(i_2 \oplus_2; \frac{n_2}{N}\right)\right]$$

$$= \hat{\tau}(i_1, i_2) \qquad (17)$$

from the properties of the WT (see Kohn (1980a)),

$$\frac{1}{N} \sum_{n=0}^{N-1} W\left(r; \frac{n}{N}\right) = \begin{cases} 0 & r \neq 0 \\ 1 & r = 0 \end{cases}$$

and $i_1 \oplus j_1 = 0$ when $j_1 = i_1$. Therefore, if N is large, then the fastest way to compute $\hat{\tau}(j_1, j_2)$'s is to first compute WFT (7), and hence the third order periodogram $I_N(n_1/N, n_2/N)$, then again use the two dimensional WFT to compute the left side of (17).

Definition 3.1

Analogous to Kohn (1980a), we define the discrete dyadic rational sequency $\lambda_{j,N} = (\lambda_{j_1,N}, \lambda_{j_2,N})$ $\lambda_{j_i,N} = \frac{j_i}{N}$ for $1 \le j_i \le N-1$, $N = 2^\nu \nu$ is a positive integer, $i = 1, 2$. If the collection $\{\lambda_{j_i(\nu_i),N}; \nu_i = 1, \ldots, L_i\}$ is close to λ_i, such that $\lambda_i \oplus \lambda_{j_i(\nu_i),N} \to 0$ as $N \to \infty$ and $|\lambda_{j_i(\nu_i),N} - \lambda_{j_i(m_i),N}| \ge N^{-1}$ for $\nu_i \neq m_i = 1, \ldots, L_i$, i=1, 2. Then, we call the pair collection $\{\lambda_{j(\nu),N} = (\lambda_{j_1(\nu_1),N}, \lambda_{j_2(\nu_2),N}); \nu_1 = 1, \ldots, L_1 \text{ and } \nu_2 = 1, \ldots, L_2\}$ is close to $\lambda = (\lambda_1, \lambda_2)$.

.

Definition 3.2

If λ can be written in the finite form $\lambda = \sum_{j=1}^{m} \lambda_j 2^{-j}$, then it is called a *dyadic rational*, otherwise, it is *dyadically irrational*. If λ can be written both in the finite and also in infinite sum, then we choose the finite representation. Similarly, (λ_1, λ_2) is called a dyadic rational if both λ_1 and λ_2 are dyadic rational.

It can be shown that $I(\lambda_1, \lambda_2)$ is asymptotically unbiased estimate of $f(\lambda_1, \lambda_2)$ but it is not a consistent estimate of $f(\lambda_1, \lambda_2)$. To obtain a consistent estimator, $I(\lambda_1, \lambda_2)$ has to be "smoothed", as in the Fourier bispectrum case (see Section 2.9 in Brillinger and Rosenblatt (1967a) page 164). Smoothing the series $\{I_N(\lambda_{j_1(m_1),N}, \lambda_{j_2(m_2),N})\}$ with the aid of the collection $\{(\lambda_{j_1(m_1),N}, \lambda_{j_2(m_2),N}); m_i = 1, \ldots, M, i = 1, 2\}$ leads us to consider the class of estimators having the form

$$\hat{f}_{M,N}(\lambda_1^{(\ell_1)}, \lambda_2^{(\ell_2)}) =$$

$$= \sum_{|m_1| \le \frac{1}{2}(M-1)} \sum_{|m_2| \le \frac{1}{2}(M-1)} H_N(m_1, m_2)\, I(\lambda_{j_1(\ell_1 \oplus m_1),N}, \lambda_{j_2(\ell_2 \oplus m_2),N}) \qquad (18)$$

where $\{H_N(.,.)\}$ is a sequence of weight function (two dimensional spectral window) and M must depend on N. In order to this estimate of bispectral density to be consistent, we impose the following assumption on M and $\{H_N(.,.)\}$.

Assumption 3.1

(i) $M = 2^s$ and $N = 2^\upsilon$ with **s** < **υ**, such that $M \to \infty$ and $\frac{M^2}{N} \to 0$ as $N \to \infty$

(ii) $H_N(m_1, m_2)$ is a non-negative, real valued function such that

$$\sum_{|m_1| \le \frac{1}{2}(M-1)} \sum_{|m_2| \le \frac{1}{2}(M-1)} H_N(m_1, m_2) = 1$$

but with $H_N(m_1, m_2) = 0(M^{-2})$ so that

$$\sum_{|m_1| \le \frac{1}{2}(M-1)} \sum_{|m_2| \le \frac{1}{2}(M-1)} H_N^2(m_1, m_2) \to 0 \quad \text{as} \quad N \to \infty$$

The dependence of M on N will be understood although they don't appear in the notation. Assumption 3.1 is a two dimensional generalization of the assumptions for the one dimensional case in the literature.

Now substituting on

$$\mathbf{I}(\lambda_{j_1(\ell_1 \oplus m_1),N}, \lambda_{j_2(\ell_2 \oplus m_2),N})$$

from Equation (16) into Equation (18) we obtain $\hat{f}_{M,N}(\lambda_1^{(\ell_1)}, \lambda_2^{(\ell_2)})$

$$= \sum_{j_1=0}^{M-1} \sum_{j_2=0}^{M-1} h\left(\frac{j_1}{M}, \frac{j_2}{M}\right) \hat{\tau}(j_1, j_2) W(j_1, \lambda_{j_1(\ell_1),N}) W(j_2, \lambda_{j_2(\ell_2),N}) \quad \textbf{(19)}$$

where $h(u_1, u_2)$ is a two dimensional real-valued function (lag window) as defined in one dimensional case.

Theorem 3.1

Let $\{X_t\}$ be a zero mean stationary time series satisfy Assumption (5) and let

$$\hat{f}_{M,N}(\lambda_1, \lambda_2) := \hat{f}_{M,N}(\lambda_1^{(\ell_1)}, \lambda_2^{(\ell_2)})$$

be defined in Equation (18) where $H_N(m_1, m_2)$ satisfies Assumption 3.1, then

(i) $E\{\hat{f}_{M,N}(\lambda_1, \lambda_2)\} = f(\lambda_1, \lambda_2) + o(1)$

(ii) $\mathbf{var}\{\hat{f}_{M,N}(\lambda_1, \lambda_2)\} = 8\Lambda_{M,N}^2 f^2(\lambda_1, \lambda_2) + 0(N^{-1})$

where

$$\Lambda_{M,N}^2 = \sum_{|m_1| \le \frac{1}{2}(M-1)} \sum_{|m_2| \le \frac{1}{2}(M-1)} H_N^2(m_1, m_2)$$

The proof of this theorem is an extension of the 2-dimentional case given by Kohn (1980 a, b) and analogous to the 3-dimensional case of the Fourier bispectrum given by Brillinger and Rosenblatt (1967). It is too long to be included here and will be published elsewhere.

4. Applications

For our illustration we consider two integer valued time series models. The first is the following integer valued autoregressive model of order 1 (INAR(1)), introduced by Al-Osh and Al-Zaid (1987)

$$X_t = \alpha \, o \, X_{t-1} + \varepsilon_t \quad \textbf{(20)}$$

where $\alpha \in [0,1]$, $\varepsilon_t\}$ is a sequence of uncorrelated non negative integer valued random variables having mean μ and finite variance σ^2 and the thinning operator "o" is defined by

$$\alpha o X_t = \sum_{i=1}^{X_t} Y_i$$

where Y_i is a sequence of i.i.d. random variables, independent of X_t, such that **Pr (Y_i =1) = 1– Pr (Y_i = 0) = α**

The seconed is the following integer valued bilinear model (INBL(1,0,1,1)), introduced by Doukhan et al. (2006)

$$X_t = a o X_{t-1} + b o (\varepsilon_{t-1} X_{t-1}) + \varepsilon_t \quad \textbf{(21)}$$

where $\{\varepsilon_t\}$ has a Poisson distribution with mean μ and the

sequences involved in the operators **a o** and **b o** have Poisson distributions with respective means **a** and **b**.

A set of 512 mutually independent random variables $\{\varepsilon_t\}$, each distributed as Poisson with $\lambda = 1$, is generated. A time series $\{X_t, t = 1, 2, ..., 512\}$ is generated from the INAR(1) model (20).The thinning variables Y_i's are chosen to be independent Bernoulli random variables with $\boldsymbol{\alpha} = 0.6$. The estimate of the Walsh-Fourier bispectral density function is calculated for the generated time series using Parzen window with truncation point M = 32 and shown in Figure 1. Another time series $\{X_t, t = 1, 2... 512\}$ is generated from the INBL(1,0,1,1) model (21). The thinning variables Y_i's are chosen to be independent Poisson random variables with parameter **a** = 0.3. The thinning variables Z_i are chosen to be independent Poisson random variables with parameter **b** = 0.5. The estimate of the Walsh-Fourier bispectral density function is calculated for the generated time series using the Parzen window with truncation point M = 32 and shown in Figure 2.

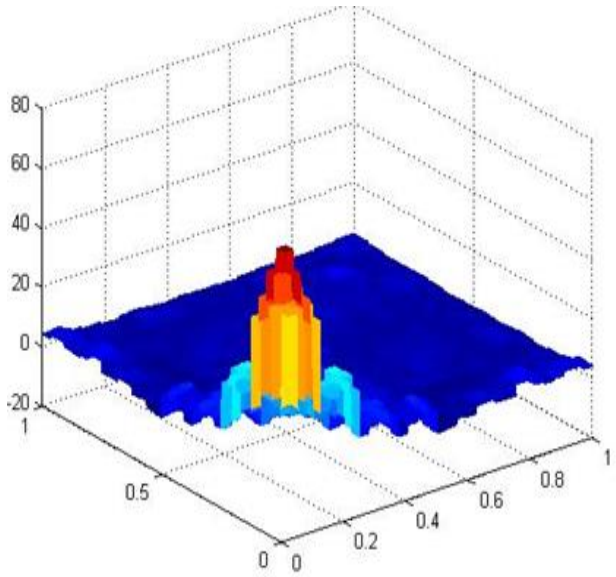


Figure 1: Walsh Bispectrum of INAR (1)
N = 512, M = 32, α = 0.6, λ = 1

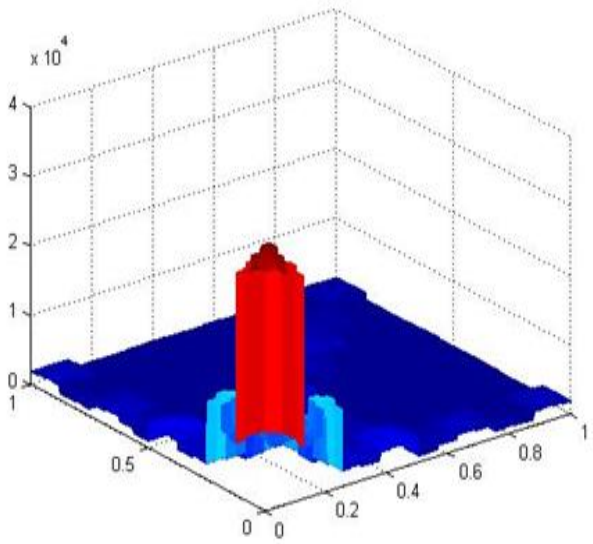


Figure 2: Walsh Fourier Bispectrum of INBL (1, 0, 1, 1)
N=512, M=32, a = 0.3, b=0.5, λ = 1

References

[1] Ahmed, N. and Rao, K R (1975) Orthogonal Transforms for Digital Signal Processing, New York: Springer-Verlag.

[2] Al-Osh, M.A. and Al-Zaid A.A. (1987) "First-order integer- valued autoregressive (INAR(1)) process". *Journal of Time Series Analysis, Vol. 8, No. 3, 261-275.* [2] Beauchamp, K G (1975) Walsh Functions and Their Applications, London: Academic Press.

[3] Brillinger, D R (1981) *Time Series: Data Analysis and Theory*, 2nd ed. San Francisco: Holden-Day.

[4] Brillinger, D. R. and Rosenblatt, M (1967). "Asymptotic Theory of k-th order Spectra in Spectral Analysis of Time Series". *Ed. B Harris, 153-188, Willy, New York.*

[5] Brockwell, P. J. and Davis, R. A. (1991) *Time Series: Theory and Methods*, 2nd Ed., New York, Springer.

[6] Doukhan, P., Latour, A. and Oraichi, D. (2006) " A simple integer-valued bilinear time series model". *Adv. Appl. Prob. 38, 559–578.*

[7] Ferryanto, S.G (1995) On Estimation of the Walsh-Fourier Spectral Density of Two Dimensional Strictly Homogeneous Random Field, *Nonparametric Statistics*, Vol. 5, pp. 391-407.

[8] Harmuth, H. (1977) Sequency Theory- Foundations and Applications, *Academic Press, New York*, 1977.

[9] Kohn, R. (1980a) On the spectral decomposition of stationary time series using Walsh functions I, *Adv. Appl. Prob.*, 12, 183-199.

[10] ——— (1980b) On the spectral decomposition of stationary time series using Walsh functions II, *Adv. Appl. Prob.*, 12, 462-272.

[11] Morettin, P.A. (1974) Walsh-function analysis of a certain class of time series, *Stoch. Processes and their applic.*, 2, 183-193.

[12] ——— (1981) Walsh spectral analysis, *SIAM Rev.*, V. 23, No. 3, pp 279-91.

[13] Priestley, M. B. (1981) "*Spectral Analysis and Time Series*", vol. 1, New York: Academic press.

[14] Stoffer, D.S. (1987) Walsh-Fourier analysis of discrete-valued time series, *J Time Series Anal.*, Vol. 8, No. 4, 449-467.

[15] ——— (1990) Multivariate Walsh-Fourier analysis, *J Time Series Anal.*, Vol. 11, 57-73.

[16] ——— (1991) Walsh-Fourier analysis and its applications (with discussion), *J Am. Stat. Ass.*, 86, 461-479.

[17] Subba Rao, T. and Gabr, M.M. (1984) "An Introduction to Bispectral Analysis and Bilinear Time Series Models" . *Lecture Notes In Statistics*, Vol. 24, Springer-Verlag, Berlin.

[18] Walsh, J.I. (1923) A closed set of normal orthogonal functions, *Amer. J. Math.*, 45, 5-24.

Research of Decision Tree on YARN Using MapReduce and Spark

Hua Wang[1], Bin Wu[1], Shuai Yang[1], Bai Wang[1], and Yang Liu[1]
[1] School of Computer Science Beijing University of Posts and Telecommunications
Beijing, China

Abstract - *Decision tree is one of the most widely used classification methods. For massive data processing, MapReduce is a good choice. Whereas, MapReduce is not suitable for iterative algorithms. The programming model of Spark is proposed as a memory-based framework that is fit for iterative algorithms and interactive data mining. In this paper, C4.5 is implemented on both MapReduce and Spark. The result of each layer of the decision tree can be kept in memory in the implementation on Spark. Through the experiments of C4.5, we observed an improvement of 950% on Spark than on MapReduce when the dataset is small. When the number of lines reached 50 million, Spark still kept an improvement of 73%. We concluded the algorithms and applications applicable for MapReduce and Spark. In the discussion section further experiments were performed to confirm our conclusions.*

Keywords: MapReduce, Spark, RDDs, iterative algorithms, decision tree

1 Introduction

In recent years the size of data and information is presenting an explosive growth trend. With the restriction of the amount of memory and computing capability of traditional standalone mode, it is more and more difficult for traditional data mining tools to deal with TB level and PB level data. As a solution to deal with huge amount of data, parallel mechanism has attracted more and more attention. MPI, PVM and MapReduce [1] were all widely used in the past years.

Comparing with traditional parallel methods, MapReduce performs especially well when the size of datasets is large, and is relatively easy to use. By providing parallelization, fault tolerance, data distribution and load balancing in a transparent and easy-to-use way, MapReduce is widely accepted and used. With the implementation of MapReduce, Apache Hadoop is widely used. Hadoop is mainly composed of two parts: MapReduce and HDFS (Hadoop Distributed File System).

With the science development, a number of applications which are based on iterative algorithms [2] appear. Hadoop MapReduce [3] is based on an acyclic data flow model. With the output of the previous MapReduce job as the input of the next MapReduce job, the iterative programs can be accomplished. In such design, the data used in each iteration is reread and reprocessed, wasting a lot of time in I/O operation. Spark [4] is an open source project developed by UC Berkeley AMPLab. With the realization of RDDs [5], a distributed memory abstraction that lets programmers perform in-memory computations on large clusters, Spark provides RDDs transforms and actions for the users to use Spark easily. YARN [6] is the resource and applications manager of a cluster and supports the existence of multiple frameworks.

Decision tree learning is a powerful method for pattern classification. Most current researchers on decision tree mining focus on improving the mining algorithm which only improves the efficiency of the algorithm rather than the capability of the data to be processed. When the amount of data to be processed increases exponentially, it becomes unsuitable in the single point data mining platform. There are also some researches of decision tree on Hadoop. While the iterative algorithms such as decision tree and k-means are not suitable for the disk based frameworks like Hadoop. The memory based frameworks like Spark are proposed with a view to the shortness of MapReduce.

In this paper, we firstly got a thorough understanding of the mechanism of MapReduce and Spark. We found that the implement of RDDs makes Spark suitable for iterative algorithms. By parallelizing the phase of choosing the best split attribute, we implemented C4.5 on MapReduce. In the implementation of C4.5 on Spark, the intermediate result of each iteration is persisted in memory. In the experiments we got the time of each iteration of different sizes of data sets. We found that the implement of C4.5 performs better than that of MapReduce with an improvement of 73%-950%. We got the conclusions that Spark is suitable for iterative algorithms, which are I/O intensive, low computing density and use specific data sets. Considering the mechanism of Spark and the processing procedure, K-means was chosen to perform further experiments. K-means on Spark was about 33 times faster than that of MapReduce. When the lines of data set reached 150 million, Spark still kept an improvement of 400%. Related works are discussed in section 6, and in section 7 we summarized our conclusions and future work. We represented our acknowledgement in section 9.

2 Background

2.1 MapReduce

The MapReduce programming model consists of two functions, map and reduce. The process of MapReduce job is shown in Fig. 1.

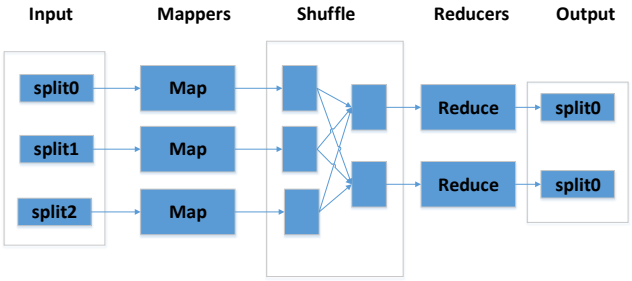


Fig. 1. The process of MapReduce job

As illustrated in Fig. 1, the input data is divided into fixed size of splits (64M by default) by the MapReduce framework. A series of key/value pairs are generated from each split. Then each set of key/value pairs are assigned to a Map task which is designed by the user to implement specific logic, and a new set of intermediate key/value pairs are generated after the Map function. In the Reduce function, each reduce task consumes the (key, list<value>) tuples from map assigned to it. In this phase, a user defined function is invoked that transforms the (key, list<value>) to an output key/value pair. The framework distributes the reduce tasks across the cluster of nodes and deals with the transportation of the appropriate fragment of intermediate data to each reduce task.

As above, the output of Map is directly written into local disk after the shuffle phase. If the algorithm is iterative, the algorithm will read data from external stable storage systems at the start of each iteration. This wastes a lot of time in network bandwidth data replication, and disk I/O.

2.2 Spark

Spark is a distributed computing framework which is designed for low-latency and iterative computation on historical data. Spark provides an easy-to-program interface that is available in Java, Python, and Scala. The major facilities provided by Spark are as follows:

2.2.1 Resilient Distributed Datasets (RDDs)

Spark provides a fault tolerant and efficient memory abstraction called Resilient Distributed Databases (RDDs). When a RDD is created, the users can decide which intermediate RDDs are to be kept in memory and control their partitioning to optimize data placement to get high-efficiency result. RDDs also provide fault tolerance by logging the transformations (map, reduceByKey, filter, etc.).

2.2.2 The operations on RDDs

The operations on RDDs are mainly classified into two categories: transformations and actions. With the operations of transformations, the user can create a new dataset from an existing RDD. All transformations in Spark are lazy in case of that they do not compute their results right away. After the operation of actions, a value is returned to the driver program.

2.2.3 Job Scheduling

When a job is committed to the master of the cluster, a DAG is built from the RDD's lineage graph. A DAG consists of several stages. The stages are divided into two categories: shuffle map stage and result stage. Shuffle map stages are those that their results are input for another stage, while result stages are those that their tasks directly compute the action that initiated a job (count, collect, save, etc.).

2.2.4 Shared Variables

Two common usage patterns of shared variables are provided by Spark: broadcast variables and accumulators. We can broadcast read-only variables and implement counters by using shared variables.

2.3 YARN (Hadoop 2.0)

YARN is the next generation of MapReduce. The programming model and data process engine in MRv1 are reused in MRv2. The principal change of MRv2 to MRv1 is that it split up the two major functionalities of JobTracker into separate daemons. The architecture of YARN with MapReduce and Spark as the applications is shown in Fig. 2.

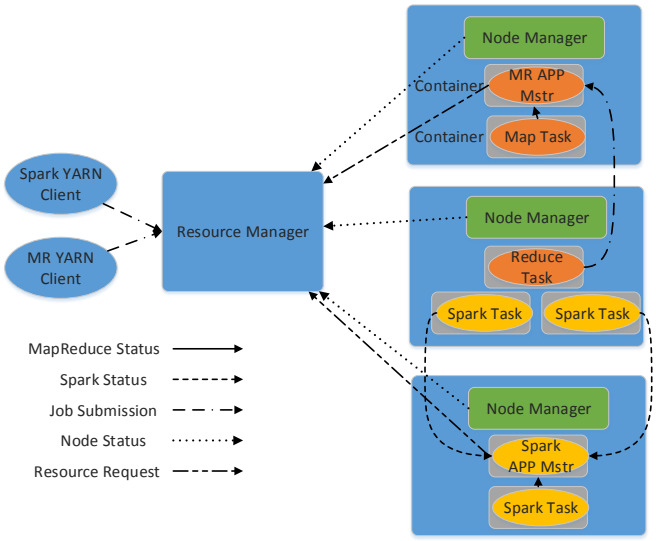


Fig. 2. Example of how Spark computes job stages. [4]

Totally speaking, YARN is also a Master/Slave architecture. ResourceManger is responsible for the uniform resource management and the schedule. When an application is submitted, an ApplicationMaster is needed to track and supervise the job.

3 Decision Tree

Decision Tree is one of the key Data Mining technologies and categorizations. In a Decision Tree, every internal node means a test on an attribute, every branch means the output of a test, and every leaf node store a class label. ID3 [7] is firstly developed by J. Ross Quinlan in 1986. C4.5 [8] is developed by J. Ross Quinlan in 1993, since then ID3 and C4.5 have been widely used and also have a lot development. In this paper, the

parallelization of C4.5 is put forward and realized by MapReduce and Spark. In the experiment of C4.5 with MapReduce and Spark, some conclusions are reached.

On account of the measure of information gain in ID3 is partial to the attributes that have a lot of lines in the data set, C4.5 chooses gain ratio as the extension of information gain. In this pager, C4.5 is selected to be parallelized and realized on MapReduce and Spark.C4.5 adopts the top-down and recursive method to construct a decision tree from the training items and the categories they belongs to. The detail procedures are shown as below.

1) Get the input data set of DSet. Each item in DSet has some attribute values and a class label;

2) Find the gain ratio from splitting on each attribute att;

3) Let att_best be the attribute with the highest gain ratio;

4) Create a decision node that splits on att_best;

5) After splitting on att_best, some subcubes are formed. For each cube of CubeChild the subcubes, go back to 2) to get att_best1 of CubeChild. Att_best1 will be the child of the node formed in 4).

Additionally, some operations of pruning will be performed to overcome the excessive fitting.

The entropy of a data set to be classified is measured as:

$$Info(D) = \sum_{i=I}^{m} p_i log_2(p_i) \quad (1)$$

The p_i means the probability of one item belongs to class C_i, and is measured by $|C_{i,D}|/|D|$. $Info(D)$ is called the entropy of D.The information except $Info(D)$ we need to get accurate classification of the data set is measured as:

$$Info_A(D) = \sum_{J=I}^{V} \frac{|D_j|}{|D|} \times Info(D_j) \quad (2)$$

The $\frac{|D_j|}{|D|}$ acts as the weight of the jth partition. $Info_A(D)$ is the expected information according A to classify the items in D.The information gain is defined as the difference between the original information $Info(D)$ and the new information $Info_A(D)$:

$$Gain(A) = Info(D) - Info_A(D) \quad (3)$$

ID3 uses $Gain(A)$ to get the split attribute. While C4.5 uses split information to normalize information gain:

$$SplitInfo_A(D) = -\sum_{j=i}^{v} \frac{|D_j|}{|D|} \times log_2\left(\frac{|D_j|}{|D|}\right) \quad (4)$$

The standard C4.5 used to split a node is gain ratio, which is shown as follows:

$$GainRatio(A) = \frac{Gain(A)}{SplitInfo(A)} \quad (5)$$

4 C4.5 on MapReduce and Spark

In order to do data mining on YARN using MapReduce and Spark, some tools and infrastructure are required. The architecture of the data mining system is shown in Fig. 3.

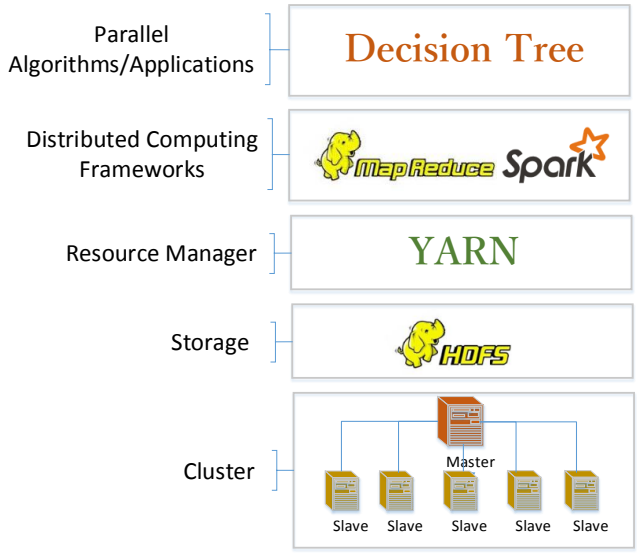


Fig. 3. The architecture of the data mining system

In this study, we used a cluster with ten nodes of Linux operating system as the base infrastructure of the whole system. On top of the infrastructure, we used HDFS (Hadoop Distributed File System) for data storage. Hadoop provides shell operations and APIs for the users to have access to the data stored on HDFS. On top of HDFS, YARN is chosen as the resource manager and applications master to manage the cluster. For data processing and analysis, both MapReduce and Spark are selected with the purpose of comparing the characteristics of MapReduce with Spark. With the MapReduce and Spark frames, it's possible for us to develop parallel algorithms or application. Here Decision Tree is chosen as an example to be implemented on MapReduce and Spark.

4.1 C4.5 on MapReduce

The traditional decision tree algorithm is memory resident, which means that all the data sets are kept in memory during the whole formation process of the decision tree. In this case, the scalability of the algorithm is under restrictions. In this article, we discussed the parallelization of C4.5.

Through the analysis of the process of C4.5, we concluded that the most important part of C4.5 is the phase of the measurement of attribute selection. Choosing the best split attribute occupies most time of the decision tree generating phase. So it is the breakout of parallelizing C4.5 tree to get the greatest degree of this phase's parallelization. In sequence of

the relative independence among different attributes, it is possible for us to use MapReduce to compute the related information needed to calculate the gain ratio of each attribute. Then, the main procedure can get the gain ratio rapidly and get the best split attribute. The main idea of parallelizing C4.5 tree likes the WordCount procedure to some degree. In this paper, we used breadth-first algorithm to get the result tree.

Map phase: Assuming that the training set is $Node_0$, there are m nodes in one layer of the tree. The nodes supposed to be satisfied with:

$$Node_0 - Node^{'} = Node_1 \cup Node_2 \cup Node_3, \ldots, \cup Node_m \quad (5)$$

The $Node^{'}$ is the set of the items that are in leaves.

The duty of map phase is to get the <key, value> form of the item in $Node_0$, and output the data as $Node_1, Node_2, Node_3, \ldots, Node_m$. Key is the id of the $Node_{id}$, the attribute att , the value of att, and the class value. The value is set 1. Map also has the duty to get the total line number of training set and the line numbers of $Node_i$. These statistical works can be done in a single map task.

The reduce phase is to get the sum number of values that has the same key from the output of map phase. Then the <key, sum>s are output to HDFS. A combiner which is similar to reducer, is added before the reducer in order to reduce the size of the data to be transmitted through network. With the result of reduce output, it's a simple job for us to get the gain ratio of each attribute in $Node_i$ and get the split attribute that has the max gain ratio. The flow diagram of the process is shown in Fig. 4.

In the function of map, we can get the split of each line to get <id+att+value+class, 1>, <id, 1>, <"total", 1> as the output of map. The reducer gets the output of map so as to get the sum of the values that have the same key. The output of reducer is put on HDFS. With the information needed to get gain ratio, we can get the best attribute among the attributes that have not been the split nodes.

4.1 C4.5 on Spark

The C4.5 on Spark has the same parallel idea with C4.5 on MapReduce. As Spark has different APIs and operation from MapReduce, being familiar with Spark and its operations is necessary for us to write a Spark application. The diagram of the working flow of Spark is shown in Fig. 5.

An application is also called a driver on Spark. When a job is submitted to YARN, a runtime environment is created. At the same time, a service called BlockManager, which adopted an architecture of master/slaved, is started on each node. Then the application is transformed to a DAG. The DAGScheduler is on duty of executing every stage of the process. The C4.5 code based on Spark is shown in Fig. 6.

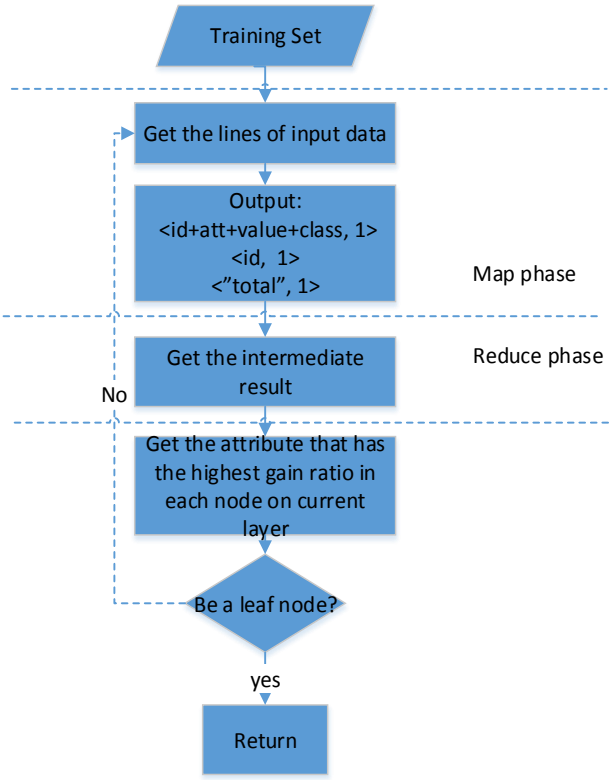


Fig. 4. The flow diagram of C4.5 on MapReduce

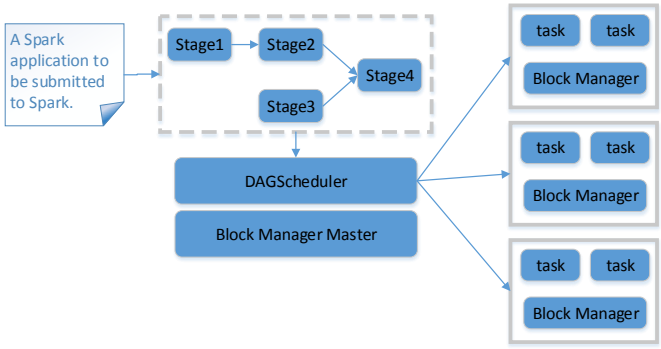


Fig. 5. The working flow of Spark applications

Some preparations are made before executing the driver program. YARN and Spark environment needs to be deployed, and the training data should have been put on HDFS. Spark uses SparkContext to get access to a cluster. We provide the master node IP, the name of the application, the SparkHome and the jar path to SparkContext. We can use SparkContext to get a RDD and read files on HDFS. In the driver program, we need to read data from meta file to get the attributes and values put into HashMaps. The function of textFile() is used to get the meta file RDD on hdfs. After this, we can get every line of RDD to do the initialization.

The whole input dataset is regarded as a RDD. We can use the .cache() method to keep the RDD in memory for reusing. The function of flatMap is almost the same as map in MapReduce framework. We can get some lists of <key, value>

Early-stage preparations:

1.Spark and YARN configuration

2.Putting data on HDFS

Run C4.5Tree Class (the Driver Program)

SparkContext:

The constructor: new SparkContext(master, appName, [SparkHome], [jars]) is called to initialize SparkContext.

Initialization:Initializa

Read and initialize attributes and their possible values from meta file.

RDD:

The input training set is regarded as a RDD on Spark through textFile(path, minSplits): RDD[String] .

flatMap:

Get a list through each input line, including:

1.<id+att+value+class, 1>

2.<id, 1>

3.<"total", 1>

Id means the unique number of a node on current layer.

reduceBykey:

Get the sum of the same key from the RDDs from flagMap.

generateTree:

Get the attribute that has the highest gain ratio in each node on current layer.

Fig. 6. The working process of C4.5 on Spark

from flatMap. The reduceByKey works as the function of reducer in MapReduce to get the sum needed to get the gain ratio of each node. When the information required is worked out, it is possible for us to get the attribute that has the highest gain ratio in each node on current layer.

5 Experiments

Some experiments are conducted to evaluate the performance of our implementation. In this paper, we used a cluster with one master and 9 slaves. All these nodes have an internal memory of 4GB and 4 cores. Each node is installed with Red Hat 4.4.7-3.

The dataset of Lymphography Domain was used in our experiment on MapReduce and Spark. In order to get different size of datasets to evaluate the performance of MapReduce and Spark, the copy method is used to get assigned number of lines.

The Lymphography Domain Data Set was obtained from the University Medical Centre, Institute of Oncology, Ljubljana, Yugoslavia. It was provided by M. Zwitter and M. Soklic. There are 19 attributes including the class attribute. All attribute values in the database have been entered as numeric values. The number of lines of the databases used in this experiment is: 50 thousand, 500 thousand, 2 million, 5million, 8 million, 10 million, 15 million, 30 million and 50 million. The size of a 50 thousand dataset is about 5M.

For that there are 6 layers of the decision tree, we record the time at the end of each layer's iteration. The performance of C4.5 on MapReduce is shown in Fig. 7. In the following figures w means ten thousand.

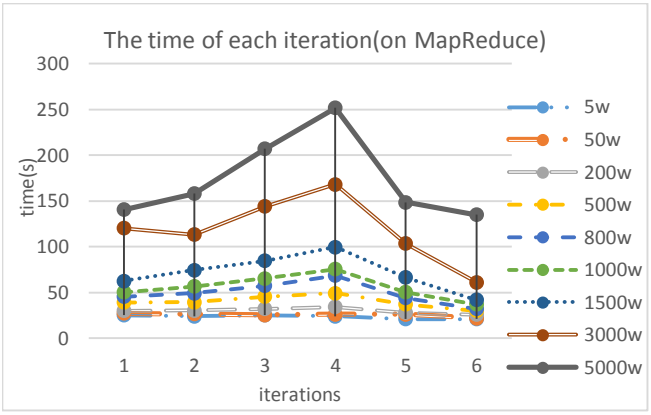


Fig. 7. The performance of C4.5 on MapReduce

In this experiment, we found that the running time is close to the maximum in the 4th layer, for the reason that the layer 4th has the most nodes. During the generation phase of decision tree, matching the candidate rules which contains only the current node's ancestor nodes takes most of the time of the whole phase. At the beginning, there are a small number of candidate nodes results in the short running time. With the number of candidate rules growing, the running time progressively grows. After the 4th layer is built, the running time of single layer reduces for the reducing of nodes. The performance of C4.5 on Spark is shown in Fig. 8.

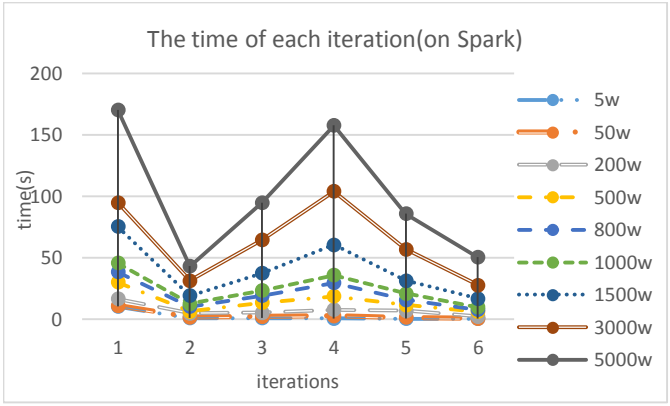


Fig.8. The performance of C4.5 on Spark

The performance curvilinear trend of C4.5 on Spark is almost the same as that of C4.5 on MapReduce except the beginning of the process. With the time used in reading data stored on HDFS and storing the dataset in memory, the running time of the first layer is relatively long. While after the first iteration, the running time of each iteration reduces. The trend after this goes almost the same as that of MapReduce. The running time reaches the peak at the 4th layer.The comparison of C4.5 between MapReduce and Spark is shown in Fig. 9.

From Fig. 9 we can find that when the number of lines is relatively small, i.e. 50 thousand, the running speed of Spark is much higher than that of MapReduce, at about 10.5 times difference. As the amount of data increases, the advantage of Spark reduces gradually. But the speed of C4.5 on Spark is still

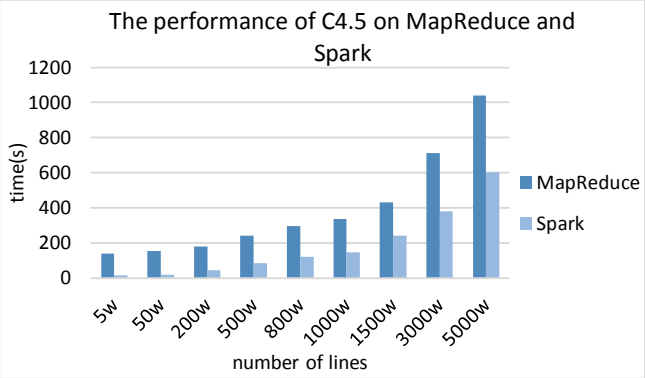


Fig. 9. The comparison of C4.5 on MapReduce and Spark

higher than that of MapReduce, with about 1.73 times faster at a date set of 50 million lines.

6 Discussion

From the experiment in section V, we found that Spark is faster than MapReduce to a certain extent. Considering the characteristics of MapReduce, Spark and the executing process of C4.5, we can get the following conclusions:

1) MapReduce is not suitable for the processing of a small amount of data due to the starting time of a MapReduce job. Compared with MapReduce, Spark does not have this drawback. Even the size of data is very small, the job of Spark also runs fast.

2) With the ability to keep data in memory, Spark is especially fit for iterative algorithms. Spark has the ability of permitting a user to cache the data that will be reused in the algorithm. This is very flexible and useful. Spark saves the time in I/O of reading and writing intermediate result, which occupies a large part of the process of MapReduce.

3) Spark is fit for the situation repeatedly using specific dataset, which can be kept in memory. If the dataset always changes during the whole process, the advantage of Spark over MapReduce becomes relatively poor.

4) Spark is fit for I/O intensive applications. Extremely speaking, the size of dataset is large, but what we do is just to get the number of lines for n times. Spark is very suitable for this situation. While, if the computing density is very high, which takes more time than that of I/O, Spark's advantage over MapReduce is not so obvious.

Through discussing the result of our experiment, it is concluded that Spark is specially fit for the algorithms that are I/O intensive and repeatedly use specific dataset. Among these, K-means [9] is a typical sample. K-means is a clustering algorithm, which aims to divide n items into k cluster, where the items in the same cluster are similar to each other, while the items in different clusters have low similarity. In the process of K-means, the input dataset, which can be kept in memory, will never change during the whole K-means process. Besides, the logic of each iteration of K-means is simpler than that of C.5. The data of K-means in this paper are produced by a specific program. The data has 30 dimensions. It is about 5.37M of 10 thousand nodes. The test datasets are in numbers of 50 thousand, 50 thousand, 1 million, 2 million, 5 million, 10 million, 20 million, 50 million, 80 million and 150 million. The comparison of K-means between MapReduce and Spark is shown in Fig.10 and Fig. 11.

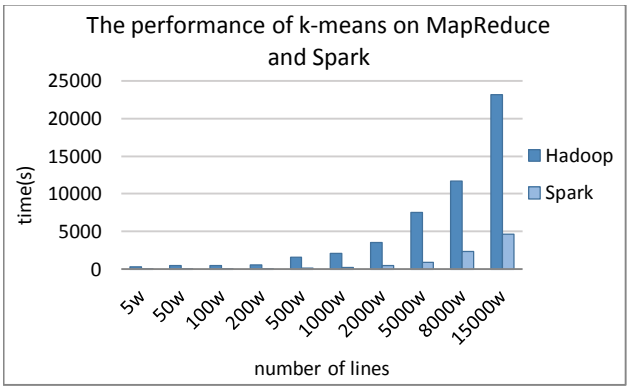


Fig. 10. The comparison of k-means on MapReduce and Spark

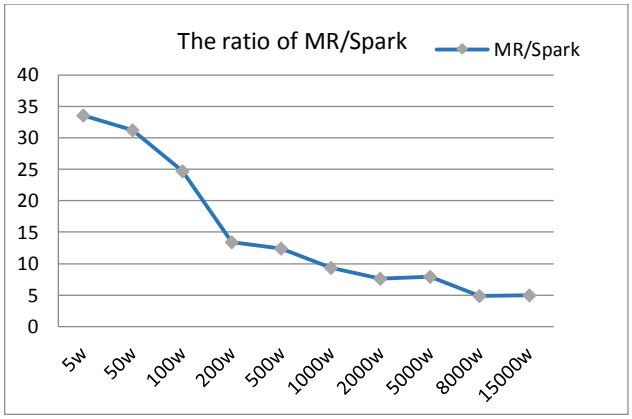


Fig. 11. The ratio of MapReduce to Spark of the performance of k-means

We find that the algorithm of K-means is very fit for Spark. At the beginning, Spark is over 30 times faster than MapReduce. With the growing of size of input data, the advantage of Spark reduces. While even the size of data reaches 150 million, the speed of K-means on Spark is still about 5 times faster than that on MapReduce. All these prove the conclusions we got in section 5.

7 Related work

Nowadays, there are some studies about data mining based on Hadoop, Mahout [10] is an open source project which contains the implementation of common machine learning algorithms based on Hadoop. Oryx [11] is the open source machine learning project of Cloudera based on Hadoop. There are also some researches about data mining on Spark. For example, Spark mllib [12] is a Spark implementation of some common machine learning functionality, which contains binary classification, regression, clustering, etc.. Transwap

data hub [13] is a big data platform based on Hadoop 2.0 and Spark, which also integrates Mahout and R statistics engine.

As to decision tree, [14] [15] and [16] provide some improvement strategies. There are also some researches about decision tree based on MapReduce. [17] and [18] are studies on the implementation of decision tree on MapReduce. Mahout also has the implementation of decision forest based on MapReduce. The research about decision tree on Spark is still rare, and there are also few studies on the comparison of advantages and applicable algorithms between MapReduce and Spark. In this paper we implemented C4.5 on both MapReduce and Spark, and concluded the situations suitable for Spark.

8 Conclusions and future work

As the use of Spark is becoming more and more widespread and YARN has become the new generation of Hadoop, the data mining based on YARN using both MapReduce and Spark has become a future trend. In this study, we implemented C4.5 on MapReduce and Spark. Through the analysis of the mechanism of MapReduce and Spark, it is found that Spark is suitable for I/O intensive and low computing density algorithms. When each iteration uses a specific dataset, Spark performs much better. Otherwise, Spark performs relatively poor. Further experiments of K-means is conducted to prove our conclusions.

This is a basic study where we parallelize C4.5 on MapReduce and Spark. We will try to implement more complicated algorithms to research how to take full advantage of Spark. Through our research of Spark, we will try to improve the performance of data mining algorithms. We will also integrate the algorithms on Spark to common data mining platforms.

9 Acknowledgement

This work is supported by the National Key Basic Resea rch and Department (973) Program of China (No.2013CB329 603) and the National Science Foundation of China (Nos.613 75058,and 71231002).This work is also supported by the Spe cial Coconstruction Project of Beijing Municipal Commission of Education.

10 References

[1] J. Dean and S. Ghemawat, MapReduce: Simplified Data Processing on Large Clusters, OSDI'04: Sixth Symposium on Operating System Design and Implementation, San Francisco, CA, December, 2004.

[2] J. Ekanayake et al., Twister: a runtime for iterative MapReduce, HPDC ' 10 Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing Pages 810-818, 2010.

[3] MapReduce, http://wiki.apache.org/hadoop/MapReduce.

[4] Matei Zaharia, Mosharaf Chowdhury, Michael J. Franklin, Scott Shenker, Ion Stoica. Spark: Cluster Computing with Working Sets.HotCloud 2010. June 2010.

[5] Matei Zaharia, Mosharaf Chowdhury, Tathagata Das, Ankur Dave, Justin Ma, Murphy McCauley, Michael J. Franklin, Scott Shenker, Ion Stoica. Resilient Distributed Datasets: A Fault-Tolerant Abstraction for In-Memory Cluster Computing. NSDI 2012. April 2012.

[6] YARN,http://hadoop.apache.org/

[7] J. R. Quinlan, "Introduction of decision tree," Mach. Learn., vol. 1, pp. 81–106, 1986.

[8] J. R. Quinlan, C4.5: Programs for Machine Learning. San Mateo, CA: Morgan Kaufmann, 1993.

[9] MacQueen, J. B. "Some Methods for classification and Analysis of Multivariate Observations". Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability 1. University of California Press. pp. 281–297. MR 0214227. Zbl 0214.46201, 1967.

[10] Mahout, http://mahout.apache.org/.

[11] Oryx, https://github.com/cloudera/oryx.

[12] Spark mllib, http://spark.apache.org/docs/0.9.0/mllib-guide.html.

[13] TRANSWARP,http://www.transwarp.io

[14] QIAN Wang-Wei, Research on ID3 Decision Tree Classification Algorithm Based on MapReduce, JISUANJI YU XIANDAIHUA, 2012.

[15] Qiu Lu, Xiao-hui Cheng, The Research of Decision Tree Mining Based on Hadoop, 2012 9th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD 2012), 2012.

[16] Rong Cao, Lizhen Xu, Improved C4.5 Algorithm for the Analysis of Sales, 2009 Sixth Web Information Systems and Applications Conference, 2009

[17] Zhu Xiaoliang, WangJian, Research and Application ofthe improved Algorithm C4.5 on Decision Tree, 2009 International Conference on Test and Measurement, 2009.

[18] Amany Abdelhalim, Issa Traore, A New Method for Learning Decision Trees from Rules, 2009 International Conference on Machine Learning and Applications, 2009.

Kernel Approach to Incomplete Data Analysis (KAIDA)

S. Y. Kung and Pei-Yuan Wu

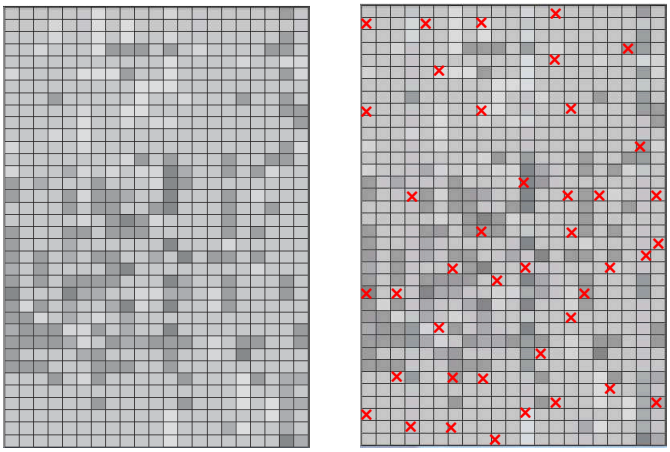
***Abstract*—In many practical scenarios, the data to be analyzed are highly incomplete due to controlled or unanticipated causes such as concerns on privacy and security as well as cost/failure/accessibility of data sensors. This motivates our study on a kernel approach to incomplete data analysis (KAIDA). The first task is to convert partially-specified vectors into fully-specified vectors. To this end, we propose data-masking schemes which lead to two types of fully specified vectors: Singly-Masked (SM) and Doubly-Masked (DM) vectors. Thereafter, we can now define several kernels such as M-linear, M-polynomial, and SM-RBF kernels. In addition we introduce two partial cosine (PC) kernels, i.e., SM-PC and DM-PC. The KAIDA was then applied to both supervised and unsupervised machine learning models. Via simulations on the Wisconsin (breast cancer) and MIT (ALL/AML) datasets, we have compared the proposed kernels. The PC kernels (DM-PC, SM-PC) and M-Poly2 kernels are among the highest performers, each of which topping the chart in certain sparsity regions.**

Keywords*—*missing data analysis, incomplete data analysis (IDA), singly-masked (SM) vectors, doubly-masked (DM) vectors, Mercer condition, kernel methods, partial cosine (PC), resilience against data sparsity, supervised learning, unsupervised learning

I. Introduction

Machine learning analysis depends very much on the specific metric used to characterize the similarity between two vectors in a Hilbert space. Conventionally, the linear or nonlinear inner-product is used as the similarity metric. However, for many (big) training dataset applications, such fully specified vectors are often absent in the learning phase and/or prediction phase. The kernel trick circumvents the need of explicitly defined vector space or even fully specified data vectors. [7]–[9] Indeed, the kernel approach may serve as a promising and unifying platform for nonvectorial and vectorial (fully specified or not) data analysis. More specifically,

- One important application scenario involves nonvectorial data analysis. In the kernel approach to nonvectorial data analysis, the role of the *kernel matrix* is substituted by a *similarity matrix*. The challenge lies in establishing a systematic formulation to effectively characterize the similarity metrics for various types of applications. [6]
- The kernel method also offers a natural approach to IDA, which is also known as missing data analysis (MDA) [9], [18], [19]. There are two critical criteria for the selection of IDA-friendly kernel functions:
  1) The kernel function must be able to cope with IDA
  2) The kernel function must show a strong resilience against data sparsity.



(a) (b)

Fig. 1. (a) A regular dataset may be represented by a data matrix describing the attribute-sample relationship. (b) An incomplete data matrix, where "×'s" represent the missing data.

Fully Specified Data Matrix. With reference to Figure 1(a), a fully-specified $M \times N$ *data matrix* can be explicitly expressed as follows:

$$\mathbf{X} = \begin{bmatrix} x_1^{(1)} & x_2^{(1)} & \cdots & x_N^{(1)} \\ x_1^{(2)} & x_2^{(2)} & \cdots & x_N^{(2)} \\ \cdots & \cdots & \cdots & \cdots \\ x_1^{(M)} & x_2^{(M)} & \cdots & x_N^{(M)} \end{bmatrix} = [\mathbf{x}_1 \ \mathbf{x}_2 \ \cdots \ \mathbf{x}_N].$$

Partially Specified Data Matrix with Random Sparsity. In IDA, it is assumed that some entries of the data matrix are missing and unknown. Moreover, the missing entries may vary from one column to another. As exemplified by Figure 1(b), the locations of missing entries are sparse in a totally random fashion, there is no assumption of any particular sparsity structure.

Kernelized Learning Models and Kernel Matrices. Kernel method is based on kernelized learning model (or better known as kernel trick [7]) characterized by a $N \times N$ symmetric kernel matrix:

$$\mathbf{K} = \begin{bmatrix} K(\mathbf{x}_1, \mathbf{x}_1) & K(\mathbf{x}_1, \mathbf{x}_2) & \cdots & K(\mathbf{x}_1, \mathbf{x}_M) \\ K(\mathbf{x}_2, \mathbf{x}_1) & K(\mathbf{x}_2, \mathbf{x}_2) & \cdots & K(\mathbf{x}_2, \mathbf{x}_M) \\ \cdots & \cdots & \cdots & \cdots \\ K(\mathbf{x}_M, \mathbf{x}_1) & K(\mathbf{x}_M, \mathbf{x}_2) & \cdots & K(\mathbf{x}_M, \mathbf{x}_M) \end{bmatrix}. \quad (1)$$

Organization. This paper will demonstrate that kernelized learning models provide an effective platform for applications pertaining to incomplete data analysis and data mining. Section II proposes a variety of data-masking induced kernel functions

designed to cope with missing data in IDA, including two variants of partial cosine (PC) kernel functions. Section III shows that the proposed kernel functions may not meet the Mercer condition. Consequently, for IDA applications, it is imperative to first convert the similarity matrix into a positive semi-definite matrix before applying a kernel learning model. Section IV shows how to apply these proposed kernel functions to various supervised learning models. Section V demonstrate that highly resilient results are obtained by KRR and/or SVM for two different datasets. Section VI extends KAIDA to unsupervised learning models.

II. Data-Masked Vectors and Kernel Functions

Zero-Padded Vectors. Under the popular zero-mean assumption, it is intuitive to pretend the actual values of the missing data are all equal to zero. Let an original vector be denoted by $\mathbf{x} \in \Re^M$ and its sparsity structure be characterized by means of a data-masking vector $\mathbf{M_x} \in \Re^M$:

$$\mathbf{M_x}(i) = \begin{cases} 1 & \textbf{if } \mathbf{x}_i \text{ is given,} \\ 0 & \textbf{if } \mathbf{x}_i \text{ is missing.} \end{cases} \quad (2)$$

The zero-padded vectors corresponding to $\mathbf{x}$ and $\mathbf{y}$ (otherwise known as singly-masked vectors, c.f. Figure 2) may be respectively represented as follow:

$$\bar{\mathbf{x}} \equiv \mathbf{x}. * \mathbf{M_x} \quad \text{and} \quad \bar{\mathbf{y}} \equiv \mathbf{y}. * \mathbf{M_y},$$

where ".*" denotes element-wise multiplication, a convention adopted by Matlab.

For IDA applications, we need to develop a sparsity-insensitive kernel function to effectively cope with partially defined vectors.

Singly-Masked (SM) Kernel. Note that, while the vectors $\mathbf{x}$ and $\mathbf{y}$ are only partially specified, the two data-masked vectors $\bar{\mathbf{x}}$ and $\bar{\mathbf{y}}$ are actually fully specified and they naturally lead to the following *Singly-Masked Gaussian Radial Basis Kernel (SM-RBF)*:

$$K_{SM-RBF}(\mathbf{x}, \mathbf{y}) = \exp\left(-\frac{\|\bar{\mathbf{x}} - \bar{\mathbf{y}}\|^2}{2\sigma^2}\right). \quad (3)$$

In addition, let us introduce a *Singly-Masked Partial-Cosine Kernel (SM-PC)* as follows:

$$K_{SM-PC}(\mathbf{x}, \mathbf{y}) = \begin{cases} \frac{\bar{\mathbf{x}}^T \bar{\mathbf{y}}}{\|\bar{\mathbf{x}}\| \|\bar{\mathbf{y}}\|} & \textbf{if } \|\bar{\mathbf{x}}\| \|\bar{\mathbf{y}}\| \neq 0 \\ 0 & \textbf{if } \|\bar{\mathbf{x}}\| \|\bar{\mathbf{y}}\| = 0. \end{cases} \quad (4)$$

Doubly-Masked Partial-Cosine (DM-PC) Kernel. With missing data, another effective kernel can be defined as the normalized inner-product based exclusively on the joint set, i.e. features co-existent in both partial vectors. More exactly, given two vectors $\mathbf{x}$ and $\mathbf{y}$ and their masking vectors denoted by $\mathbf{M_x}$ and $\mathbf{M_y}$ respectively, their pairwise-masked partial vectors can be defined as follows:

$$\tilde{\mathbf{x}} \equiv \mathbf{x}. * \mathbf{M_x}. * \mathbf{M_y} \text{ and } \tilde{\mathbf{y}} \equiv \mathbf{y}. * \mathbf{M_x}. * \mathbf{M_y}. \quad (5)$$

The Doubly-Masked (DM) vectors are depicted in Figure 2, which lead us to a useful partial-cosine kernel function:

$$K_{DM-PC}(\mathbf{x}, \mathbf{y}) = \begin{cases} \frac{\tilde{\mathbf{x}}^T \tilde{\mathbf{y}}}{\|\tilde{\mathbf{x}}\| \|\tilde{\mathbf{y}}\|} & \textbf{if } \|\tilde{\mathbf{x}}\| \|\tilde{\mathbf{y}}\| \neq 0 \\ 0 & \textbf{if } \|\tilde{\mathbf{x}}\| \|\tilde{\mathbf{y}}\| = 0. \end{cases} \quad (6)$$

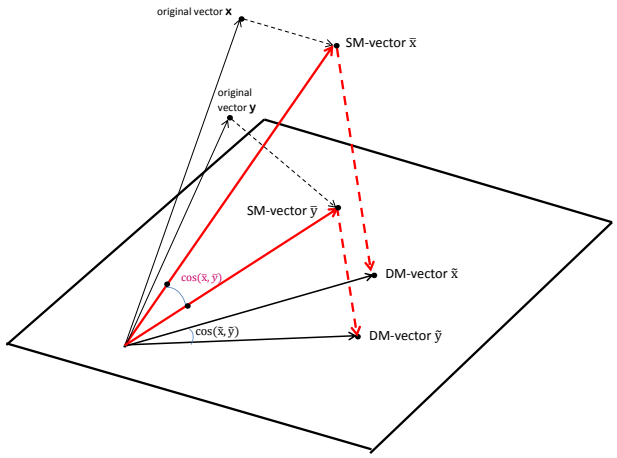


Fig. 2. Illustration of singly-masked and doubly-masked cosine kernels. Note that SM-vectors are fully specified by the masking vector (c.f. Eq.2) associated with each individual vector and the data matrix corresponding to the SM-vectors is well and uniquely defined. In contrast, the mapping to the DM-vectors is pairwise-dependent, i.e. its projected subspace is prescribed by the pairwise co-existing features (c.f. Eq. 5).

Note that, the DM-vector associated with vector $\mathbf{x}$ will change when its partner $\mathbf{y}$ changes to $\mathbf{y}'$. This makes it impossible to define a data matrix associated with the doubly-masked mapping. Nevertheless, the kernel matrix associated with the doubly-masked mapping is well-defined. This property is vital to the feasibility of KAIDA.

M-Linear and M-Poly Kernels. For these kernels, as evidenced by Figure 2 and derivations below, there is no need to differentiate between SM and DM:

- *Masked Linear Kernel (M-Linear).*

$$K_{M-Linear}(\mathbf{x}, \mathbf{y}) = \bar{\mathbf{x}}^T \bar{\mathbf{y}} = \tilde{\mathbf{x}}^T \tilde{\mathbf{y}}. \quad (7)$$

- *Masked Polynomial Kernel (M-Poly).*

$$K_{M-Poly}(\mathbf{x}, \mathbf{y}) = \left(1 + \frac{\bar{\mathbf{x}}^T \bar{\mathbf{y}}}{\sigma^2}\right)^p = \left(1 + \frac{\tilde{\mathbf{x}}^T \tilde{\mathbf{y}}}{\sigma^2}\right)^p. \quad (8)$$

III. The Vital Mercer Condition for KAIDA

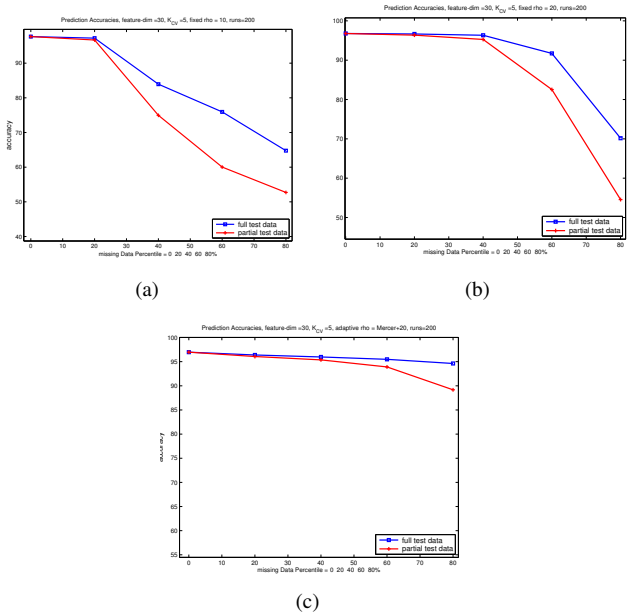
For KAIDA, it is vital to study the Mercer condition of the kernel function selected for incomplete data analysis. Let $K(\mathbf{x}, \mathbf{y})$ be a continuous symmetric kernel that is defined in a closed interval for $\mathbf{x}$ and $\mathbf{y}$. The function $K(\mathbf{x}, \mathbf{y})$ is called a Mercer kernel if it meets the Mercer condition that

$$\int K(\mathbf{x}, \mathbf{y}) h(\mathbf{x}) h(\mathbf{y}) d\mathbf{x} d\mathbf{y} \geq 0, \quad (9)$$

for any square-integrable function $h(\mathbf{x})$, i.e. $\int h(\mathbf{x})^2 d\mathbf{x}$ is finite. If so, according to Mercer's Theorem [5], there exists a reproducing kernel Hilbert space $\mathcal{H}$ and a mapping $\phi : \mathbf{x} \to \phi(\mathbf{x}), \quad \phi(\mathbf{x}) \in \mathcal{H}$ such that $k(\mathbf{x}, \mathbf{y}) = \phi(\mathbf{x})^T \phi(\mathbf{y})$. This is the prerequisite of formally applying kernel methods.

Suppose first that the data matrix is fully specified, i.e. there are no missing data, then the (full) cosine kernel as defined as

$$K_{cos}(\mathbf{x}, \mathbf{y}) = cos(\mathbf{x}, \mathbf{y}) = \frac{\mathbf{x}^T \mathbf{y}}{||\mathbf{x}||\ ||\mathbf{y}||}$$



(a) (b)

(c)

Fig. 3. The critical role of Mercer kernel matrix can be demonstrated by the drastic change of performances displayed by these figures. (a) If $\rho = 10$, the Mercer condition fails frequently when the missing ratio exceeds 20%, gravely compromising the accuracies. (b) If $\rho = 20$, the Mercer condition holds reasonably well for a missing ratio as high as 40%. (c) The best performance is obtained with a Mercer-adjusted parameter: $\rho = 20 + max(0, -min(eig(\mathbf{K}))$.

is a Mercer kernel. The proof is based on the fact that

$$\int K_{cos}(\mathbf{x},\mathbf{y})h(\mathbf{x})h(\mathbf{y})d\mathbf{x}d\mathbf{y} = \| \int \frac{h(\mathbf{x})}{||\mathbf{x}||}\mathbf{x}d\mathbf{x}\|^2 \geq 0, \quad (10)$$

for any square-integrable function $h(\mathbf{x})$. Thus the Mercer condition is verified and the cosine kernel will assure all its induced kernel matrices will always be positive semi-definite.

By the same token, it can be shown that the Mercer condition holds valid for SM-PC. In fact, with the only exception on the DM-PC, the Mercer condition holds for all the other kernel functions proposed in Section II.

Role of the Ridge Parameter ρ on Mercer Condition. Generally speaking, the DM-PC kernel fails the Mercer condition. This may result in a non-positive semi-definite kernel matrix and, consequently, undesirable numerical properties if kernel methods are directly applied. A popular remedy is to convert the kernel matrix into a Mercer matrix by incorporating a positive ridge parameter ρ to the kernel matrix. [9] Our simulations confirm that the DM-PC kernel function is indeed vulnerable to the failure of Mercer condition. On the other hand, once a proper precaution on the Mercer condition is incorporated into the DM-PC kernel matrix, the performance tends to improve significantly.

More elaborately, the performance curves depicted in Figure 3(a)-(c) help highlight the vital importance of Mercer condition. With a small ridge parameter, say $\rho = 10$, the Mercer condition apparently fails frequently if the missing ratio exceeds 20%. The consequence is a drastic drop in accuracy as shown in Figure 3(a). With the ridge parameter increased to $\rho = 20$, the Mercer condition holds well for a broad range of missing ratio, up to 40%, cf Figure 3(b). Finally, as demonstrated by Figure 3(c), the best performance is obtained by adopting a Mercer-adjusted parameter. (More exactly, $\rho = 20 + max(0, -min(eig(\mathbf{K}))$.)

Learning versus testing resilience. Figure 3(c) further demonstrate that the DM-PC kernel appears to offer a strong resilience against data sparsity as long as its (lack of) Mercer condition is properly taken care of.

- **Sparsity in Training Data:** Our simulation on the Wisconsin data set [1] shows that DM-PC kernel exhibits an almost impeccable resilience against data sparsity in training vectors, c.f. the upper blue curves in Figure 3. This is evidenced by the fact that an accuracy as high as 95.5% (out of the peak performance of 97%) may still be retained even in the absence of 60 % of the original data.
- **Sparsity in Training and Testing Data:** With incompletely given test vector, we observe a significant drop in accuracy when compare with what achieved by fully specified test vectors, c.f. the lower red curves in Figure 3. Nevertheless, we hasten to note the overall resilience remains very strong against missing testing data. This is evidenced by the fact that an accuracy as high as 94% may still be obtained by DM-PC KAIDA even in the absence of 60 % of the original data.

IV. KAIDA Supervised Learning Models

We shall show that the kernel approach may be applied to supervised learning problems, where the training dataset and its corresponding teacher values $([\mathcal{X}, \mathcal{Y}] = \{ [\mathbf{x}_1, y_1], [\mathbf{x}_2, y_2], \ldots, [\mathbf{x}_N, y_N] \})$ are both made available during the learning phase. The proposed kernel functions may be applicable to several prominent supervised learning models, including e.g. KRR and SVM. [12]–[14]

A. Kernel Ridge Regressor (KRR)

The KRR learning model [12], [13] incorporates a ridge parameter to assure a positive semi-definite kernel matrix. For full data analysis, the learning model may be defined over either the original space or the kernel-induced space. For IDA, the learning model may no longer be definable over the original space. Nevertheless, we still have the option of adopting the following KRR learning model aiming at finding

$$\underset{\mathbf{a}\in\Re^N,\ b\in\Re}{argmin} \ \|\mathbf{K}\mathbf{a} + \mathbf{e}\, b - \mathbf{y}\|^2 + \rho \mathbf{a}^T\mathbf{K}\mathbf{a}, \quad (11)$$

where $\mathbf{K}$ is the $N \times N$ symmetric kernel matrix prescribed by Eq. 1, and $\mathbf{y} = [y_1 \cdots y_N]^T \in \{-1, +1\}^N$, $\mathbf{e} = [1 \cdots 1]^T$.

KRR Learning Algorithm for IDA. The optimal solution for *empirical decision vector* $\mathbf{a}$ and the threshold b can be solved by the following matrix equation [9]:

$$\begin{bmatrix} \mathbf{K} + \rho\mathbf{I} & \mathbf{e} \\ \mathbf{e}^T & 0 \end{bmatrix} \begin{bmatrix} \mathbf{a} \\ b \end{bmatrix} = \begin{bmatrix} \mathbf{y} \\ 0 \end{bmatrix}. \quad (12)$$

KRR Prediction Algorithm for IDA. After $\mathbf{a}$ and b are learned, the discriminant function is then given by

$$f(\mathbf{x}) = \overrightarrow{k}(\mathbf{x})^T\mathbf{a} + b = \sum_{i=1}^{N} a_i K(\mathbf{x}_i, \mathbf{x}) + b. \quad (13)$$

The decision is based on the sign of the discriminant function:

$$\mathbf{x} \in \begin{cases} \mathcal{C}_+ & \text{if } f(\mathbf{x}) \geq 0 \\ \mathcal{C}_- & \text{if } f(\mathbf{x}) < 0 \ . \end{cases} \tag{14}$$

B. Support Vector Machine (SVM)

SVM Learning Algorithm for IDA. The key component in SVM learning is to identify a set of support vectors useful for shaping the decision boudary. In SVM the empirical *decision vector* $\mathbf{a}$ is solved by the optimization problem:

$$\max_{\mathbf{a}} \quad L(\mathbf{a}) = \mathbf{a}^T\mathbf{y} - \frac{1}{2}\mathbf{a}^T[\mathbf{K} + \rho\mathbf{I}]\mathbf{a}, \tag{15}$$

subject to $\mathbf{e}^T\mathbf{a} = 0$ and $0 \leq a_i y_i \leq C$, $i = 1, \ldots, N$. Note that, for the optimizer to have a meaningful solution, it is necessary to convert $\mathbf{K}$ to be a Mercer matrix, e.g. $\mathbf{K} \rightarrow \mathbf{K} + \rho\mathbf{I}$, especially for the DM-PC kernel, an idea borrowed from the Ridge-SVM learning model [9].

SVM Prediction Algorithm for IDA. After $\mathbf{a}$ and b are learned, the decision can again be formed by Eqs. 13 and 14.

V. KAIDA Supervised Learning Applications

In all the subsequent experiments, we shall assume that the training and testing data (1) have the same missing ratios (from 0% to 60%), and (2) are randomly selected with a 4:1 data size ratio.

A. Wisconsin Breast Cancer Dataset [1]

The original data matrix is a 30×569 matrix, i.e. there are 569 samples each represented by a 30-dimensional feature vector. Each feature (row) is bias adjusted (to zero-mean) and normalized (to unit variance) across all the samples with known entries.

KRR with five induced-kernels Figure 4 shows the performance of all the data-masking induced kernel functions. Note that the accuracies of the proposed DM-PC and SM-PC KAIDA are substantially higher than that by all traditional kernels, with DM-PC holding a slight advantage over SM-PC. In terms of prediction accuracy, we observe that

$$DM{-}PC > SM{-}PC > M{-}Poly2 > M{-}Linear > SM{-}RBF.$$

Comparison between KRR and SVM In the same experiment we have also compared the KAIDA performances based on KRR versus SVM. The results of various kernels are depicted in Fig.5. Note that, for the Wisconsin dataset, SVM consistently outperforms KRR. Also for reasons yet to be determined, the performance by SVM for the SM-PC kernel is substantially lower than what achievable by KRR.

B. MIT ALL/AML Dataset [2], [3]

The original data matrix is a 7129×72 matrix. This may be considered as a *high-dimensional* dataset. Just like many other genomic datasets, the attributes in this dataset are represented by genes which are inherently embedded with abundant inter-feature redundancy. Moreover, there exist many house keeping genes. It is therefore imperative to apply a filtering method to pre-select the most useful features. In our experiment, we pre-select 200 such features based on the so-called Partial Fisher-Discriminant-Ratio (PFDR) [21]:

$$\text{PFDR}(j) = \frac{(\overrightarrow{\mu}_j^+ - \overrightarrow{\mu}_j^-)^2}{(\sigma_j^+)^2 + (\sigma_j^-)^2}, \tag{16}$$

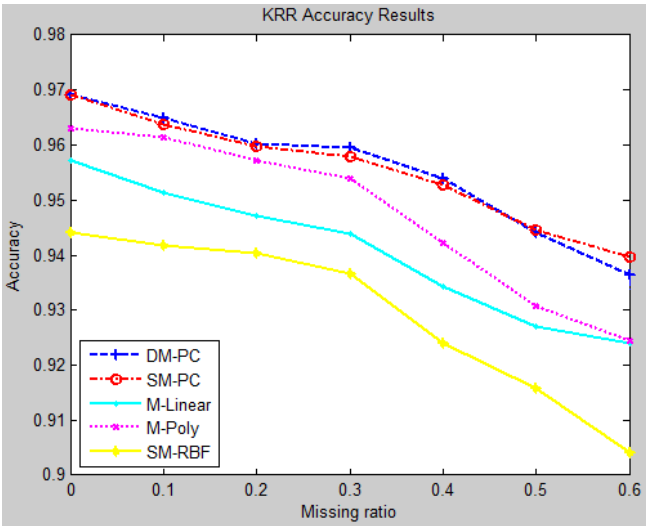


Fig. 4. Five accuracy curves pertaining to Wisconsin Breast Cancer Dataset: (1) DM-PC, (2) SM-PC (3) M-Linear (4) M-Poly2 (bandwidth = 4), and (5) SM-RBF (with a bandwidth = 4).

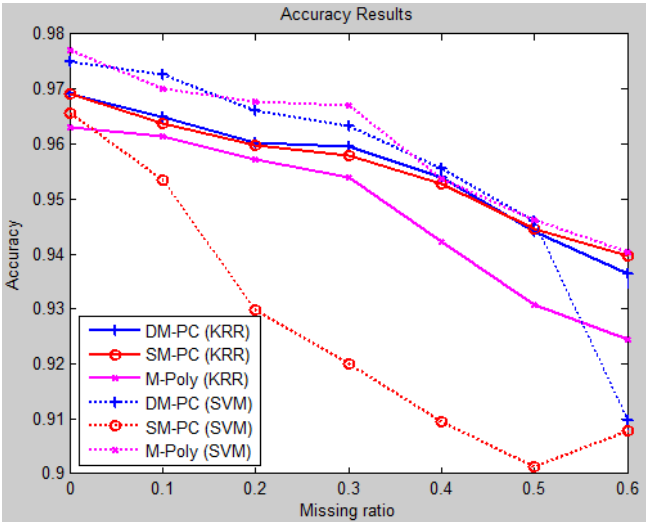


Fig. 5. KRR versus SVM KAIDAs for Wisconsin Breast Cancer Dataset.

where $\overrightarrow{\mu}_j^+$, $\overrightarrow{\mu}_j^-$, σ_j^+ and σ_j^- are derived from the subset $\mathcal{C}^{(j)}$ which denotes the collection of samples where the j'th feature is available. Other than this PFDR pre-selection, the remaining experimental procedure is basically the same as before.

KRR with five induced-kernels Figure 6 shows the accuracies delivered by the five proposed kernel functions. We note that, for the MIT dataset, the DM-PC KAIDA is largely better than the other kernels in the entire sparsity region. By and large, we have

$$DM{-}PC > SM{-}PC > M{-}Poly2 > M{-}Linear \approx SM{-}RBF.$$

Comparison between KRR and SVM In the same experiment we have also compared the KAIDA performances based on KRR versus SVM. The result is shown in Fig.7. In a sharp contrast to the finding on the Wisconsin dataset, it appears that KRR consistently outperforms SVM in MIT ALL/AML dataset. Note that, given full training/test data, a 94.4% prediction accuracy was previously reported by Li, Zhu, and Ogihara [4]. As a comparison, the proposed PC kernels deliver high accuracies even with 50% missing ratio (DM-PC: 96.7%, SM-PC: 96.5%).

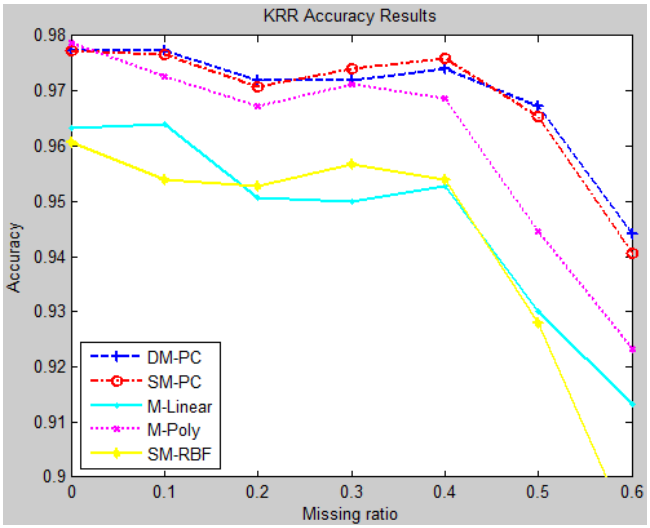


Fig. 6. Five accuracy curves for MIT ALL/AML Dataset: (1) DM-PC, (2) SM-PC (3) M-Linear (4) M-Poly2 (bandwidth = 10), and (5) SM-RBF (with a bandwidth = 10).

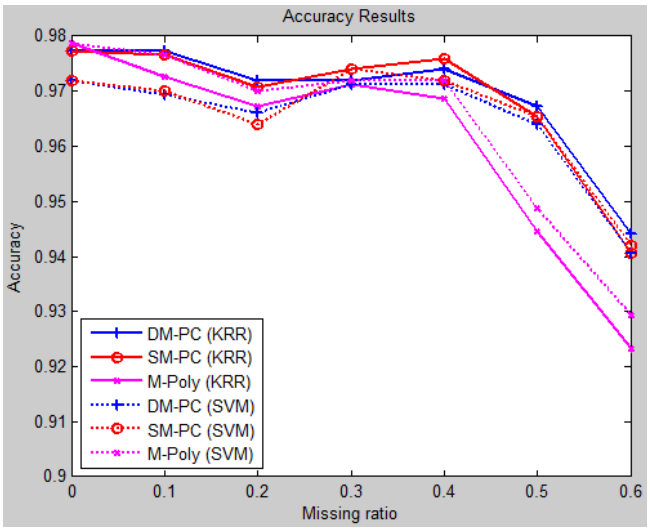


Fig. 7. KRR versus SVM KAIDAs for MIT ALL/AML Dataset.

C. Discussion

It is well recognized that the performance comparison between KRR and SVM will be highly data-dependent. For the Wisconsin dataset, SVM consistently outperforms KRR. In fact, M-Poly2 SVM delivers the highest accuracies among all the 10 combinations (5 kernels $\times$ 2 classifiers), except the cases with missing ratios ($= 10\%$ and $= 40\%$) for which the DM-PC SVM claims the top spot. In contrast, for MIT ALL/AML dataset, KRR outperforms SVM. Indeed, DM-PC KRR largely delivers the highest accuracies among all the combinations, except the cases with missing ratios ($= 30\%$ and $= 40\%$) for which the SM-PC KRR is the top performer.

The achievable accuracies are surprisingly good even when the missing ratio grows very high. We attribute such a performance to the following factors:

1) Because there exists many irrelevant attributes such as house keeping genes in genomic datasets, the proposed Partial Fisher-Discriminant-Ratio (PFDR) is an effective tool to pre-select the most useful features for IDA applications. Our finding confirms that the prediction performance was substantially boosted by reducing the high-dimensional MIT dataset to 200 PFDR features.
2) KAIDA approach to supervised learning appears to be very promising. Three kernels (DM-PC, SM-PC, and M-Poly2) are in average better than the other two (M-Linear and SM-RBF). Note that each of the three kernels) may claim its own winning sparsity regions and it would be premature to rule out any of these kernels.
3) In particular, with any *missing ratio* $\leq 60\%$, the DM-PC kernel appears to be resilient since it consistently yields very high accuracies. This warrants a prominent role of DM-PC for IDA applications. For the DM-PC, it is impossible to formulate the learning model in the original or intrinsic data matrix [9] and we must resort to the kernel-matrix learning models (cf. Eqs. 11 and 15).

VI. KAIDA Unsupervised Learning Models

Likewise, KAIDA may also be applicable to unsupervised cluster discovery, e.g. K-means or SOM. Due to the space limitation, we shall focus our treatment to kernelized K-means. [15], [16] For its extension to kernelized SOM, see [20].

A. Kernelized K-means Learning Model

When applying kernel K-means to incomplete dataset, the optimizer aims at finding (here K denotes the number of clusters)

$$\underset{\mathbf{a}_k \in \Re^N, k=1,\cdots,K}{argmin} \sum_{j=1}^{N} \underset{k=1,\cdots,K}{min} \{\mathbf{K}_{jj} - 2\mathbf{a}_k^T \overrightarrow{\mathbf{k}}_j + \mathbf{a}_k^T \mathbf{K} \mathbf{a}_k \}, \tag{17}$$

where $\mathbf{K}$ is the $N \times N$ kernel matrix and $\overrightarrow{\mathbf{k}}_j$ denotes the j-th column of $\mathbf{K}$.

Roles of Mercer condition in Unsupervised KAIDA. Assuming that Mercer condition is not met and $\mathbf{K}$ is not positive semi-definite, then the cost function in Eq. 17 has no lower bound, i.e. it fails to represent a meaningful minimization criterion. Fortunately, the introduction of a sufficiently large ridge parameter ρ can assure a positive semi-definite kernel matrix, i.e. $\mathbf{K}+\rho\mathbf{I}$, a process named as "*spectral-shift*" [9], can become a Mercer matrix. This in turn assures that the K-means optimizer will have a meaningful solution and the convergence property of the K-means iterations can be guaranteed. [9]

Now the optimizer aims at finding

$$\underset{\mathbf{a}_k \in \Re^N, k=1,\cdots,K}{argmin} \sum_{j=1}^{N} \underset{k=1,\cdots,K}{min} \{\mathbf{K}_{jj} - 2\mathbf{a}_k^T \overrightarrow{\mathbf{k}}_j + \mathbf{a}_k^T [\mathbf{K}+\rho\mathbf{I}] \mathbf{a}_k \}, \tag{18}$$

The spectral-shift approach is popular for unsupervised non-vectorial or incomplete data analysis thanks to a powerful shift-invariance property: the optimal solutions for Eq. 17 and Eq. 18 will be exactly the same. For the proof of the shift-invariance property, see e.g. Dhillon et al. [17] and Kung [9].

The kernel K-means learning model can effectively circumvent the computational problem created by the missing data. There are many methods useful for running the PC-based kernel K-means, including: (1) Kernel trick based method.

[7] and (2) Spectral K-means.[1] [9]–[11] These two methods are mathematically equivalent, so we shall first focus on the spectral K-means method and then establish the connection between the two formulations.

B. Spectral K-means Algorithm

The spectral K-means algorithm contains two steps:

1) The PC-based kernel matrix $\mathbf{K}$ can be formed either for SM-PC (Eq. 4) or DM-PC (Eq. 6). Then perform the spectral decomposition on $\mathbf{K}'$:

$$\mathbf{K}' = \mathbf{K} + \rho\mathbf{I} = \mathbf{U}^T\mathbf{\Lambda}\mathbf{U} = \mathbf{E}^T\mathbf{E}.$$

Note that

$$\mathbf{E} = \mathbf{\Lambda}^{\frac{1}{2}}\mathbf{U},$$

where the j^{th} column of the matrix $\mathbf{E}$ is exactly $\overrightarrow{\mathbf{s}}(\mathbf{x}_j)$. Now each training vector, say $\mathbf{x}_i$, may be mapped to a kernel-induced spectral vector prescribed by the following formula:

$$\overrightarrow{\mathbf{s}}(\mathbf{x}_i) = \mathbf{\Lambda}^{-\frac{1}{2}}\mathbf{U}\overrightarrow{\mathbf{k}}(\mathbf{x}_i), \quad i = 1, \cdots, N \tag{19}$$

2) Apply the conventional K-means to partition the N spectral vectors. The objective is to minimize the criterion:

$$\sum_{k=1}^{K}\sum_{\mathbf{x}_t \in \mathcal{C}_k} \|\overrightarrow{\mathbf{s}}(\mathbf{x}_t) - \overrightarrow{\boldsymbol{\mu}}_k\|^2, \tag{20}$$

where $\overrightarrow{\boldsymbol{\mu}}_k$ denotes the node vector (i.e. the centroid in this case) in $\mathcal{S}$ for the cluster $\mathcal{C}_k$.

C. Derivation of centroids from partial training vectors

In order to provide a common ground for comparing different kernel function, we must resort to their clustering performances displayed in the original vector space. Our comparisons consist of (1) visualization of centroids and (2) quantitative comparison in terms of MSE.

Fully-specified data set: To facilitate our discussion on the estimation of the centroid $\overrightarrow{\mathbf{m}}_k,\ k = 1, \cdots, K$, in the original vector space, we make use of following denotations. Let $\overrightarrow{\boldsymbol{\mu}}_k$ denote the centroid of the k-th cluster in the spectral space. Then

$$\overrightarrow{\boldsymbol{\mu}}_k = \frac{\sum_{i\in\mathcal{C}_k}\overrightarrow{\mathbf{s}}_i}{N_k}.$$

For any nonlinear kernels, there exists no one-to-one mapping which may be used to map centroids in the spectral vector space back to the original vectors space. Put it simply, the forward mapping in Eq. 19 is a nonlinear function and, consequently, due to the nonlinear distortion

$$\overrightarrow{\boldsymbol{\mu}}_k = \frac{\sum_{i\in\mathcal{C}_k}\overrightarrow{\mathbf{s}}_i}{N_k} \ \neq\ \mathbf{\Lambda}^{-\frac{1}{2}}\mathbf{U}\overrightarrow{\mathbf{k}}\left(\frac{\sum_{i\in\mathcal{C}_k}\mathbf{x}_i}{N_k}\right).$$

In fact, the close/remote a vector $\mathbf{x}_i$ is to the centroid the less/more is the nonlinear distortion. This motivates us to adopt a distance-adjusted estimation of the k-th centroid in the original space:

$$\overrightarrow{\mathbf{m}}_k = \frac{\sum_{i\in\mathcal{C}_k} w_{ik}\mathbf{x}_i^{(j)}}{\sum_{i\in\mathcal{C}_k} w_{ik}}, \tag{21}$$

where we propose the following weights assignments:

$$w_{ik} = \exp\left(-\frac{\|\overrightarrow{\mathbf{s}}_i - \overrightarrow{\boldsymbol{\mu}}_k\|^2}{\sigma^2}\right). \tag{22}$$

Coping with Missing Data. With partially unknown training vectors, the task of having to derive the centroid from partial training vectors becomes more challenging. Two guidelines for computing the j-th entry of the k-th centroid $\overrightarrow{\boldsymbol{\mu}}_k$:

- Only the vectors in the k-th cluster with a fully specified j-th feature can vote. This set of vectors can be conveniently represented as $C_k \cap C^{(j)}$.
- The voting weights will be inversely proportional to the corresponding PCD (c.f. Eq.25).

The estimation formula in Eq. 21 is now modified as follows:

$$m_k^{(j)} = \frac{\sum_{i\in\mathcal{C}_k\cap\mathcal{C}^{(j)}} w_{ik}x_i^{(j)}}{\sum_{i\in\mathcal{C}_k\cap\mathcal{C}^{(j)}} w_{ik}}, \tag{23}$$

where again we set (c.f. Eq.22)

$$w_{ik} = \exp\left(-\frac{\|\overrightarrow{\mathbf{s}}_i - \overrightarrow{\boldsymbol{\mu}}_k\|^2}{\sigma^2}\right).$$

Kernel Trick Approach. The equivalence between the spectral K-means and kernel trick approach in Eq. 18 may be highlighted by the fact that

$$\mathbf{K}_{ij} = \overrightarrow{\mathbf{s}}(\mathbf{x}_i)^T\overrightarrow{\mathbf{s}}(\mathbf{x}_j). \tag{24}$$

It implies that the kernelized learning model and the conventional learning model on the spectral space $\mathcal{S}$ are interchangeable in any 2-norm based clustering learning formulations. [9] Furthermore, the solutions from the two formulations can be mathematically related as follows

$$\mathbf{a}_k \ = \ \mathbf{U}^T\mathbf{\Lambda}^{-\frac{1}{2}}\overrightarrow{\boldsymbol{\mu}}_k\ , \quad k = 1, \cdots, K.$$

A kernel-trick approach may be adopted to efficiently compute the squared-value of the *pattern-centroid distance* (PCD) between any training pattern $\overrightarrow{\boldsymbol{\phi}}_t$ and each centroid $\overrightarrow{\boldsymbol{\mu}}_k$ (for $k = 1, \cdots, K$):

$$\begin{aligned}\|\overrightarrow{\mathbf{s}}_t - \overrightarrow{\boldsymbol{\mu}}_k\|^2 &= \|\overrightarrow{\mathbf{s}}_t\|^2 + \|\overrightarrow{\boldsymbol{\mu}}_k\|^2 - 2\overrightarrow{\mathbf{s}}_t\cdot\overrightarrow{\boldsymbol{\mu}}_k \\ &= K(\mathbf{x}_t, \mathbf{x}_t) + \mathbf{a}_k^T\mathbf{K}'\mathbf{a}_k - 2R_{tk},\end{aligned} \tag{25}$$

where R_{tk}, denoting the *data-centroid similarity*(DCS) between the t-th data and the k-th centroid, also has the following kernel-trick derivation [9]:

$$R_{jk} = \frac{1}{N_k}\sum_{i\in\mathcal{C}_k}\mathbf{K}_{ji},\ j = 1, \cdots, N. \tag{26}$$

D. Experimental Results

Reconstruction and Visualization of Centroids in the Orig-

[1] If the Mercer condition fails, then the factorization may lead to imaginary number and thus invalidate the optimization criterion behind K-means and other models.

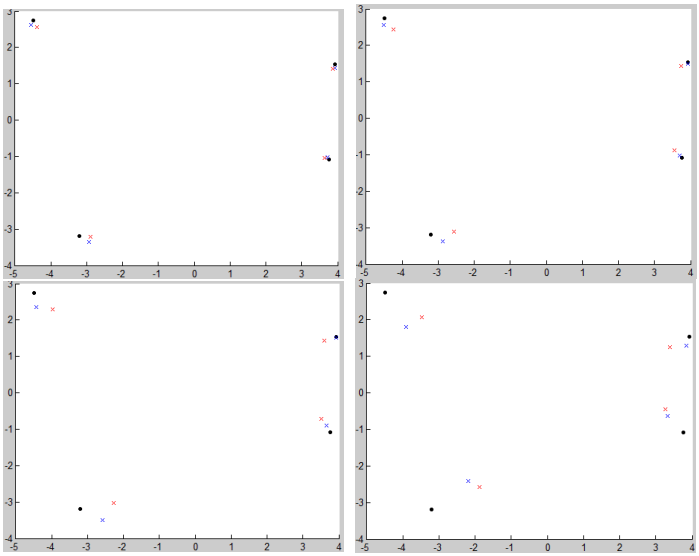


Fig. 8. K means centroids for Wisconsin Breast Cancer Dataset, with various missing rates (a) 20%, (b) 40%, (c) 60% and (d) 80%. Here (and also in the next figure) the centroids are obtained as an average result of 100 runs.

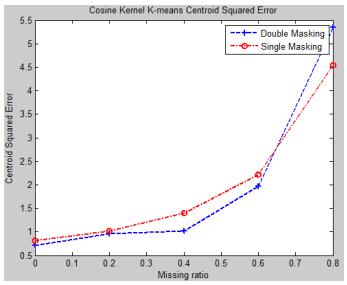


Fig. 9. K-means results for Wisconsin Breast Cancer Dataset. The gaps are measured in MSE. (1) Gap between the original centroids and SM-PC reconstructed centroids and (2) gap between the original centroids and DM-PC reconstructed centroids.(The gaps are measured in MSE.)

inal Vector Space. Using Eq.23 we have reconstructed the centroids for DM-PC and SM-PC based on the Wisconsin dataset. With reference to Figure.8, even with very high sparsity, the centroids reconstructed by the proposed data completion method (with $\sigma = 0.25$) provide a reasonable representation of the full-data K-means centroids (large •). (Here the centroids are obtained as an average result of 100 runs.) As expected, the higher the data sparsity the reconstructed centroids move further away from the original centroids. In terms of their proximity to the original centroids, it is evident that DM-PC (blue $\times$'s) outperforms SM-PC (red $\times$'s)

Quantitative Analysis in terms of MSEs. Figure 9 offers a more quantitative (i.e. MSE) comparison on the K-means results for Wisconsin Breast Cancer Dataset. The MSE metric is used to quantitatively characterize (1) the gap between the original centroids and SM-PC reconstructed centroids and (2) the gap between the original centroids and DM-PC reconstructed centroids. Again, it appears that DM-PC yields a lower MSE than SM-PC.

MIT Dataset. Our simulation study on the MIT dataset (with 200 highest PFDR features) leads to a finding largely consistent with the Wisconsin data set.

VII. Conclusion

Kernelized learning models provide a unified platform for vectorial, nonvectorial, and incomplete data mining applications. This paper proposes a novel *kernel approach to incomplete data analysis* (KAIDA). Both the theoretical and simulation studies underscore the vital importance of using suitable kernel functions to cope with absence or sparsity of data in IDA applications.

References

[1] https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic)

[2] http://www.broadinstitute.org/cgi-bin/cancer/publications/pub_paper.cgi?mode=view&paper_id=43

[3] T. R. Golub, et al. Molecular Classification of Cancer: Class Discovery and Class Prediction by Gene Expression Monitoring. Science 15 October 1999:Vol. 286 no. 5439 pp. 531-537.

[4] T. Li, S. Zhu, and M. Ogihara. Using Discriminant Analysis for Multi-class Classification. In Proceedings of the 2003 International Conference on Data Mining , pages 589-592, 2003.

[5] J. Mercer. Functions of positive and negative type, and their connection with the theory of integral equations. Trans. London Phil. Soc., A209:415V446, 1909.

[6] M. W. Mak, J. Guo, and S. Y. Kung. PairProSVM: Protein subcellular localization based on local pairwise profile alignment and SVM. IEEE/ACM Trans. Comput. Biol. Bioinformatics, 5(3):416V422, 2008.

[7] M. Aizerman, E. A. Braverman, and L. Rozonoer. Theoretical foundation of the potential function method in pattern recognition learning. Automation Remote Control, 25:821V 837, 1964.

[8] B. Scholkopf and A. J. Smola. Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond. Cambridge, MA: MIT Press, 2002.

[9] Kung S.Y., *Kernel Methods and Machine Learning*. Cambridge University Press, 2014.

[10] H. Hotelling. Analysis of a complex of statistical variables into principal components. J. Educational Psychol. 24:498-520, 1933.

[11] B. Scholkopf, A. Smola, and K.-R. Muller. Nonlinear component analysis as a kernel eigenvalue problem, Neural Comput., 10:1299V1319, 1998.

[12] Hoerl A. E. and Kennard R. W. , *Ridge Regression: Biased Estimation for Nonorthogonal Problems.* Technometrics, vol. 12, No. 1, pp. 55-67 Feb., 1970.

[13] A. N. Tychonoff. On the stability of inverse problems. Dokl. Akad. Nauk SSSR, 39(5):195V198, 1943.

[14] V. N. Vapnik. The nature of Statisticl Lerning Theory. New York: Springer-Verlag, 1995.

[15] R. O. Duda, P. E. Hart, and D.G. Stork. Pattern Classification, 2nd edition. New York: Wiley, 2011.

[16] E. W. Forgy. Cluster analysis of multivariate data: Efficiency vs. interpretability of classifications. Biometrics, 21:768V769, 1965.

[17] I. S. Dhillon, Y. Guan, and B. Kulis. Kernel K-means, Spectral Clustering and Normalized Cuts. In Proc. 10th ACM KDD Conference, Seattle, WA, August 2004.

[18] J.W. Graham. Missing data analysis: Making it work in the real world. Annual review of psychology, 2009

[19] T. D. Pigott. A Review of Methods for Missing Data. Educational Research and Evaluation 2001, Vol. 7, No. 4, pp. 353-383

[20] T. Kohonen. Self-Organizing Maps, 2nd edition. Berlin: Springer, 1997.

[21] P. Pavlidis, J. Weston, J. Cai, and W. N. Grundy. Gene functional classification from heterogeneous data, In *Int. Conf. on Computational Biology*, (Pittsburgh, PA), pp. 249–255, 2001.

On the usage of Sorting Networks to Big Data

Blanca López and Nareli Cruz-Cortés
Artificial Intelligence Laboratory,
Centro de Investigación en Computación,
Instituto Politécnico Nacional (CIC-IPN),
México D.F., México Country

Abstract— *Sorting data in a computer is maybe the most popular classical task in Computer Science. For the majority of applications the main goal is to minimize the number of comparisons and execution time that the sorting algorithm consumes.* Sorting Networks *are algorithms that perform exactly the same number of comparisons to order any input permutation for a given input data size. That is, each step does not depend on the result of a previous comparisons. Thus, designing* Sorting Networks *with a minimal number of comparisons becomes a very important task. However, it is an NP-hard problem. Actually, the optimal* Sorting Networks *with a minimal number comparisons (or at least close to the optimal) for small input data sizes from 3 to 16 are published in the specialized literature. Of course, these input data sizes are very small to be used in real world problems. In this work we propose a new strategy to improve the QuickSort performance by coupling it with some Sorting Networks to large input data. The results demonstrate it helps reducing the sorting execution time.*

Keywords: Sorting Networks, QuickSort

1. Introduction

Sorting Algorithms are maybe one of the most studied problems in Computer Science, from the theoretical and practical points of view. Applications of them can be found in Data Processing Systems, Network Communication Systems, Image Processing, Artificial Intelligence, Cryptography, Computer Security, Information Systems, among many others.

A large set of Sorting Algorithms can be found in the specialized literature, such as: quicksort, bubble sort, merge sort, shell sort, heapsort, insertion, introsort, shear sorting, etc. Choosing the most efficient algorithm usually depends on the type of application at hand. In general, the Sorting Algorithms can be classified into two groups: the adaptive and non-adaptive. An adaptive algorithm executes its compare-interchange operations depending on the input data. On the other hand, the non-adaptive algorithms have fixed operations which are executed no matter the configuration of the input data (e. g. all the possible permutations). They always execute the same compare-interchange operations. Sorting Networks (SN) are an example of the non-adaptive algorithms.

Taking advantage of the divide-and-conquer strategy utilized by the QuickSort, it is designed a strategy where some SN are coupled to it in order to reduce the comparisons performed by the QuickSort.

The remaining of this paper is organized as follows. In Section 2 some basic concepts about Quicksort and Sorting Networks are presented. In Section 3 the proposal is explained. Section 4 presents the experiments and results. Finally in Section 5 some conclusions are drawn.

2. Basic Concepts

2.1 Quicksort Algorithm

Quicksort (also known as Partition-Exchange Sort) was first presented in 1960 by Tony Hoare [4]. It uses a divide-and-conquer strategy by dividing a large list into two smaller sublists. A sublist with the smallest values and another with the greatest. Then, each sublist is recursively ordered. The algorithm is as follows:

1) Choose an element from the list that will be called *pivot*.
2) Order the list in such a way that all the values which are less than the pivot will be located to its left (before the pivot). Further, all the values greater than the pivot will be located to its right (after the pivot). This way, the value in the pivot is on its final position.
3) For each sublist, repeat the previous steps in a recursively manner until the sublists size is zero or one.

This idea is illustrated in Figure 1. QuickSort is a very efficient algorithm that on the average and best cases makes $O(n \log n)$ comparisons for sorting n elements. In the worst case it makes $O(n^2)$. Some variants to this algorithm have been presented in [6][3] where their authors proposed some modifications to reduce the execution time.

2.2 Sorting Networks

SN are algorithms with the main feature of being *oblivious*, it means that their current operations (comparisons) do not depend on the input data or the previous comparisons [5][7]. Unlike other well known sorting algorithms (bubble

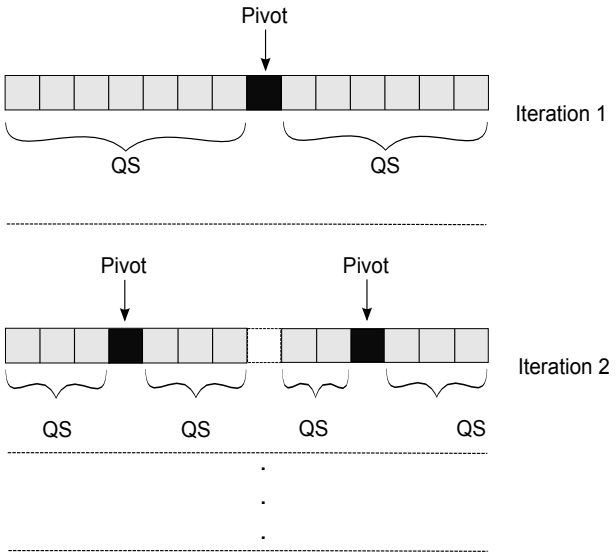


Fig. 1
RECURSIVE PARTITION OPERATION OF THE QUICKSORT ALGORITHM

sort, quicksort, etc.), the sequence and number of comparisons are exactly the same no matter the input configuration (permutation). The SN exhibits two main features:

- The comparisons (called comparators) are fixed before the SN execution,
- Some comparisons can be executed in a parallel manner.

A SN is composed by a set of *comparators*, where each of them executes an action *compare-interchange* between two elements (a, b). The element a must be not grater than b, if so, the values must be interchanged to (b, a). So, for a given input list with size n, the set of comparators conforming the SN are applied to it, then the output is the list monotonically non decreasing ordered.

Typically, the SN are graphically represented by n horizontal lines representing the n input data. Further, some vertical lines that represent comparisons between the value at its top extreme and the value at its bottom. If the value at the top is grater than the value at the bottom, these values must be swapped.

The input data are placed at the left, then, after they have traveled across the horizontal lines and executed the comparisons found, the output is obtained at the right. The data must be ascendant sorted from top to bottom.

See for example a SN for $n = 4$ inputs illustrated in Figure 2. Each input data is set on the horizontal lines labeled as x_0, x_1, x_2, x_3. The vertical lines are the comparators c_0, c_1, c_2, c_3, c_4, each receiving two values, i. e., the comparator c_0 receives the values x_0 and x_1, and so on. All the data values go from left to right executing a

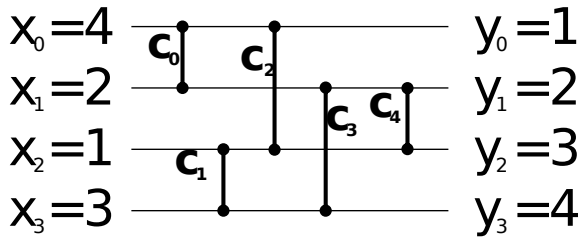


Fig. 2
SORTING NETWORK FOR $n = 4$ INPUTS.

compare-interchange each time a comparator is found. So, the comparators c_0 and c_1 are executed first, then c_2 and c_3, and finally c_4. c_0 evaluates $4 > 2$, thus the values of x_0 and x_1 are swapped. c_1 evaluates $1 < 3$, so the values of x_2 and x_3 remain without change. This process continues until all the comparators are applied, so the final sorted list y_0, y_1, y_2, y_3 at the right accomplishes $y_0 \leq y_1 \leq y_2 \leq y_3$.

As a matter of fact, if an optimal SN for input size n can be designed (i. e. with minimal number of comparators), then it means that is the best manner to sort n data. Designing SN with minimal number of comparators and/or high parallelism is a classical interesting problem in Computer Science. Actually, nowadays it is an open research area.

It is important to notice that the optimal SN for input size greater than $n = 16$ are not know. Actually, only lower bounds regarding the number of comparators are theoretically known [5]. The most studied SN is the one with input size $n = 16$, which is a relatively small value, considering the huge quantity of information that the modern systems must handle. The best known SN $n = 16$ has only 60 comparators, for example, the one designed by Green [5] is illustrated in Figure 3.

In [2] K. E. Batcher proposed an interesting algorithm called Merge Odd-Even to merge two SN into one. That is, if we have a SN with input size n, then, it is possible to obtain a SN with input size $2n$ by merging two copies of the original SN size n each. By following this algorithm it is possible to obtain SN with larger input sizes [1].

An example, to increase the size of input data in $2n$ from SN for $n = 4$. A set of operations to order and two output lists "g" and "h" are considered. In the Figure 4 are shown two lists to re-arrange. The list "t" has the numbers $\{t_1, t_2..., t_g\}$ in ordered. At the same time, second list called "w" are composed by $\{w_1, w_2, ..., w_h\}$. The "g + h" is the output of the merging network, the numbers of the merged lists in ascending order are $\{u_1, ..., u_{g+h-1}, u_{g+h}\}$. i.e., at first, a list "g + h" can be build by merging network with the odd-indexed numbers of the two input lists and the even-

[1]Usually SN for input sizes greater than $n = 16$ are considered as *large*.

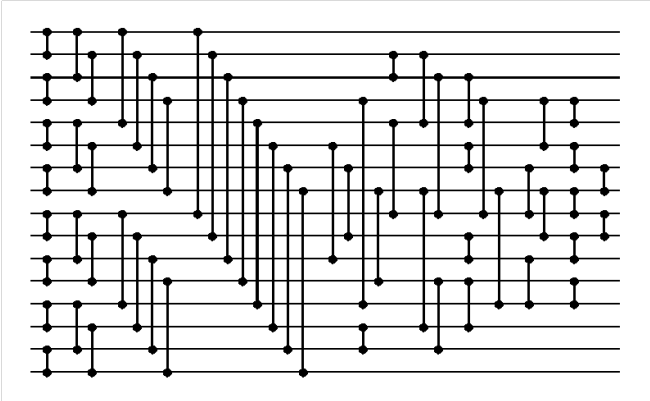


Fig. 3
SN WITH INPUT SIZE $n = 16$ DESIGNED BY GREEN. IT IS THE BEST KNOW WITH 60 COMPARATORS.

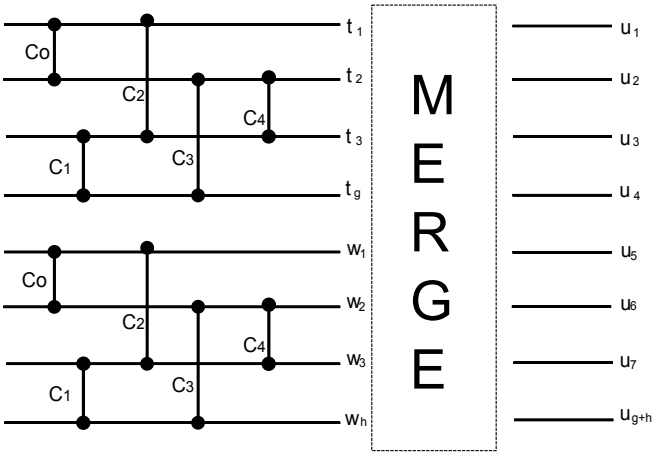


Fig. 4
ODD-EVEN MERGESORT SCHEME FOR TO ORDER TWO SN FOR $n = 8$ INPUTS IS CONSTRUCTED BY TWO SN FOR $n = 4$.

indexed numbers of the two input lists. The lowest output of the odd merge is left alone and becomes the lowest number of final list. The steps to ordered two lists are :

1) Merge the keys of "t" by odd-indexed $\{t_1, t_3, t_5, ...\}$ with the "w" odd-indexed $\{w_1, w_3, w_5, ...\}$ to form the sequences $\{x_1, x_2, x_3, ...\}$ of the keys in order.
2) Merge the keys of "t" by even-indexed $\{t_2, t_4, t_6, ...\}$ with the "w" even-indexed $\{w_2, w_4, w_6, ...\}$ to form the sequences $\{y_1, y_2, y_3, ...\}$ of the keys in order.
3) Then set $u_1 = x_1$ and use parallel comparators to set $u_{2i} = min(x_{i+1}, y_i)$ and $u_{2i+1} = max(x_{i+1}, y_i)$ $i = \{1, 2, 3, ...\}$

The step 1 and step 2 can be apply in parallel and each of the involves about $(g + t)/2$ keys. A more detailed information about the Theorem is [2][5][1][8].

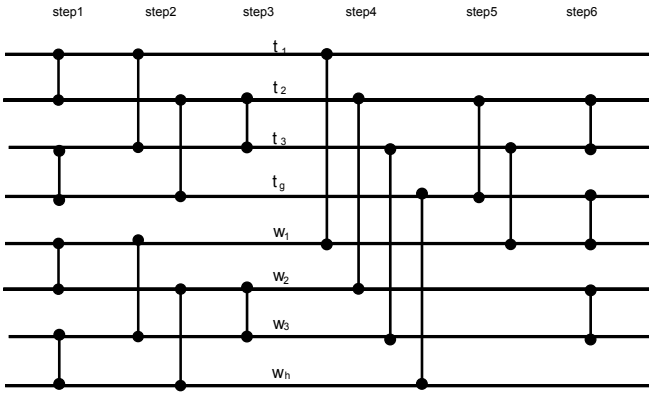


Fig. 5
ODD-EVEN MERGESORT SCHEME. TWO SN FOR $n = 8$ INPUTS IS CONSTRUCTED BY TWO SN FOR $n = 4$.

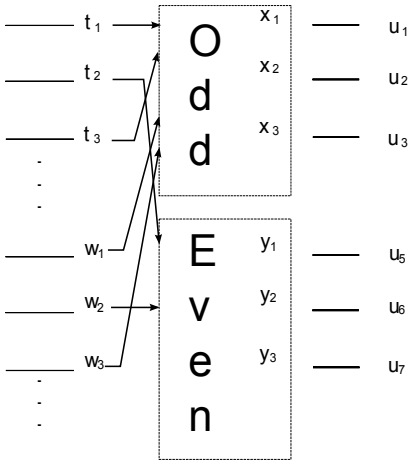


Fig. 6
ODD-EVEN MERGING SCHEME.

In regard to build a SN to $n = 8$ from two SN for $n = 4$ with the Odd-Even merge method, the Figure 5 exhibit the number of comparators that it has full. It has 19 comparators in 6 steps or "layers". The step $1 - 3$ arranged two list to $n = 4$, both in non decreasing order. In step 4 was used four comparators to re-arrange (merge) the 8 elements and then, two comparators to merge the four ordered lists (step 5) and then, in step 6 was used three comparators to merge the ordered sequences to form one ordered list containing all 8 elements.

Nevertheless, this method works better for small input sizes, and decreases its performance as the input sizes increase, i.e. for large input sizes the resulting SN would have more comparators than the optimal.

Notice that for a given input data size n the corresponding SN has fixed its comparators, which means that a new SN must be designed if the input data size is different than n.

In this sense the SN are less flexible than the conventional adaptive sorting algorithms.

3. Proposal

Considering that the QuickSort is a very efficient algorithm based on a divide-and-conquer strategy, and the optimal SN are only known for small input data sizes, this proposal consists on coupling a SN with the QuickSort algorithm to order big input data in an efficient manner. The algorithm will be named Quick+SN.

The general idea is to apply QuickSort in conventional way to the input data as many times as necessary until the sublists are small enough to be sorted by a given SN of input size n. This idea is illustrated in Figure 7. With this small change in the QuickSort it is possible to improve its execution time while maintaining its flexibility, in the sense that the algorithm is able to order any input data size.

Let us suppose that we have selected a determined SN to work with an input size equal to $n = 16$ (which is the greatest near-optimal known). For a given input data A to be ordered with size Z (where Z is a large number), the algorithm Quick+SN is defined as follows:

1) Divide an array A into two parts (a_l and a_r) by selecting the element in the middle position as pivot denoted by m.
2) Compare each element of a_l and a_r against the pivot and move all the elements less than m to the left array a_l, and all the elements greater than m to the right array a_r.
3) Verify if the resulting lists are size n:
   - If so, then the sublist is sorted by the SN.
   - If not, then go to step 1 recursively.

The output of this algorithm is the list A completely ordered.

Notice that Quick+SN works by applying a specific SN for a determined input size n. Therefore, only if a resulting list has exactly n elements then the SN can be applied to it. Each time that the QuickSort splits the list into two, it is not possible to know the resulting sizes. Hence, it is not possible to know in advance the number of times that the SN will be applied.

4. Experimental design

In order to assess the proposed algorithm efficiency, a set of experiments were conducted. The Quick+SN was applied to data lists (numbers) with different input sizes, input permutations. Additionally, it was experimented by using different SN. The configuration for each experiment is a combination of the following options:

- Input data size: Two lists conformed by 1,000,000 and 10,000,000 numbers to be sorted were used as input.
- Input permutations: Three different configurations of the input lists were used: Randomly generated numbers, numbers which are already ordered, and inverse ordered numbers.
- Two different SN were utilized for $n = 16$ and $n = 256$.

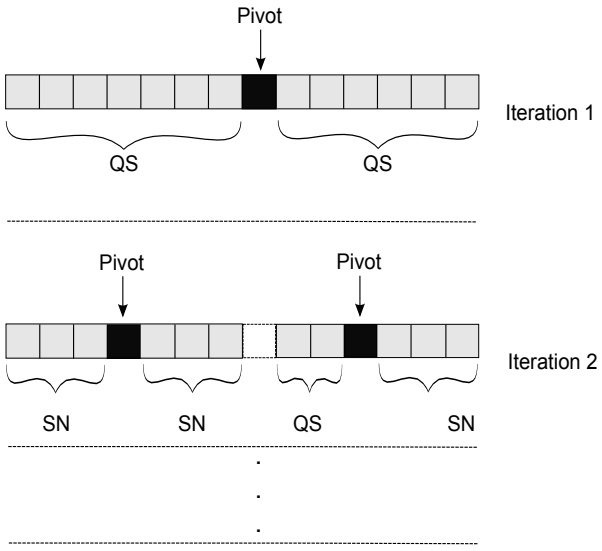


Fig. 7
SCHEME OF THE PROPOSED QUICK+SN ALGORITHM

Therefore, we have twelve experiment sets with different configurations. Each set was executed 30 times. The time and comparisons performed was taken by the Quick+SN applied to each set was obtained. The statistical results about the time are shown in Table 1 and 2 and respect to the number of comparisons are exhibits in the Table 3 and 4. All of them have as first column the input configuration, as second column show the size of the input data, in the third columns are showing the time consumed (in seconds) using Quick+SN and the other case, the number of comparisons performed. In the last column the original Quicksort results are shown. The Table 1 and 3 are presented the time and comparison performed of a SN for $n = 16$. The SN with $n = 256$ was designed by combining copies of Green's SN (shown in Figure 3) with the algorithm Merge Odd-Even mentioned in Section 2.2. Its performance is presented in Table 2 and 4.

It can be observed for the case of random sorted input data, that the Quick+SN with $n = 16$ outperforms the original QuickSort. However, for the cases where the input data was already sorted or inverted, the QuickSort obtained better results than Quick+SN $n = 16$. Thus, for all the configurations and input data sizes, the Quick+SN with $n = 256$ obtained best results with the minimal execution times.

Table 1

STATISTICAL RESULTS (IN SECONDS) OF QUICK+SN AND QUICKSORT. SN TO $n = 16$ WAS UTILIZED.

Input Configuration	Input Size Z	**QS+SN** with $n = 16$		Original QuickSort
Random	1,000,000	Average	0.129118	0.124313
		Median	0.114469	0.112387
		Best	0.112189	0.10922
		Worst	0.239472	0.226902
	10,000,000	Average	1.123340	1.132025
		Median	1.120732	1.131792
		Best	1.119377	1.123426
		Worst	1.149899	1.15022
Ordered	1,000,000	Average	0.111787	0.111292
		Median	0.109365	0.109441
		Best	0.109172	0.109157
		Worst	0.130594	0.126134
	10,000,000	Average	1.121114	1.116137
		Median	1.119669	1.116119
		Best	1.119038	1.116025
		Worst	1.135499	1.11630
Inverse	1,000,000	Average	0.100668	0.097639
		Median	0.100208	0.097374
		Best	0.099006	0.097194
		Worst	0.101557	0.102348
	10,000,000	Average	1.136367	1.127812
		Median	1.13466	1.125864
		Best	1.134569	1.121864
		Worst	1.18243	1.171423

Table 2

STATISTICAL RESULTS (IN SECONDS) OF QUICK+SN AND QUICKSORT. SN TO $n = 256$ WAS UTILIZED.

Input Configuration	Input Size Z	**QS+SN** with $n = 256$		Original QuickSort
Random	1,000,000	Average	0.1110462	0.124313
		Median	0.108356	0.112387
		Best	0.107208	0.10922
		Worst	0.124825	0.226902
	10,000,000	Average	1.056246	1.132025
		Median	1.055244	1.131792
		Best	1.052281	1.123426
		Worst	1.074649	1.15022
Ordered	1,000,000	Average	0.109837	0.111292
		Median	0.10799	0.109441
		Best	0.107806	0.109157
		Worst	0.124511	0.126134
	10,000,000	Average	1.050993	1.116137
		Median	1.05931	1.116119
		Best	1.050836	1.116025
		Worst	1.052154	1.11630
Inverse	1,000,000	Average	0.092927	0.097639
		Median	0.09282	0.097374
		Best	0.092252	0.097194
		Worst	0.093349	0.102348
	10,000,000	Average	1.050894	1.127812
		Median	1.050839	1.125864
		Best	1.050763	1.121864
		Worst	1.051983	1.171423

Table 3

STATISTICAL RESULTS (IN COMPARISONS PERFORMED) OF QUICK+SN AND QUICKSORT. SN TO $n = 16$ WAS UTILIZED

Input Configuration	Input Size Z	**QS+SN** with $n = 16$		Original QuickSort
Random	1,000,000	Average	21123958.73	20962847.72
		Median	20113717	20095717
		Best	20049370	20049190
		Worst	29118486	28203428
	10,000,000	Average	236476813.09	236476536.64
		Median	236477269	236476378
		Best	236476282	236476088
		Worst	236477269	236476983
Ordered	1,000,000	Average	20200661.81	20152879.55
		Median	20049710	20049230
		n Best	20049523	20049230
		Worst	21375129	20945874
	10,000,000	Average	236476813.09	236476536.64
		Median	236476728	236476378
		Best	236476282	236475960
		Worst	236477269	236476983
Inverse	1,000,000	Average	20200661.82	20152879.55
		Median	20049710	20049283
		Best	20049523	20049230
		Worst	21375129	20049230
	10,000,000	Average	236476813.09	236476536.67
		Median	236476728	236476378
		Best	236476282	236476307
		Worst	236477269	236476983

Table 4

STATISTICAL RESULTS (IN COMPARISONS PERFORMED) OF QUICK+SN AND QUICKSORT. SN TO $n = 256$ WAS UTILIZED

Input Configuration	Input Size Z	**QS+SN** with $n = 256$		Original QuickSort
Random	1,000,000	Average	20238789.7	20962847.72
		Median	20095717	20095717
		Best	20019190	20049190
		Worst	28274331	28203428
	10,000,000	Average	236476741.45	236476536.64
		Median	236476378	236476378
		Best	236475960	236476088
		Worst	236479236	236476983
Ordered	1,000,000	Average	20132879.55	20152879.55
		Median	20049283	20049230
		n Best	20049230	20049230
		Worst	20199704	20945874
	10,000,000	Average	236476541.45	236476536.64
		Median	236475960	236476378
		Best	236475960	236475960
		Worst	236479236	236476983
Inverse	1,000,000	Average	20148658.28	20152879.55
		Median	20049283	20049283
		Best	20049230	20049230
		Worst	20945874	20049230
	10,000,000	Average	236477000.60	236476569.67
		Median	236476378	236476378
		Best	236475960	236476378
		Worst	236479236	236476983

5. Conclusions

Taking advantage of their inherent features, it was proposed a combination of the well-known algorithm QuickSort with the algorithm called Sorting Networks. The experimental results showed that the proposal is competitive by obtaining better execution times than the original QuickSort. It was also noticed that the execution times were improved when SN for larger input data sizes were utilized.

It is necessary to experiment with larger input data, and also with SN for different sizes. Further, a formal study related to the algorithms complexity is necessary.

6. Acknowledgments

The authors acknowledges support from CONACyT through projects number 180421 and 132073. The first author acknowledges support from CONACyT through a scholarship to pursue graduate studies at CIC-IPN.

References

[1] S. W. Baddar and K. E. Batcher. *Designing Sorting Networks: A new Paradigm*. Springer, 2011.

[2] K. E. Batcher. Sorting networks and their applications. In *Proceedings of the April 30–May 2, 1968, spring joint computer conference*, AFIPS '68 (Spring), pages 307–314, New York, NY, USA, 1968. ACM.

[3] D. Cantone and G. Cincotti. Quickheapsort, an efficient mix of classical sorting algorithms. In Gambosi G. Bongiovanni G.C. and Petreschi R., editors, *CIAC*, volume 1767 of *Lecture Notes in Computer Science*, pages 150–162. Springer, 2000.

[4] C. A. R. Hoare. Algorithm 64: Quicksort. *Commun. ACM*, 4(7):321–, July 1961.

[5] D. E. Knuth. *The Art of Computer Programming, Volume III: Sorting and Searching, 2nd Edition*. Addison-Wesley, 1998.

[6] U. Kocamaz. Increasing the efficiency of quicksort using a neural network based algorithm selection model. *Inf. Sci.*, 229:94–105, April 2013.

[7] D. G. OConnor and R. J. Nelson. Sorting system with n-line sorting switch. *United States Patent number 3,029,413*, 6, April, 1962 1962.

[8] Kenneth E. Batcher Sherenaz W. Al-Haj Baddar.

Client Based Power Iteration Clustering Algorithm to Reduce Dimensionality in Big Data

Jayalatchumy. D[1], Thambidurai. P [2]
[1, 2] Department of CSE, PKIET, Karaikal, India

Abstract - *Clustering is a group of objects that are similar among themselves but dissimilar to objects in other clusters. Clustering large dataset is a challenging task and the need for increase in scalability and performance formulates it to use parallelism. Though the use of Big Data has become very essential, analyzing it is demanding. This paper presents the (pC-PIC) parallel Client based Power Iteration clustering algorithm based on parallel PIC originated from PIC (Power Iteration Clustering). PIC performs clustering by embedding data points in a low dimensional data derived from the similarity matrix. In this paper we have proposed a client based algorithm pC-PIC that out performs the job done by the server and reduces its execution time. The experimental results show that pC-PIC can perform well for big data. It's fast and scalable. The result also shows that the accuracy in producing the clusters is almost similar to the original algorithm. Hence the results produced by pC-PIC are fast, scalable and accurate.*

Keywords: PIC, p-PIC, pC-PIC, Big Data, Clustering.

1 Introduction

Clustering is the process of grouping a set of physical or abstract objects into classes of similar objects. Survey papers, e.g., [1][2] provide a good reference on clustering methods. Sequential clustering algorithms work well for the data size that is less than thousands of data sets. However, the data size has been growing up very fast in the past decade due to the rapid improvement of the information observation technology.

The characteristics volume, velocity and variety are referred to as big data by IBM. Big data is used to solve the challenge that doesn't fit into conventional relational database for handling them. The techniques to efficiently process and analyze became a major research issue in recent years.

One common strategy to handle the problem is to parallelize the algorithms and to execute them along with the input data on high-performance computers. Compared to many other clustering approaches, the major advantage of the graph-based approach is that the users do not need to know the number of clusters in advance. It does not require labeling data or assuming number of data clusters in advance. The major problem of the graph-based approach is that it requires large memory space and computational time while computing the graph structure [5]. The limitation comes from computing the similarity values of all pairs of the data nodes

Moreover, all the pairs must be sorted or partially sorted since the construction of the graph structure must retrieve the most similar pair of the data nodes. This step is logically sequential and thus hard to be parallelized. Unfortunately, this step is necessary and it takes most of the computational time and memory space while performing clustering. Therefore, to parallelize the graph-based approach is very challenging. One popular modern clustering algorithm is spectral clustering. Spectral clustering is a family of methods based on Eigen decompositions of affinity, dissimilarity or kernel matrices [5][7].

PIC replaces the Eigen decomposition needed by spectral clustering with matrix vector multiplications, which can reduce computational complexity. By performing clustering on several datasets it has been proved that PIC [5] is not only accurate but also fast. The PIC algorithm can handle large data's but fitting the similarity matrix into the computer's memory is

not feasible. For these reasons we move on to parallelism across different machines. Due to its efficiency and performance for data communications in distributed cluster environments, the work was done on MPI as the programming model for implementing the parallel PIC algorithm

2 Power Iteration Clustering

Spectral clustering has its own advantages over other conventional algorithms like K-means and hierarchical clustering. The use computing eigenvector is time consuming [5]. Hence PIC is designed to find pseudo-eigenvector thus it can overcome the limitation.

The effort required to compute the eigenvectors is relatively high, O(n3), where n is the number of data points. PIC [9] is not only simple but is also scalable in terms of time complexity O(n) [5]. A pseudo-eigenvector is not a member of the eigenvectors but is created linearly from them. Therefore, in the pseudo Eigen vector, if two data points lie in different clusters, their values can still be separated.

Given a dataset $X = (x_1; x_2 \ldots .x_n)$, a similarity function $s(x_i; x_j)$ is a function where $s(x_i; x_j) = s(x_j ; x_i)$ and $s >= 0$ if $i \neq j$, and $s = 0$ if $i = j$. An affinity matrix $A \in R^{nxn}$ is defined by $A_{ij} = s(x_i; x_j)$. The degree matrix D associated with A is a diagonal matrix with $d_{ii} = \sum_{ij} Aij$: A normalized affinity matrix W is defined as $D^{-1}A$. Thus the second-smallest, third-smallest,. . . , k^{th} smallest eigenvectors of L are often well-suited for clustering the graph W into k components[10].

The main steps of Power iteration clustering algorithm are described as follows [9] [5]:

1) Calculate the similarity matrix of the given graph.
2) Normalize the calculated similarity matrix of the graph, $W=D^{-1} A$.
3) Create the affinity matrix $A \in R^{n*n}$ W from the normalized matrix, obtained by calculating the similarity matrix.
4) Perform iterative matrix vector multiplication is done V^{t+1}
5) Cluster on the final vectors obtained.
6) Output the clustered vectors.

Input: A data set $x=\{x_1, x_2, \ldots x_n\}$ and normalized affinity matrix W

//Affinity matrix calculations and normalization

1. Construct the affinity matrix from the given graph, $A \in R^{n*n}$
2. Normalize the affinity matrix by dividing each element by its row sum, $W=D^{-1} A$.

//Steps for iterative matrix- vector multiplication

repeat

$V^{t+1} = (WV^t)/\|WV^t\|_1$
$\sigma^{t+1} = |V^{t+1}-V^t|$
Acceleration$=\|\sigma^{t+1}-\sigma^t\|$
Increase t
Until stopping criteria is met.

//Steps to generate initial vector
Generate initial vector, $V_o=R/\|R\|_1$ where R
is the row sum of W.
Steps for clustering
Cluster on the final vector V^t.
Output the clusters.

Fig 1: Pseudo code for PIC

The affinity matrix is,

$$\begin{bmatrix} 4 & 2 & 2 \\ 2 & 5 & 3 \\ 2 & 3 & 5 \end{bmatrix}$$

The normalized row sum for the first row is 4.9

$$W = \begin{bmatrix} 0.8 & 0.4 & 0.4 \\ 0.3 & 0.8 & 0.4 \\ 0.3 & 0.4 & 0.3 \end{bmatrix}$$

The row sum R is calculated as,

$$\begin{bmatrix} 1.5 \\ 1.6 \\ 1.0 \end{bmatrix}$$

Hence the normalized matrix is $\|R\|= 2.41$. The value of V_o is calculated as 0.414.When t=0, V^1 is obtained from the following $V^1=WV^0/\|WV^0\|$ Hence,

$$V^1 = \begin{bmatrix} 0.65 \\ 0.61 \\ 0.40 \end{bmatrix}$$

After all the necessary calculations, V^1 and V^0 after substitution produce zero, hence we conclude that the number of clusters produced is two. The experimental result of implementation of PIC algorithm for various input and various clusters generated for the given inputs are shown as graphs in the fig 2 given below.

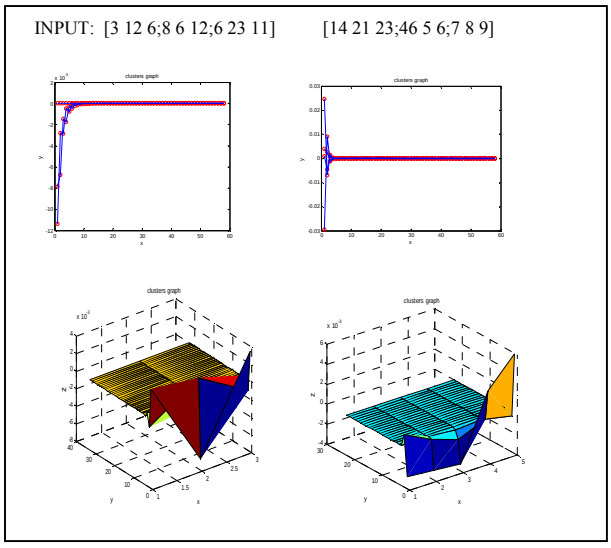


Fig 2.The graph for various inputs

3 Parallel PIC (p-PIC)

Parallelization is a method to improve performance and achieve scalability. Many techniques have been used to distribute the load over various processors. There are several different parallel programming frameworks available [12]. The message passing interface (MPI) is a message passing library interface for performing communications in parallel programming environments [12]. Because of the efficiency and performance on a distributed environment, work has been done on MPI [5] as the programming model and implemented the parallel PIC algorithm. The algorithm for parallel PIC is as follows [5] and the flowchart is shown in fig 3.

Step 1: Get the starting and end indices of cases from master processor.
Step 2: Read in the chunk of case data and also get a case broadcasted from master.
Step 3: Calculate similarity sub-matrix, A_i ,a n/p by n matrix.
Step 4: Calculate the row sum, R_i , of the sub-matrix and send it to master.
Step 5: Normalize sub-matrix by the row sum, $W_i = D_i^{-1} A_i$.
Step 6: Receive the initial vector from the master, v^{t-1}.
Step 7: Obtain sub-vector by performing matrix-vector multiplication, $v_i^t = \square W_i v^{t-1}$.
Step 8: Send the sub-vector, v_i^t , to master.

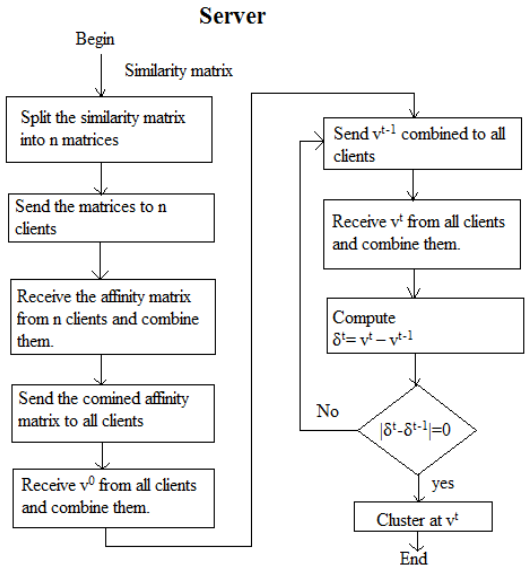


Fig 3: Flowchart for p-PIC using MPI

4 Client Based p-PIC

The time taken for transferring the data from the server to client takes much of the time for execution. Since initial vectors has to be calculated and sent to the master each time to find the vectors, it consume more time. To reduce this process time the algorithm is designed in such a way that the client takes the responsibility of handling much work reducing work of server. The algorithm of client based power iteration clustering is as follows. The master receives the data from the dataset. The data are spilt based on the number of slaves (n). On receiving the data from the master, each slave starts its work of computation. Each slave receives the data file from the master and finds the row sum .The row sum is sent back to the master. Now the master finds the initial vector which is sent to the slave. The calculation of initial vector and number of clusters is calculated and the process ends when the stopping criteria is met. The architecture for the parallel client based PIC flowchart is given below in fig 4.

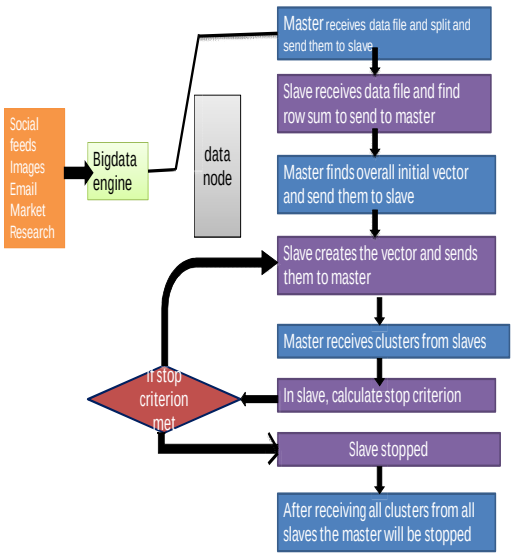


Fig 4: Working of pC-PIC algorithm

5 Experimental Results

The effectiveness of the original PIC for clustering has been discussed by Lin and Cohen [9]. The scalability of p-PIC have been shown by Weizhong Yana [5]. In this paper will focus on scalability in parallel implementation of the pC-PIC algorithm. We implemented our algorithm over a number of synthetic dataset of many records. We also created a data generator to produce a dataset used in our experiment. The size (n) of the dataset varies from 10000 to 100000 numbers of rows. We performed the experiment on local cluster. Our local cluster is HP Intel based and the number of nodes is 6. The stopping criteria for the number of clusters created are approximately equal 0. In this paper we used speed up (execution time) as the performance measure for implementing the pC-PIC.

6 Comparisons of PIC, P-PIC and Pc-PIC

We present performance results of pC-PIC in terms of speedups on different no of processors and the scalability of algorithm with different database size are found .We compare p-PIC with pC-PIC and have shown that the performance have been increased along with the scalability. Fig 5and 6 shows the time executed for various size of datasets. The data sizes are measured in MB and the time taken is calculated in milliseconds. The graph gives a comparison of PIC, p-PIC and p-PIC in MapReduce framework. Fig 7 gives the comparison of various datasets and its corresponding execution time.

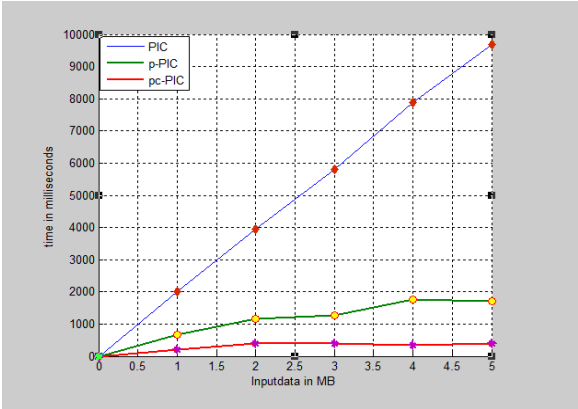


Fig 5: Comparison of Speedup for PIC, p-PIC and p-PIC in MapReduce

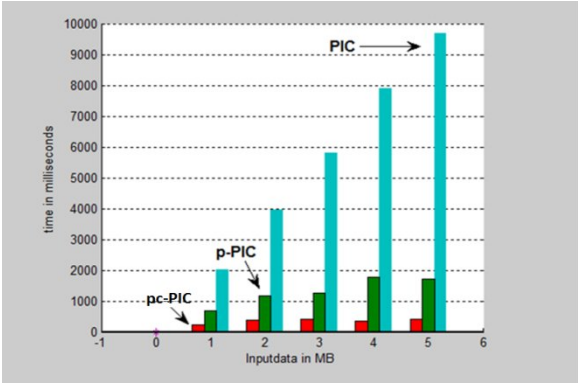


Fig 6: Data (MB) vs Execution time (ms)

Dataset Size (KB)	PIC (ms)	p-PIC (ms)	Pc-PIC (ms)
1000	2012	672	218
2000	3947	1189	385
3000	5838	1259	399
4000	7879	1760	406

Fig 7: Comparison of PIC, p-PIC and pC-PIC

7 Conclusion

In this paper we have designed a new client based algorithm for PIC namely the pC-PIC and have generated cluster for dataset of various size. The results have been compared with sequential PIC and p-PIC using MPI. The results show that the clusters formed using pC-PIC is almost same as that of the other algorithms. The performance has been increased to a greater extend by reducing the execution time. It has also been observed the performance increases along with the increase in the data size. Hence it is more efficient for high dimensional dataset.

8 Future Work

Detecting the failure node that crashes the entire system is necessary. The aim of fault tolerance system is to remove such nodes which cause failures in the system [8]. Using Hadoop the problem of fault tolerance can be avoided. As a future work we can address how node failures can be avoided using Mapreduce and can be compared with other frameworks. Hadoop is fault tolerant and it also provides a mechanism to overcome it.

9 References

[1] Jain, M. Murty, and P. Flynn, "Data clustering: A review", ACM Computing Surveys 31 (3) (1999) 264–323.

[2] Xu, R. and Wunsch, D. "Survey of clustering algorithms", IEEE Transactions on Neural Networks 16 (3) (2005).

[3] Kyuseok Shim, "MapReduce Algorithms for Big Data Analysis", Proceedings of the VLDB Endowment, Vol. 5, No. 12,Copyright 2012 VLDB Endowment 21508097/12/08.

[4] Chen,W.Y , Song, Y, Bai,H. and Lin. C, "Parallel spectral clustering in distributed systems", IEEE Transactions on Pattern Analysis and Machine Intelligence 33 (3) (2011) 568–586

[5] Weizhong Yana et.al , " P-PIC: Parallel power iteration clustering for big data", Models and Algorithms for High-Performance Distributed Data Mining. Volume 73, Issue 3, March 2013

[6] W. Zhao, H. Ma, Q. He, "Parallel K-means clustering based on MapReduce", Journal of Cloud Computing 5931 (2009) 674–679.

[7] H. Gao, J. Jiang, L. She, Y. Fu, "A new agglomerative hierarchical clustering algorithm implementation based on the map reduce framework", International Journal of Digital Content Technology and its Applications 4 (3) (2010) 95–100.

[8] Kyuseok Shim, "MapReduce Algorithms for Big Data Analysis", Proceedings of the VLDB Endowment, Vol. 5, No. 12,Copyright 2012 VLDB Endowment 21508097/12/08.

[9] Frank Lin frank, William W.," Power Iteration Clustering",International Conference on Machine Learning, Haifa, Israel, 2010.

[10] F. Lin, W.W. Cohen, Power iteration clustering, in: Proceedings of the 27th International Conference on Machine Learning, Haifa, Israel, 2010.

[11] Sakai, T. and Imiya, A. "Fast spectral clustering with random projection and sampling", Lecture Notes in Computer Science 5632 (2009) 372–384.

[12] Quinn, M.J. "Parallel Programming in C with MPI and OpenMP", McGraw-Hill,Boston, Mass, UA, 2008

SESSION

BIG DATA MANAGEMENT AND FRAMEWORKS

Chair(s)

TBA

Real-Time Data/Information Storage and Retrieval in Autonomous AI Systems

James A. Crowder, John N. Carbone
Raytheon Intelligence, Information, and Services
16800 E. Centretech Parkway, Aurora, Colorado 80011

Abstract - *Current and future space, air, and ground systems are growing in complexity and capability, creating a serious challenge to operators who monitor, maintain, and utilize systems in an ever growing network of assets [Crowder 1996]. The growing interest in autonomous systems with cognitive skills to monitor, analyze, diagnose and predict behaviors real time makes this problem even more challenging. Systems today continue to struggle with satisfying the need to obtain actionable knowledge from an ever increasing and inherently duplicative store of non-context specific, multi-disciplinary information content. Additionally, increased automation is the norm and truly autonomous systems are the growing future for atomic/subatomic exploration and within challenging environments unfriendly to the physical human condition. Simultaneously, the size, speed, and complexity of systems continue to increase rapidly to improve timely generation of actionable knowledge. However, development of valuable readily consumable knowledge density and context quality continues to improve more slowly and incrementally. New concepts, mechanisms, and implements are required to facilitate the development and competency of complex systems to be capable of autonomous operation, self-healing, and thus critical management of their knowledge economy and higher fidelity self-awareness of their real-time internal and external operational environments. Presented here are new concepts and notional architectures to solve the problem of how to take the fuzziness of information content and drive it towards context-specific topical knowledge development. We believe this is necessary to facilitate real-time cognition-based information discovery, decomposition, reduction, normalization, encoding, memory recall (knowledge construction), and most importantly enhanced/improved decision making for autonomous AI systems.*

Keywords: Artificial Intelligence, Autonomous Robotics, Real-Time Autonomous Systems, Newtonian Mechanics, Quantum Mechanics, Autonomous Decision Making.

1. Introduction

An Artificially Intelligent System (AIS), in order to be truly autonomous, must be provided with real-time cognition-based information discovery, decomposition, reduction, normalization, encoding, and memory recall (knowledge construction), all in real time, to improve decision making for autonomous robotic systems. Cognitive systems must be able to integrate information into their current Cognitive Conceptual Ontology [Crowder, Taylor, and Raskin 2012] in order to be able to "think" about, correlate and integrate the information into the overall AIS memories. When describing how science integrates with information theory, Brillouin [Brillouin 2004] defined knowledge succinctly as resulting from a certain amount of thinking and distinct from information which had no value, was the "result of choice," and was the raw material consisting of a mere collection of data. Additionally, Brillouin concluded that a hundred random sentences from a newspaper, or a line of Shakespeare, or even a theorem of Einstein have exactly the same information value. Therefore, information content has "no value" until it has been thought about and thus turned into knowledge. Decision-making is a great concern due to for handling ambiguity and the ramifications of erroneous inferences. Often there can be serious consequences when actions are taken based upon incorrect recommendations and can influence decision-making before the inaccurate inferences can be detected and/or even corrected. Underlying the data fusion domain is the challenge of creating actionable knowledge from information content harnessed from an environment of vast, exponentially growing structured and unstructured sources of rich complex interrelated cross-domain data. This is a major challenge for autonomous AI systems that must deal with ambiguity without the advantage of operator-based assistance.

Dourish [Dourish 2004a] expressed that the scientific community has debated definitions of context and it's uses for many years. He discussed two notions of context, technical, for conceptualizing human action relationship between the action and the system, and social science, and reported that "ideas need to be understood in the intellectual frames that give them meaning." Hence, he described features of the environment where activity takes place [Dourish 2004b]. Alternatively, Torralba [Torralba 2003] derived context based object recognition from real-world from scenes, described that one form of performing the task was to define the 'context' of an object in a scene was in terms of other previously recognized objects and concluded, that there exists a strong relationship between the environment and the objects found within, and that increased evidence exists of early human perception of contextual information. Dey [Dey 2001] presented a Context Toolkit architecture that supported the building of more optimal context-aware applications, because, he argued, that context was a poorly used resource of

information in computing environments and that context was information which must be used to characterize the collection of states or as he called it the "situation abstraction" of a person, place or object relevant to the interaction between a user and the application. Similarly, when describing a conceptual framework for context–aware systems, Coutaz et al. [Coutaz, Crowley, Dobson, and Garlan 2005] concluded that context informs recognition and mapping by providing a structured, unified view of the world in which a system operates. The authors provided a framework with an ontological foundation, an architectural foundation, and an approach to adaptation, which they professed, all scale alongside the richness of the environment. It was concluded that context was critical in the understanding and development of information systems. Winograd [Winograd 2001] noted that intention could only be determined through inferences based on context. Hong and Landay [Hong and Landay 2001] described context as knowing the answers to the "W" questions (e.g. Where are the movie theaters?). Similarly, Howard and Qusibaty [Howard and Quisibaty 2004] described context for decision making using the interrogatory 5WH model (who, what, when, where, why and how). Lastly, Ejigu et al. [Ejigu, et. al. 2008] presented a collaborative context aware service platform, based upon a developed hybrid context management model. The goal was to sense context during execution along with internal state and user interactions using context as a function of collecting, organizing, storing, presenting and representing hierarchies, relations, axioms and metadata.

These discussions outlines the need for an AIS cognitive framework which can analyze and process knowledge and context [Crowder and Carbone 2012]; representing context in a knowledge management framework comprising processes, collection, preprocessing, integration, modeling and representation, enabling the transition from data, information and knowledge to new knowledge. Described here is a cognitive processing framework and memory encoding and storage methodology for capturing contextual knowledge as well as decision making in a knowledge repository that corresponded to a specific context instance.

2. AIS Memory Management

2.1 Sensory Memories

The Sensory Memory within the AIS memory system are those memory registers where raw, unprocessed information ingested via AIS environmental sensors and are buffered to begin initial processing. The *AIS* sensory memory system has a large capacity to accommodate large quantities of possibly disparate and diverse information from a variety of sources [Crowder 2010]. And although it has a large capacity, it has a short duration. The information that is buffered in this sensory memory must be sorted, categorized, turned into information fragments, metadata, contextual threads, and attributes (including emotional attributes) and then sent on to the working memory (Short-Term Memory) for initial cognitive processing. This cognitive processing is known as Recombinant Knowledge Assimilation (RNA), where raw information content is discovered from the information domain, decomposed & reduced, compared, contrasted, and associated into new relationship threads within a temporary working knowledge domain and subsequently normalized into pedigree within the knowledge domain for future use [Carbone 2010]. Hence, based upon the information gathered in initial Sensory Memory processing, Cognitive Perceptrons, manifested as Intelligence information Software Agents (ISAs), are spawned, as in relative size swarms, to create initial "thoughts" about the data. Subsequently, hypotheses are generated by the ISAs. The thought process information, along with the ISA sensory information are then sent to a working memory region which will alert the artificial cognition processes within the AIS to begin processing. [Crowder and Friess 2010a&b]. Figure 1 illustrates the Sensory Memory Lower Ontology.

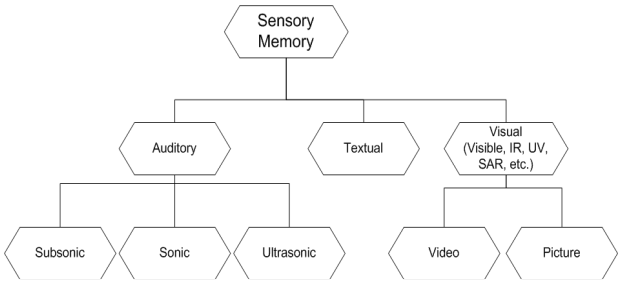


Figure 1 - Sensory Memory Lower Ontology

2.2 Short-Term Artificial Memories

Short-Term or "Working" memory within the AIS is where new information is transitionally stored in a Temporary Knowledge Domain [Carbone 2010] while it is being processed into new Knowledge. This follows the paradigm that information content has no value until it is thought about [Brillouin 2004]. Short-Term memory is where most of the reasoning within the AIS happens. Short-Term memory (STM) provides a major functionality, called "rehearsals" that allows the AIS to continually refresh, or rehearse, the Short-Term memories while they are being processed and reasoned about, so that memories do not degrade until they can be sent on to Long-Term Memory and acted upon by the artificial consciousness processes within the AIS's cognitive framework [Crowder and Carbone 2011a]. It should be noted that Short-Term memory is much smaller in relative space needed to

process information content as compared to long term memory. Short-Term memory should be perceived not necessarily as a physical location, as in the human brain, but rather as a rapid and continuous processing of information content relative to a specific AIS directive or current undertaking. One must remember that the Short-Term memory which includes all external and internal sensory inputs will trigger a rehearsal if the AIS discovers a relationship to either a previously interred piece of information content in short or long term memory. Figure 2 illustrates the Short-Term Memory Lower Ontology for the AIS.

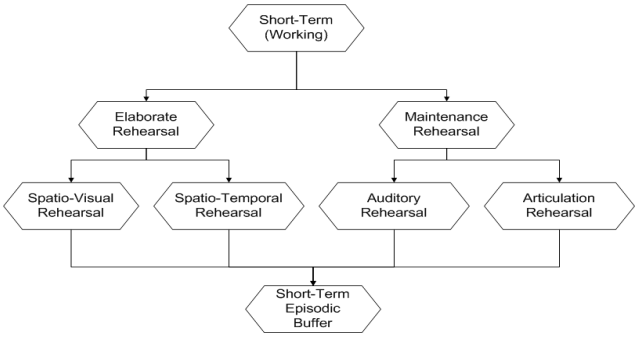


Figure 2 - AIS Short-Term Memory Lower Ontology

2.3 Long-Term Artificial Memories

Long-Term Memory (LTM), in the simplest sense, is the permanent Knowledge Domain where we assimilate our memories [Carbone 2010]. If information we take in through our senses doesn't make it to LTM, we can't and don't "remember" it. Information that is processed in the STM makes it to LTM through the process of rehearsal, processing, encoding, and then association with other memories. In the brain, memories are not stored in files, or in a database. Memories, in fact, or not stored as whole memories at all, but instead are stored as information fragments. The process of recall, or remembering, constructs memories from these information fragments that are stored in various regions of the brain, depending on the type of information. In order to create our AIS in a way that mimics human reasoning, we follow the process of storing information fragments and their respective encoding in different ways, depending on the type and context of the information, as discussed above. Each simple discrete fragment of objective Knowledge includes an n-dimensional set of quantum mechanics based mathematical relationships to other fragments/objects bundled in the form of eigenvector optimized Knowledge Relativity Threads (KRT) [Carbone 2010] [Carbone and Crowder, 2011].These KRT bundles include closeness, and relative importance value among others. This importance is tightly coupled, per the math, to the AIS emotional storage as a function of desire or need, as described in Figure 3, where the LTM Lower Ontology is illustrated. There are three main types of LTM [Crowder 2010a]; Explicit or Declarative memories, Implicit memories, and Emotional memories.

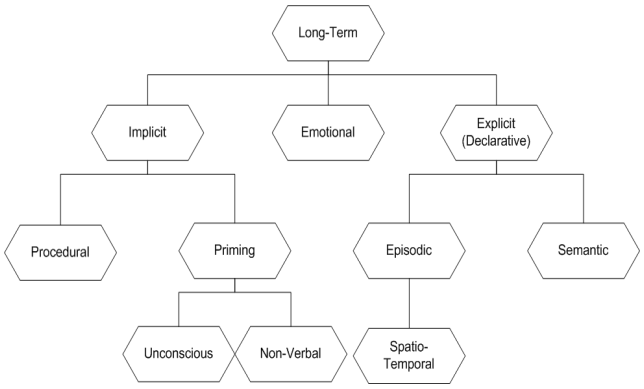


Figure 3 - Artificial Long-Term Memory Lower Ontology

3. Artificial Memory Processing and Encoding

3.1 Short-Term Artificial Memory Processing

In the human brain, STM corresponds to that area of memory associated with active consciousness, and is where most of the cognitive processing takes place. It is also a temporary storage and requires rehearsal to keep it fresh until it is compiled into Long-Term Memory (LTM). In the AIS, the memory system does not decay over time, however, the notion of "memory refresh" or rehearsal is still a valid concept as the Artificial Cognitive Processes work on this information. However, the notion of rehearsal means keeping track of "versions" of STM as it is being processed and evaluated by the artificial cognition algorithms, which is why it appears to feedback onto itself (rehearsal loop). This is illustrated in Figure 4, the AIS STM Attention Loop. There are three distinct processes that are handled within the STM that determine where information is transferred after cognitive processing (Crowder 2010a).

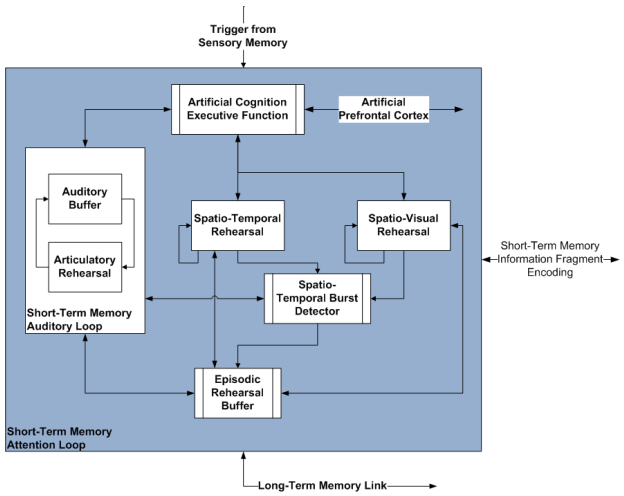


Figure 4 - Short-Term Artificial Memory Attention Loop

This processing is shown in Figure 5. The Artificial STM processing steps are:

- **Information Fragment Selection**: this involves filtering the incoming information from the AIS Artificial Preconscious Buffers into separable information fragments and then determining which information fragments are relevant to be further processed, stored, and acted on by the cognitive processes of the AIS as a whole. Once information fragments are created from the incoming sensory information, they are analyzed and encoded with initial topical information, as well as Metadata attributes that allow the cognitive processes to organize and integrate the incoming information fragments into the AIS's overall LTM system. The Information Fragment encoding creates a small, Information Fragment Cognitive Map that will be used for the organization and integration functions.
- **Information Fragment Organization**: these processes within the Artificial Cognition framework create additional attributes within the Information Fragment Cognitive Map that allow it to be organized for integration into the overall AIS LTM framework. These attributes have to do with how the information will be represented in LTM and determine how these memory fragments will be used to construct new memories, or recall, memories later by as needed by the AIS, using Knowledge Relativity Thread representation to capture the context of the Information Fragment and each of it's qualitative relationships to other fragments and/or bundles of fragments already created.
- **Information Fragment Integration**: Once the Information Fragments within the STM have been KRT encoded, they are compared, associated, and attached to larger, Topical Cognitive Maps that represent relevant subject or topics within the AIS's LTM system. Once these Information Fragment Cognitive Maps have been integrated, processed, and reasoned about, including emotional triggers or emotional memory information, they are sent on to both the LTM system, as well as the AIS Artificial Prefrontal Cortex to determine if actions are required.

One of the major functions within the STM Attention Loop is the Spatio-Temporal Burst Detector. Within these processes, Binary Information Fragments (BIFs) are ordered in terms of their spatial and temporal characteristics. Spatial[1] and Temporal transitions states are measured in terms of mean, mode, median, velocity, and acceleration and are correlated between their spatial and temporal characteristics and measurements. Rather than just looking at frequencies of occurrence within information, we also look for rapid increases in temporal or spatial characteristics that may trigger an inference or emotional response from the cognitive processes. It is not that an AIS system processes information content differently based upon how rapidly content is ingested, it is simply that an AIS must be able to recognize instances when information content might seem out of place within the context of a situation: e.g., a single speeding car within a crowd of hundreds of other cars. An AIS, not only optimizes its processing on the supply side of the knowledge economy, but has to recognize, infer, and avoid distraction on what focuses the demand side of its knowledge economy places upon operations and directives. State transition bursts are ranked according to their weighting (velocity and acceleration), together with the associated temporal and/or spatial characteristics, and any triggers that might have resulted from this burst processing (LaBar and Cabeza 2006). This Burst Detection and processing may help to identify relevant topics, concepts, or inferences that may need further processing by the Artificial Prefrontal Cortex and/or Cognitive Consciousness processes (Crowder and Friess 2011 a&b).

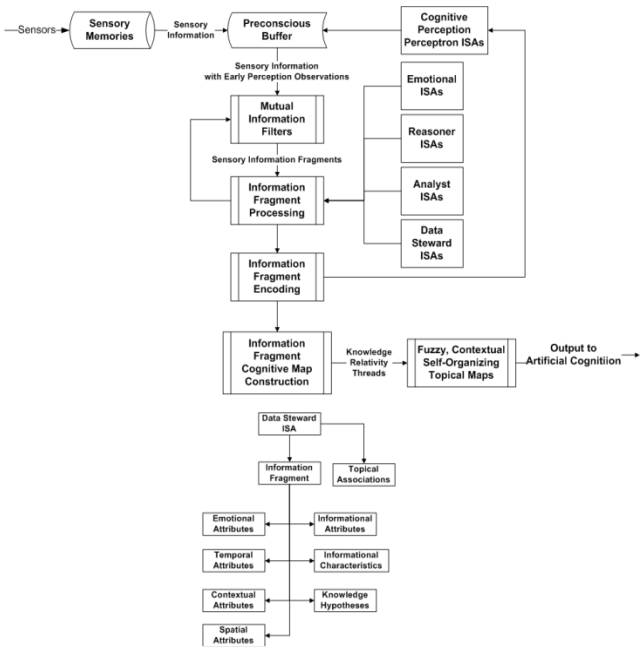


Figure 5 - AIS Information Fragment Encoding

Once processing within the STM system has completed and all memories are encoded, mapped to topical associations, and their contexts captured, their knowledge relativity thread bundled representations are created and are sent on to the Cognitive Processing engine Memories

[1] Spatial in this reference can be geographically (either 2-D or 3-D), cyber-locations, or other characteristics that may be considered "spatial" references or characteristics.

that are deemed relevant to "remember" are integrated into the Long Term Memory system.

3.2 Long-Term Artificial Memory Processing

The overall AIS High-Level memory architecture is shown in Figure 6. The one thing of note is the connection between Emotional memories and both Explicit and Implicit memories. Emotional Memory carries both Explicit and Implicit characteristics.

Explicit or Declarative Memory is utilized for storage of "conscious" memories or "conscious thoughts." Explicit memory carries those information fragments that are utilized to create what most people would "think of" when they envision a memory. Explicit memory stores things, i.e., objects, and events, things that are experienced in the person's environment. Information fragments stored in Explicit Memory are normally stored in association with other information fragments that relate in some fashion. The more meaningful the association, the stronger the memory and the easier the memory is to construct/recall when you choose to [Yang and Raine 2009]. In our AIS, Explicit Memory is divided into different regions, depending on the type or source of information. This division of regions is because different types of information fragments within the AIS memories are encoded and represented differently, each with its own characteristics that make it easier to construct/recall the memories later when the AIS needs the memories. In the AIS LTM, we utilize Fuzzy, Self-Organizing, Contextual Topical Maps to associate currently processed Information Fragments from the STM with memories stored in the LTM (Crowder, Scally, and Bonato 2011).

LTM information fragments are not stored in databases or as files, but encoded and stored as a triple helix of continuously recombinant binary neural fiber threads that represent:

- The Binary Information Fragment (BIF) object along with the BIF Binary Attribute Objects (BAOs).
- The BIF Recombinant Knowledge Assimilation (RNA) Binary Relativity Objects.
- The Binary Security Encryption Threads.

Built into the RNA Binary Relativity Objects are Binary Memory Reconstruction Objects, based on the type and source of BIF, that allow memories to be constructed for recall purposes.

There are several types of Binary Memory Reconstruction Objects, they are:

- Spectral Eigenvectors that allow memory reconstruction using Implicit and Biographical LTM BIFs
- Polynomial Eigenvectors that allow memory reconstruction using Episodic LTM BIFs
- Socio-Synthetic Autonomic Nervous System Arousal State Vectors that allow memory reconstruction using Emotional LTM BIFs
- Temporal Confluence and Spatial Resonance coefficients that allow memory reconstruction using Spatio-Temporal Episodic LTM BIFs
- Knowledge Relativity and Contextual Gravitation coefficients that allow memory reconstruction using Semantic LTM BIFs

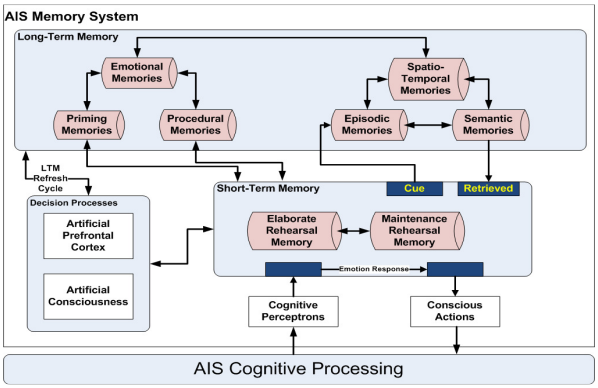


Figure 6 - High-Level Artificial Memory Architecture

4. Conclusions and Discussion

We believe this framework provides an artificially intelligent architecture and methodology that will allow autonomous operations. The use of the ISA architecture, combined with the cognitive structures described here, have the potential to radically change and enhance autonomous systems in the future. More work is needed to refine the agent technologies and learning sets, but we feel this has much potential.

7. References

1. Brillouin, L. 2004. Science and information theory. *Dover*.
2. Newell, A. 2003. Unified Theories of Cognition. *Cambridge MA: Harvard University Press*.
3. Crowder, J. A., 1996. X33/RLV Autonomous Reusable Launch System Architecture. *NASA Report 96-RLF-1.4.5.5-005*, Lockheed Martin, Littleton, CO.
4. Crowder, J. A. 2010a. The Continuously Recombinant Genetic, Neural Fiber Network. *Proceedings of the AIAA Infotech@Aerospace-2010*, Atlanta, GA.
5. Crowder, J. A. 2010b. Anti-Terrorism Learning Advisory System (ATLAS): Operative Intelligent Information Agents for Intelligence Processing. *Proceedings of the AIAA Infotech@Aerospace-2010*, Atlanta, GA.

6. Crowder, J. A., 2010c. Flexible Object Architectures for Hybrid Neural Processing Systems. *Proceedings of the 11th International Conference on Artificial Intelligence*, Las Vegas, NV.
7. Crowder, J. A., and Carbone, J. 2011a. Recombinant Knowledge Relativity Threads for Contextual Knowledge Storage. *Proceedings of the 12th International Conference on Artificial Intelligence*, Las Vegas, NV.
8. Crowder, J. A., and Carbone, J. 2011b. Transdisciplinary Synthesis and Cognition Frameworks. *Proceedings of the Society for Design and Process Science Conference 2011*, Jeju Island, South Korea.
9. Crowder, J. and Friess S. 2012. Artificial Psychology: The Psychology of AI. *Proceedings of the 3rd International Multi-Conference on Complexity, Informatics, and Cybernetics*, Orlando, FL
10. Crowder, J. 2012a. Cognitive System Management: The Polymorphic, Evolutionary, Neural Learning and Processing Environment (PENLPE). *Proceedings for the AIAA Infotech@Aerospace 2012 Conference*, Garden Grove, CA.
11. Crowder, J. 2012b. The Artificial Cognitive Neural Framework. *Proceedings for the AIAA Infotech@Aerospace 2012 Conference*, Garden Grove, CA.
12. Crowder, J., Raskin, V., and Taylor, J. 2012. Autonomous Creation and Detection of Procedural Memory Scripts. *Proceedings of the 13th Annual International Conference on Artificial Intelligence*, Las Vegas, NV.
13. Raskin, V., Taylor, J. M., & Hempelmann, C. F. 2010. Ontological semantic technology for detecting insider threat and social engineering. *New Security Paradigms Workshop*, Concord, MA.
14. Rosenblatt F. 1962. Principles of Neurodynamics. *Spartan Books*.
15. Scally, L., Bonato M., and Crowder, J. 2011. Learning agents for Autonomous Space Asset Management. *Proceedings of the Advanced Maui Optical and Space Surveillance Technologies Conference*, Maui, HI.
16. Taylor, J. M., & Raskin , V. 2011. Understanding the unknown: Unattested input processing in natural language, *FUZZ-IEEE Conference*, Taipei, Taiwan.
17. Dourish, P. 2004a. Where the action is: The foundations of embodied interaction. *The MIT Press.*
18. Dourish, P. 2004b. What we talk about when we talk about context. *Personal and ubiquitous computing, vol. 8, pp. 19-30.*
19. Torralba, A. 2003. Contextual priming for object detection. *International Journal of Computer Vision, vol. 53, pp. 169-191.*
20. Dey, A. 2001. Understanding and using context. *Personal and ubiquitous computing, vol. 5, pp. 4-7.*
21. Coutaz, J., Crowley, J., Dobson, S., and Garlan, D. 2005. Context is key. *Communications of the ACM, vol. 48, pp. 53.*
22. Winograd, T. 2001. Architectures for context. *Human-Computer Interaction, vol. 16, pp. 401-419.*
23. Hong, J. and Landay, J. 2001. An infrastructure approach to context-aware computing. *Human-Computer Interaction, vol. 16, pp. 287-303.*
24. Howard, N. and Qusaibaty, A. 2004. Network-centric information policy. *Proceedings of the Second International Conference on Informatics and Systems.*
25. Ejigu, D., Scuturici, M., and Brunie, L. 2008. Hybrid approach to collaborative context-aware service platform for pervasive computing. *Journal of Computers, vol. 3, pp. 40*

Transparency and Efficiency in Grid Computing for Big Data

Paul L. Bergstein

Dept. of Computer and Information Science
University of Massachusetts Dartmouth
Dartmouth, MA
pbergstein@umassd.edu

Abstract *– Many big data applications need to process large amounts of data stored in heterogeneous databases which are distributed across multiple grid nodes. One of the key issues in developing such applications is transparency, i.e. users and applications should not need to be aware of the data heterogeneity or distribution details. Another important issue is efficiency, especially optimization of queries involving distributed joins.*

We have previously described the development of a data mediation service which provides the transparency that enables users and applications to pull data from a foreign data source without any knowledge of its actual structure or semantics. The mediator translates a query written against a well known (local) data source into a query against the foreign data source, executes the query, and then translates the data into the local format. In this scenario, all of the data retrieved is obtained from a single source.

In this paper, we describe how our mediator provides transparency and efficient join processing in a grid environment where a single query may require combining data from multiple distributed sources.

Keywords: Big data, data mediation, data integration, grid-dbms, distributed join processing.

1. Introduction

In the current age of big data, many organizations need to efficiently process large amounts of heterogeneous distributed data. The problems are particularly difficult when the nodes have been developed separately, in which case the heterogeneity of the data creates major obstacles to effective processing. Conflicts may exist in both the structure and the semantics of the data involved. Furthermore, the structure and semantics of a data source may change over time.

The data mediation approach to data interoperability relies on a common ontology that can be used to describe the structure and semantics of each data source. A data mediator uses these descriptions to resolve structural and semantic inconsistencies between nodes exchanging information. In a variation of this approach, a shared view is created, and the mediator translates queries written against the shared view. In either case, mediation has the advantages that there is no need to agree on standard formats, the metadata is made explicit (so it may be reused), and translations only occur where the structure or semantics between two systems differ. The last point is particularly important for efficiency when large amounts of data are involved.

Our big data mediation service (BDMS) uses a layered architecture as shown in Figure 1. In the sections 2 and 3 we will briefly describe the basic operation of the mediator in providing location and structure transparency. Then, in section 4, we will focus on our recent work on efficient mediation in a big data environment.

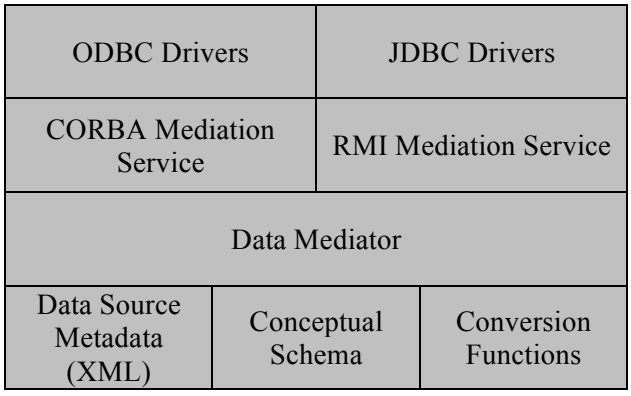


Figure 1

2. Background

Our data mediator was originally based on the following scenario: Suppose a user who knows the schema of only their local database, System A, wishes to retrieve information from a foreign database, System

B. They write a query against the schema of System A, but indicate that they would like to use System B as the data source. The mediator translates the query against System A into one or more queries against System B, executes the queries, and translates the results into the local format of System A.

In our current work we also consider a slightly different scenario: Suppose an application using ODBC or JDBC has been written to use a particular data source, System A, but we now want to use a different source, System B, with a different structure or semantics. We accomplish the change simply by plugging in our Mediator ODBC or JDBC driver in place of the System A driver. The application can now use the new data source without rewriting any code or queries. Notice that simply plugging in the System B driver would only work if Systems A and B had identical structure and semantics, otherwise mediation is required.

2.1 Conceptual Schema

In our implementation, the common ontology is expressed as a shared conceptual schema, which includes both ordinary classes (e.g. University, Student) and domain classes (e.g. Money, Date). The attributes of ordinary classes have domain classes as their types. For example, Student might have an attribute called graduation-date with type Date. For each domain class, we specify subclasses (sub-domains) for the known representations. When a new data source is registered with the mediator, it will typically be necessary to add sub-domains for data representations that are unique to that data source.

The conceptual schema is never populated in our system. It is used only as a reference for defining the structure and semantics of the actual data sources. In particular, the conceptual schema is *not* used as an intermediate data representation when transferring data from one source to another. Instead, the mediator synthesizes a plan for direct conversion between the data sources based on their structures and semantics as defined by their individual mappings to the conceptual schema.

2.2 Conversion Functions

The mediator uses a repository of functions for converting between representations within a domain. In our (java) implementation all conversion functions have the same interface. They take a java Properties object as parameter, and return a Properties object as the result, so they naturally support many-to-many mappings. For example, a position might be specified using a Properties object with latitude and longitude attributes, or (using Universal Transverse Mercators) with zone, easting, and northing attributes. In this case we have a two-to-three mapping. The repository is implemented as a java class with methods for each conversion function. The repository uses java introspection to search for suitable conversion functions.

2.3 Metadata

In order to register a data source with the mediator, a description of the data source (its metadata) must be supplied in XML format. For each data source there is a separate XML file prepared by someone familiar with that data source. Currently, the XML files are prepared manually, but we plan to develop tools to help generate these files. The metadata includes information required to connect to the data source as well as mappings to the conceptual schema that define the structure and semantics of the data source. We use XML[1] to map data elements of real databases onto attributes of ordinary classes in the conceptual schema. Each mapping to an attribute of an ordinary class includes the subdomain of the data element.

In the simplest case each data element of System A corresponds one-to-one with an element of the conceptual schema, which in turn corresponds one-to-one with an element of System B. If the conceptual schema contains an ordinary class called Employee with a salary attribute of type Salary, and System A has a Worker relation with a pay-rate attribute, then the XML file for System A would map Worker/pay-rate to Employee/salary and it would also map pay-rate to one of the subdomains of Salary such as Annual/USDollars or Monthly/Euros[2]. Similar mappings from System B provide the mediator with the information needed for translation.

Mappings between the conceptual schema and an actual database are not always one-to-one. Suppose that in the conceptual schema Professor's have a phone-number attribute of type PhoneNumber, but in the actual database Instructor's have area-code, exchange, and extension attributes. For the mediator to work, the PhoneNumber domain class must have a subdomain, say ACEE, for the area-code/exchange/extension representation of phone numbers, with attributes corresponding to the three parts of a phone number. Each of the area-code,

[1] For brevity, in this paper we mostly describe mappings without showing the XML syntax since the XML is trivial but verbose.

[2] In theory, the issues of currency units and frequency of payment should be separate, but we combine them for the sake of simplicity in our implementation, in order to focus on more interesting concerns.

exchange, and extension attributes is mapped to the Professor/phone-number attribute (and also to the appropriate attribute of the ACEE subdomain). The mapping (from actual to conceptual) is many-to-one.

If another database uses the same representation of phone numbers, so we have mappings like:

A: (code, exchg, ext) → phone-number
B: (area, exg, extension) → phone-number

then the translation will not use conversion functions (even if the data elements have different names). In other cases, such as:

A: (latitude, longitude) → position
B: (zone, easting, northing) → position

a conversion function is required.

Sometimes a data source isn't a very good match for the conceptual schema. This is likely to happen, for example, when a new data source is added after the conceptual schema has been completed. Consider, for example, a conceptual schema that has entity classes for full-time students and part-time students, and a data source with graduate students and undergraduate students. In this case we map attributes, e.g. gpa, from both graduate and undergraduate students to attributes of both full-time and part-time students. Additionally, we supply conditions that determine, for example, which graduate students are part-time and which are full-time. These conditional mappings [5] are specified in both directions (to and from the conceptual schema).

2.4 Data Mediator

The data mediator manages the conversion function repository, the conceptual schema, and the data source metadata. It is responsible for synthesizing query and translation plans. When a query against the schema of System A is executed using System B as the data source, the mediator translates the query against System A into one or more queries against System B, executes the queries, and translates the results into the local format of System A. The details of our algorithm are beyond the scope of this paper, but will be reported elsewhere.

The mediator is implemented entirely in Java and uses JDBC to access the desired data source. Therefore, the mediator can be used to exchange data between any data sources that have JDBC drivers available, including most relational databases, all ODBC data sources (via a JDBC/ODBC bridge driver), and XML data (using an available XML JDBC driver).

3. Query Mediation

In this section we describe the mediator's processing of simple queries written against the schema of a well known (local) data source when the actual data resides in a different (foreign) data source. We start by considering simple queries consisting of only *select* and *from* clauses. In the next section we will consider the more complex issues of processing the *where* clause. For our examples we will use the local and foreign schemas for airplane data in Figure 2. For simplicity, we have not shown the shared conceptual schema.

Local schema:
Airplanes (aid, latitude, longitude, fuel_capacity, range, wingspan)

Foreign schema:
Aircraft (craftId, zone, easting, northing, fuel_tank_size, cruising_range, wingspan)

Figure 2

3.1 Select Clause Translation

The select clause is translated by replacing the name of each data element in the list with the data element(s) from the foreign data source that map to the same attribute(s) in the shared conceptual schema.

In the simplest case, the local element maps to a single attribute in the shared schema which in turn maps to a single element of the foreign data source, and the replacement mapping between the local and foreign data elements inferred by the mediator is one-to-one. In our example, the mediator would infer a one-to-one replacement of fuel_capacity with fuel_tank_size wherever fuel_capacity occurs in the select clause of the original query.

However, in general, the inferred replacements are one-to-many both because the local element may map to many attributes in the conceptual schema, and because each attribute of the conceptual schema may map to many elements of the foreign data source. For example, since the local attribute *latitude* (along with longitude) maps to the concept of position, and the foreign attributes *zone*, *easting*, and *northing* also map to the concept of position, *latitude* would be replaced with *zone*, *easting*, and *northing* when the query is translated. This one-to-three replacement is correct since the mediator needs all three UTM attributes to calculate a latitude. Note that if the select clause of the original query contained both latitude and longitude, the mediator would infer a one-to-three mapping for

each of them. In a subsequent step, the mediator eliminates requests for duplicate columns.

3.2 From Clause Translation

The table names of the from clause are translated in a manner similar to the columns in the select clause. A single table in the where clause of the original query may map to multiple tables in the conceptual schema and each of those may map to multiple tables in the foreign schema.

In this case where the inferred replacement is one-to-many, there will be a separate query generated for each replacement. For example, if the foreign data source had its aircraft data split into two tables, say Jets and Propeller Aircraft, a single query on the Airplanes table of the local data source would result in two separate queries in the foreign data source – one selecting from Jets and one from Propeller Aircraft. The mediator would execute both queries and combine the results.

The other complication is that table mappings may be conditional [5]. This would come into play if we switched the local and foreign data sources for our example. In this case the mediator would replace Jets with Airplanes in the from clause, but not all airplanes are jets.

When mappings between a data source and the conceptual schema are conditional, the conditions are specified as part of the mapping. The mediator adds the appropriate mapping conditions to the where clause of the original query. The *to* conditions of the foreign schema mapping (specifying which foreign entities map *to* a conceptual class) and the *from* conditions of the local schema (specifying which conceptual entities map *from* the conceptual class) are added to the where clause of the query.

The *to* conditions are already written in terms of the foreign data source and don't require translation. The *from* conditions of the local schema, however, must be translated before the query can be executed in the foreign data source. The where clause processing is discussed in section 4.

3.3 Data Translation

After the translated queries have been executed in the foreign data source, the results must be translated into the format expected in the local data source. If there was a one-to-one replacement of an attribute in the select clause with a corresponding attribute from the foreign data source in the same format, no conversion is necessary. Otherwise, a conversion function from the mediator's repository is used. The values retrieved from the foreign data source are packaged as a java Properties object, passed to the appropriate conversion function, and the desired value is then extracted from the returned Properties object. For example, if latitude in the original query was replaced by zone, easting, and northing in the translated query, these three values from each row would be packaged as a Properties object and the latitude value would be extracted from the new Properties object with values for latitude and longitude returned from the conversion function.

3.4 Where Clause Processing

The central problem in where clause processing is to translate conditions involving data elements of the local schema into conditions that can be specified against the foreign schema. The simplest situation is where the local and foreign data elements are in the same format and correspond one-to-one. For example, if the where clause contains the condition *range > 1000,* and Airplanes range and Aircraft cruising_range are in the same format (units, scale, etc.), the mediator can simply replace *range* with *cruising_range*.

The next simplest situation is where, for example, range and cruising_range correspond one-to-one but are in different formats. If range is in kilometers and cruising_range is in miles, the mediator can apply a conversion function to the constant to generate the condition *cruising_range > 621.37.* The mediator can also modify conditions by applying operators to attributes, e.g. replacing expression *range* with *cruising_range * 0.62137*, although there are few cases in practice where this is useful.

Unfortunately, conditions involving attributes that do not map one-to-one are much more difficult to translate. Consider, for example, translating the condition *latitude > 40* into terms of zone, easting, and northing. While a human with adequate understanding of the two positioning systems could produce a translation, our mediator cannot.

In our early implementations we attempted to translate all where clause conditions and the mediator would throw an exception when presented with queries it could not handle. Once we realized that some where clause conditions could never be translated efficiently, we tried a radically different approach. In this new approach we eliminated the where clause altogether before executing the query in the foreign data source. After the data was returned the mediator applied the where clause to each data tuple as it was translated to the format of the local schema. By applying the where clause conditions to the translated data, it was not necessary to translate the conditions.

While this approach worked, it has a major drawback. Since the where clause is evaluated in the mediator, rather than the foreign data source, potentially large quantities of data that are not part of the final result must be brought across the network into the mediator.

Another suggestion was to implement conversion functions in the actual data sources. In this case the condition *latitude > 40* would be translated to *conv(zone, easting, northing) > 40* where *conv* is a conversion function defined in the foreign data source. However, this approach also has major drawbacks. First, not all data sources support this kind of function. More importantly, the approach would not scale. One of the important features of the mediation approach is that each data source is mapped only to the conceptual schema. Supplying each data source with conversion functions for every data element of every other data source is not realistic.

Our current approach is a compromise between the extremes of translating all conditions or eliminating the where clause entirely. In the most recent approach the mediator starts by rewriting the where clause in conjunctive normal form (CNF). The conjuncts can then be applied independently in sequential fashion. The conjuncts are partitioned into translatable and untranslatable groups. As many conjuncts as possible are translated and added to the where clause of the translated query for execution in the foreign data source, thereby minimizing the network traffic. The untranslatable conjuncts are applied in the mediator as the data returned from the foreign data source is translated into the format of the local data source.

Consider the query:

select aid from airplanes
where (latitude > 40 AND wingspan > 20)
OR (range > 2000 AND fuel_capacity > 500)

The mediator will start by rewriting the where clause conditions in CNF as:

(latitude > 40 OR range > 2000) AND
(latitude > 40 OR fuel_capacity > 500) AND
(wingspan > 20 OR range > 2000) AND
(wingspan > 20 OR fuel_capacity > 500)

The first two conjuncts contain the condition on latitude which cannot be translated so they will be applied in the mediator. The last two, however, are easily translated by replacing wingspan, range, and fuel capacity with the corresponding attribute names from the foreign data source, and converting the constant values into to the appropriate units. The last two conditions are translated and applied in the foreign data source to eliminate unnecessary network traffic.

4. Grid Mediation for Big Data

Aloisio et. al. [15] have identified seven basic requirements that a Grid-DBMS must provide: security, transparency, easiness, robustness, efficiency, dynamicity, and intelligence. They further identify five forms of transparency which must be addressed in a Grid-DBMS: physical data location, network, data replication, data fragmentation, and DBMS heterogeneity. While they do not suggest a specific grid middleware, our mediator is well suited for adaptation to meet the identified requirements. In this section, we describe our initial efforts to adapt the mediator to a grid environment for big data, focusing primarily on transparency and efficiency issues.

4.1 Transparency

When used in a grid environment, the mediator treats the nodes as a single distributed data source rather than a collection of alternative sources of the same information. Also, the mediator may use additional metadata to identify replicated nodes. Otherwise, the basic operation is essentially the same. The nodes are mapped onto a conceptual schema exactly as before. Users can customize their view of the Grid-DBMS by mapping an "actual" schema onto the conceptual schema. This is exactly as before, except that the actual schema is not populated. Clients write queries in terms of their "view" and the mediator performs the necessary translations.

The mediator service was designed from the beginning to hide details of physical data locations, network issues, and database heterogeneity from clients. These aspects of the mediator are unmodified when used as middleware in a big data grid environment.

The mediator was also designed to handle translation between data sources that use different partitioning schemes. It handles both horizontal partitioning (e.g. Aircraft into Jets and Propeller Planes) and vertical partitioning (e.g. Projects into ProjectFinancials and ProjectSchedules). Once again, the mediator functionality can be used unmodified in a Grid system. Although vertical partitioning of data will improve the efficiency of some queries, it may necessitate additional distributed join operations for others. We take a unique approach to distributed joins, which is discussed in the next section.

4.2 Efficient Implementation of Transparency

Our efforts in the area of efficiency are currently focused on minimizing the overhead associated with

providing transparency, and on efficiently processing distributed joins. The mediation approach to transparency is inherently more efficient than the commonly employed interlingua approach. In the interlingua approach data is converted from its source format into a common intermediate form and then from the intermediate form to its target format.

In this approach, every data element is translated to and from the intermediate form even if the source and target formats for that element are identical. The mediation approach avoids this inefficiency by performing translations only where it is necessary. The mediator synthesizes a plan for direct conversion between the data source and target without involving an intermediate representation. Data elements that have identical source and target representations pass through the mediator without translation.

Of course, there is additional runtime overhead for plan synthesis compared to the interlingua approach. However, in a big data environment, the planning cost is insignificant compared to the translation cost since the cost of planning is independent of the amount of data to be translated. Furthermore, the big data mediator may cache plans for commonly executed queries.

4.2 Efficient Join Processing

Our approach to distributed join processing is based on the same basic idea as semi-joins. We try to reduce the network traffic as much as possible at the expense of increased local processing.

Ordinarily, we think of join conditions involving both of the relations to be joined. However, the conditions in the *on* clause of an SQL theta join may contain arbitrary conditions. Furthermore, joins are frequently followed by selections (i.e. *where* clause). We start by combining the join conditions with conditions from the selection (if there is one) according to the following rewriting rule:

$$\sigma_{\Theta_2}(r \bowtie_{\Theta_1} s) \Rightarrow r \bowtie_{\Theta_1 \wedge \Theta_2} s$$

Next, we rewrite the join condition ($\Theta_1 \wedge \Theta_2$) in conjunctive normal form (CNF), and partition the resulting terms into three groups: conditions that involve only r (Θ_r), conditions that involve only s (Θ_s), and conditions that involve both r and s (Θ_{rs}):

$$r \bowtie_{\Theta_1 \wedge \Theta_2} s \Rightarrow r \bowtie_{\Theta_r \wedge \Theta_s \wedge \Theta_{rs}} s$$

Now we perform selections locally in the data sources of r and s, and only the tuples from r that satisfy Θ_r and the tuples of s that satisfy Θ_s are brought into the mediator to compute the join:

$$\sigma_{\Theta_r}(r) \bowtie_{\Theta_{rs}} \sigma_{\Theta_s}(s)$$

This approach can be combined with semi-joins to further reduce the network overhead. Lu and Carey [17] demonstrated that the additional computational overhead of semi-joins can be higher than the savings in communication costs when the nodes involved are on a high speed local area network and data volumes are small. However, these conditions are rare in big data applications.

Similarly, there is a potentially high cost involved with transforming join conditions to CNF. In the worst case, the number of terms can increase exponentially (although we have not experienced this problem in practice). Note that the overhead related to CNF conversion depends on the number of terms in the join conditions, not on the volume of data to be joined. In our experience, where the data volume is large, this overhead is insignificant compared to the savings in communication overhead. However, we intend to empirically investigate this issue in more detail in the future.

5. Related Work

There are numerous other researchers [1-4, 7-12] who have investigated mediation as a way of resolving structural and semantic conflicts between data sources. However, as far as we can determine, there are no previous reports of adapting a mediation service to a Grid-DBMS environment.

6. Future Work

In the immediate future we will continue to focus on improving data replication features and efficiency. In a Grid-DBMS data is often replicated to improve performance and reliability. Currently, the mediator simply chooses a data source at random in the case of replicated nodes. In the future, we would like to provide a load balancing feature.

We will also attempt to determine the conditions where the savings in network overhead justify CNF transformations and/or semi-joins.

Another area where the mediator requires additional work is in security. In its current form, the mediator does not support encrypted connections to the grid nodes and relies on the individual nodes to perform authentication and authorization of requests. Future work will address both of these shortcomings.

7. References

[1] E. Sciore, M. Siegel, and A. Rosenthal, "Using Semantic Values to Facilitate Interoperability Among Heterogeneous Information Systems", *ACM Transactions on Database Systems*, vol. 19(2), June 1994, pp. 254-290.

[2] G. Wiederhold, "Mediators in the Architecture of Future Information Systems", Readings in Agents, Eds. M. N. Huhns and M. P. Singh, San Francisco, CA, USA: Morgan Kaufmann, 1997, pp. 185-196.

[3] P. B. Lowry, "XML data mediation and collaboration: A proposed comprehensive architecture and query requirements for using XML to mediate heterogeneous data sources and targets," *34th Annual Hawaii International Conference On System Sciences (HICSS)*, Maui, Hawaii, January 3-6, 2001, pp. 2535-2543.

[4] C. H. Goh, S. Bressan, S. Madnick, and M. Siegel, "Context interchange: new features and formalisms for the intelligent integration of information", *ACM Transactions on Information Systems*, vol. 17(3), July 1999, pp. 270.

[5] P. Bergstein and V. Shah, "Conditional Mapping in Data Mediation", *Proceedings of the International Conference on Information and Knowledge Engineering (IKE 2004)*. June 21-24, 2004, Las Vegas, Nevada, USA. CSREA Press 2004, ISBN 1-932415-27-0.

[6] P. Bergstein and A. Sikder, "A JDBC Data Mediation Service", *Proceedings of the International Conference on Information and Knowledge Engineering (IKE 2005),* pages 45-50, June 20-23, 2005, Las Vegas, Nevada. CSREA Press, ISBN 1-932415-81-5.

[7] L. S. Seligman and A. Rosenthal, "XML's Impact on Databases and Data Sharing", IEEE Computer, vol. 34(6), 2001, pp. 59-67.

[8] G. Neugebauer, "GLUE – Using Heterogeneous Sources of Information in a Logic Programming System", *Proceedings of the KI'97 Workshop on Intelligent Information Integration*, Freiburg, 1997.

[9] L. Serafini and F. Giunchiglia and F. Mylopoulos and P. Bernstein, "The Local Relational Model: A Logical Formalization of Database Coordination", *Proceedings of CONTEX'03*, 2003.

[10] H. Wache and H. Stuckenschmidt, "Practical Context Transformation for Information System Interoperability", *Lecture Notes in Computer Science*, vol. 2116, 2001, p. 367.

[11] B. Ludäscher, A. Gupta, and M. Martone, "Model-Based Mediation with Domain Maps", *17th International Conference on Data Engineering (ICDE '01)*, Washington-Brussels-Tokyo, April 2001.

[12] C. Baru, A. Gupta, B. Ludäscher, R. Marciano, Y. Papakonstantinou, P. Velikhov, and V. Chu, "XML-based Information Mediation with MIX", Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data: SIGMOD '99, Philadelphia, PA, June 1-3, 1999, SIGMOD Record, vol. 28(2), 1999, pp. 597-599.

[13] P. Bergstein, "An ODBC CORBA-Based Data Mediation Service", *Proceedings of the International Conference on Information and Knowledge Engineering (IKE 2006),* pages 196-202, June 26-29, 2006, Las Vegas, Nevada. CSREA Press, ISBN 1-60132-003-5.

[14] P. Bergstein, "Query Translation and Where Clause Processing in Data Mediation", *Proceedings of the International Conference on Information and Knowledge Engineering (IKE 2007),* pages 61-66, June 25-28, 2007, Las Vegas, Nevada. CSREA Press, ISBN 1-60132-050-7.

[15] G. Aloisio, M. Cafaro, S. Fiore, and M. Mirto, "The Grid-DBMS: Towards Dynamic Data Management in Grid Environments", *Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC'05)*, volume 2, pages 199-204, 2005, IEEE Computer Society, ISBN 0-7695-2315-3.

[16] D. Kossmann, "The State of the Art in Distributed Query Processing", *ACM Computing Surveys*, Vol. 32, No. 4, December 2000, pages 422-469.

[17] H. Lu and M. Carey, "Some Experimental Results on Distributed Join Algorithms in a Local Network", *Proceedings of the 11th International Conference on Very Large Data Bases (VLDB'85)*, August 1985, Stockholm, Sweden, pages 292-304.

Integration of Big Data Components for NoSQL Problems

G. Speegle[1], **E. Baker**[1]
[1]Department of Computer Science, Baylor University, Waco, Texas, USA

Abstract— *The NoSQL movement is predicated on the belief that relational databases (RDBMS) are inadequate for applications with strong demands on the four V's – volume, velocity, variety and veracity. For NoSQL data stores to replace RDBMS as the dominant model for the next 30 years, not only must they provide capabilities beyond RDBMS, but it must also co-opt the advatnages of the RDBMS. One distinct advantage is a non-procedural query language coupled with query optimization over low-level operations resulting in high performance with easy programming. In this paper, we present an initial analysis of this problem by creating an analytic engine consisting of a relational database, a HADOOP file processing component and a GraphLab distributed memory component. We apply this system to a real-world bioinformatics problem encountered by the GeneWeaver system.*

Keywords: NoSQL, Query Optimization, Big Data, Biological Data

1. Introduction

The relational data model has been a part of database design since the 1970s, and arguably the dominant database model since the 1980s. The longevity and broad adoption of this model is due, in large part, to its robust capabilities, including transaction semantics, recovery, high-level query language and automatic query optimization.

Alternately, NoSQL data models provide improved performance with highly scalable and flexible execution by exploiting dynamically scalable parallel computation. These improvements are gained with certain trade-offs compared to relational databases and are accompanied by varying degrees of services to manage overhead expectations. For example, the Hadooop Ecosystem consists of tools which provide distributed file systems (HDFS), columnar database support (HBase), an SQL-like interface (Hive), and a graph processing tool (Giraph), among others [20]. The Berkeley Data Analytics Stack [24] integrates distributed resource management (Mesos), streaming computation support (Spark), an SQL-like interface (Shark), and more. Google has released tools like Big Table for distributed computation and Pregel for graph computations [5].

This research has been supported by the National Institutes of Health under grant R01AA018776.

The NoSQL data models also assume that the memory and computation power for queries is in the range of 100s-1000s of machines. While this may be true for high-end applications such as Facebook or Twitter, NoSQL queries are of interest to enterprises without that level of resources. In both cases, cost limitations are encountered as the number of NoSQL queries exceeds resource availability. There currently exists few efforts to quantitate the impact of scalable NoSQL computations in a manner that attempts to predict query costs against relative benefit, in terms of resource allocation and query speed. In particular, the plethora of tools available to answer a NoSQL query provide different benefits based on the memory and computation power available. This is analogous to the performance of different algorithms used by relational databases to execute complex SQL queries (e.g., joins). Although the different algorithms can produce the desired answer, the memory available influences the choice of the best algorithm.

The data analytics intensive discipline of bioinformatics is well-situated to benefit from the optimized application of high-performance NoSQL query execution. Biological network data (protein interaction networks, microarray correlation graphs, metabolic modeling networks) provides a common metaphor for system-wide analysis of homogeneous or heterogeneous data sets for representing distinct components among shared or divergent biological processes. A common query of microarray correlation graphs, for example, asks for the common subnetwork conserved between disease states or species, and which subnetworks are consistently altered between these observations. In addition, the general scale-free nature of biological interactions produces data of varying degrees of density [3], depending on inferred threshold inclusion or the coverage of the empirical repository. Data derived from microarray correlation networks is an example of a readily available empirical data source of varying density (as defined by average node degree) and sufficient size to stress NoSQL approaches to subgraph combinatorics.

By leveraging network biological data sets we can examine the common subgraph problem in a resource restricted environment with two distinctly different computation engines available. Here, we use the Hadoop MapReduce tool [20] with the GraphLab graphical computation engine [16]. We also use a fixed resource of a single site implementation with HDFS as the communication mechanism between the two tools. Based on the size and number of graphs, the optimal algorithm for solving the problem

shifts, indicating a query optimization is needed to enable the best performance for NoSQL systems.

2. Related Work

In this section, we examine the traditional relational database model, specifically as used within the GeneWeaver project, the Hadoop ecosystem, the GraphLab computational model, efforts at integrating diverse NoSQL tools and the bioinformatics domain. These components define the problem space for analytical query optimization.

2.1 Relational Model

RDBMS query processing is a well defined process [13]. SQL is input and translated into an operational language such as extended relational algebra. Multiple access paths are created for the implementation of the extended relational algebra program, considering alternatives varying form the use of indices to the order of the operations. Each of these multiple alternatives is evaluated using statistics ranging from the size of the tables to the distribution of values for an attribute. Once the best alternative is selected, the operations are scheduled. Typically, optimization continues until a sufficiently good implementation is generated or a time limit is reached. Operations are arranged in a tree-like structure, where the parent requests data from its children (unary operations have only a single child). The leaves of the tree are accesses to tables stored in the database. Each operation supports a "getNext" method which produces the next set of data items when called. This allows programs to execute in a pipeline providing data when needed. Of course, for some operations, most of the work is performed at initialization (e.g., a sort). Many of the primary problems faced in RDBMS query optimization have analogies with our AQL processing. Our operational language is currently a shell script for executing large data analytic programs on HDFS. The number of alternatives we consider is much smaller than in an RDBMS, but expansion of options is important for the continued improvement of the analytic engine. Finally, the equivalent of pipelining is future work.

2.2 Hadoop

Hadoop [20] is an open source implementation by Apache of the Google BigTable system, developed to support the Google search engine [5]. The principal aspect of Hadoop used in this project is Hadoop MapReduce and Hadoop Distributed File System (HDFS). HDFS allows robust parallel access to large data by automated replication and access between nodes. Data is distributed in 64-256 MB chunks and data is accessed by the node best suited for the task. For example, a distributed file can be read in parallel by every node. MapReduce is a programming paradigm well suited for batch processing of large data (e.g., building the Google web search index). All data is accessed in key-value pairs, both for input and output. In the map phase, data is read (in parallel) from HDFS and assigned a key based on the physical location of the data. The value is the data stored in the file. The output of the map phase is sorted by key so that each instance of the reducer processes only the pairs with the same key. This process is called the shuffle. The output of the reducers are stored in HDFS. MapReduce performance suffers from two primary problems. First, reading and writing large amounts of data to the disk can be slow, even when done in parallel. However, this process provides much of the robust nature of Hadoop MapReduce, since any node can resume a task assigned to any other node by accessing the HDFS. Second, the amount of work done by a reducer may be very unbalanced. One key may be very common while another is very rare. The reducer processing the common key will take significantly longer to finish. This is known as time skew. For our motivational problem, a special purpose MapReduce algorithm for finding connected components called Hash-to-Min could be used [18]. Hash-to-Min requires $2 \log d$ rounds and $3(|V|+|E|)$ communication per round. The goal of our optimization is to require fewer rounds.

2.3 GraphLab

GraphLab is a parallel computation engine based on representing data in a graph [16]. The graph model is automatically distributed across computers by sharing some nodes between sites. Thus, a computation on a shared node requires communication between the sharing sites, but otherwise computation is carried out locally. Data is stored on nodes and edges, and computation is initiated at each vertex. The computation can access the data stored at the vertex, all of the neighbors of the vertex and all of the edges on the vertex. Unlike Hadoop MapReduce, GraphLab iterates very quickly and can perform many calculations very rapidly. However, GraphLab has some limitations, such as the inability to modify the graph structure once it has been read. For this project, we limit ourselves to the tools provided by GraphLab. Thus, no special purpose tools are written to solve a particular problem.

2.4 Integration

There are many systems proposed for processing graph databases, ranging from industrial strength products such as Neo4j [12] to integrated analytical systems like BDAS [24] and Hadoop [20] to research prototypes like GRACE05 [21] and GRACE [22]. In this section, we provide a brief overview of the integration issues of the different systems.

In BDAS and Hadoop, an "ecosystem" of tools can be chosen by the user in order to solve analytic problems. The toolsets are large and growing. However, to our knowledge there is not an automatic mechanism for combining tools. Thus, an end user requires a programmer to write applications which merge a graph intersection tool with a connected components tool. Our framework will allow tools

meeting a specification to be integrated without additional programming.

Neo4j attempts to provide a scalable solution for generic graph representations of a variety of data [12]. It relies on the Cypher query language to rapidly retrieve edges and nodes, perform basic graph operations (unions, graph walks, and statistics). Among an emerging set of graph-based data structures, Neo4j has gained prominence because of its adherence to RDBMS constructs, such as ACID compliance, query language optimization, and a native REST interface. Re-interpreting relational data models as graph models and competing with horizontal scalability continue to be active areas of engagement in graph database research.

Research prototypes are usually focused on fast graph processing, and as such, are competitors to GraphLab as opposed to our Analytic Engine. GRACE05 uses the DOT language from GraphViz [9] for specifying queries. Obviously, DOT can be extended to handle analytic operations, but at this time we are choosing our own AQL in order to focus on query optimization issues.

2.5 Bioinformatics Resources

The field of bioinformatics owes its success, in large part, to the intentional curation and analysis of genome-scale data sets. However, until now, these data sets are often limited in scope by data types, access, or analysis complexity. Several informatics approaches have been proposed to move bioinformatics beyond genomics and into large scale integrative systems biology, these include the BioGrid [6], caBIG [7], and NIF [11], among others. These data-driven resources have attempted to aggregate data and data resources around scalable components, shared data structures, and a common semantics. Other approaches attempt to leverage the natural fit of graph structures to biological data and provide real time analysis of heterogeneous data types, including GeneWeaver [1], GeneNetwork [23], and [8], among numerous others. While well-suited to their domains of interest, each approach is limited by the tight coupling of data structures and analysis tools.

3. Integrated Model

In order for BigData to efficiently utilize NoSQL frameworks as viable alternatives for well-structured and traditional relational models, a common component interaction approach is required. This is analogous to the existence of RDMS query optimization tools. Individual components within a large NoSQL framework can change due to advances in technology or based on the query or type representations. To date, there exists no AQL (Analytic Query Language) that effectively manages queries across a set of NoSQL components, attempting to optimize query processing based on individual components, query type, and data characteristics.

3.1 AQL

We define a simple AQL as the input for our analytic engine. As the field develops, we fully expect an industry accepted AQL-like langauge will evolve, similar to the development of SQL over the last 35 years. Our initial AQL is designed to accommodate the bioinformatics queries used in our motivating example. Also note that a compiler for the AQL is under development, but does not currently exist. At this time, the definitions are to help develop the underlying analytic engine rather than process queries.

A query consists of a graph which can be formatted or filtered. Currently, the result of a query is a graph, but extensions to other output types is part of our future work. Graphs can be directed or undirected, and edges may or may not have weights. Our present work limits graphs to edge lists.

Graphs are formatted in the AS clause. A graph can be converted from undirected to directed by adding reciprocal edges. We do not support converting from directed to undirected. Similarly, weights can be removed with the keyword, UNWEIGHTED, and weights can be added with simple (constant) functions. One interesting project is to add graph properties such as the number of neighbors to the WEIGHT function.

Filtering operations are in the WITH clause, and are currently limited to GraphLab functions or simple predicates based on weights. While the weight predicates are straightforward to apply, the results of the GraphLab functions are not always graphs. The motivating example provides evidence to some of the interesting results in Figure 2.

$\langle query \rangle ::= [format]\langle simple-query \rangle [filter]$
$\langle format \rangle ::= \text{AS}[\text{unweighted}]|[\text{weight} = \langle NUM \rangle]$
$\langle filter \rangle ::= \text{WITH}\langle operation \rangle$

A simple query is a collection of operations applied to graphs. Baylor University graduate student Rovshen Nazarov is writing a compiler for this portion of AQL, extending the work in [10]. This compiler is based on the parsers in [15], [17] and supports intersection, union and difference of graphs. The keywords are INTER, UNION, and DIFF, with strict precedence of UNION the highest and INTER the lowest. Since the result of a query is a graph, whenever a graph is required, a query can be used instead.

$\langle simple-query \rangle ::= FROM\langle graph-list \rangle$
$\langle graph-list \rangle ::= \langle graph \rangle|\langle graph-list \rangle\langle OP \rangle\langle graph-list \rangle$
$\langle graph \rangle ::= \text{graphId}|\langle query \rangle$
$\langle OP \rangle ::= \text{UNION}|\text{DIFF}|\text{INTER}$

As an example, the query to find all of the common subnetworks in a set of four graphs identified simply as g1 through g4 can be specified as

```
FROM g1 INTER g2 INTER g3 INTER g4
WITH connected-components
```

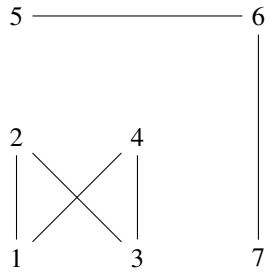


Fig. 1: An example of interpreting the original graph results with two components

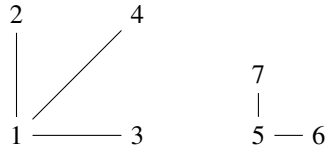


Fig. 2: Connected components interpreted as a graph

Similar to SQL where the results of a query can be used as a subquery, if the results of an AQL query is a graph, the query can be used in the FROM clause. Some of the GraphLab tools do not generate graphs, such as the triangle counting tool which returns a scalar value. Currently, the results of such tools cannot be used in the FROM clause.

Note even when the results of the query can be viewed as a graph, the results may not be expected. For example, the connected components tool generates a set of ordered pairs of node ids. These ordered pairs can be viewed as a graph, but that is not the intention of the tool. Currently, we allow this result to be used in the FROM clause, since the representation corresponds to a graph. However, see Figures 1 and 2 for potential problems.

As an interesting side note, the ability to use functions as part of the AS clause in assigning a weight to an edge could be very useful. For example, suppose we only wanted to consider subgraphs above the size of 1 edge. Adding the larger degree of the nodes incident on the edge as the weight would provide such filtering.

```
FROM (
     AS WEIGHT=max_degree
     FROM g1 INTER g2 INTER g3 INTER g4
     WITH connected-components
)
WITH WEIGHT > 1
```

3.2 Relational Component

The GeneWeaver model (`www.geneweaver.org`) is intended to integrate experimental data and perform analysis across a diversity of species and data types [1], [14]. Successful analytics is achieved by ascertaining bipartite relationships between sets of genes and defined biological processes, disease states, or behavior constructs [2]. Discrimination power is captured by annotating bipartite relationships to curated community resources, including literature, community ontologies, or other user-defined criteria, and applying a wide variety of combinatorial approaches to interrogate aggregated sets for common genetic components [4], [19]. Well-optimized relational data models can store data and metadata for rapid retrieval, while combinatoric analysis, such as maximal biclique or common subgraph discovery, is computed outside of the relational database environment.

In this case, the relational component of the analytic processing system provides robust capabilities for data without the high demands of NoSQL systems. In the GeneWeaver system, for example, the relational database is responsible for maintaining consistent gene ids across different biological data sets. Thus, data is updated only occasionally and is limited in size to a single graph.

3.3 MapReduce Component

Hadoop MapReduce is capable of manipulating very large datasets in a batch processing mode. Therefore, the analytic engine uses MapReduce to calculate the various graph intersections and unions. Since we are using unaltered tools from GraphLab, all weight processing is also done in MapReduce. Given that the key performance bottleneck for MapReduce is the number of iterations, we combine weight processing with intersection and union processing. Algorithm 1 contains pseudocode for the Map function and Algorithms 2 and 3 contain pseudocode for the reduce functions.

Algorithm 1 Pseudocode for the analytic engine Map function for intersection and union. Each edge serves as the key for the reduce phase. The weight is used as a filter.

```
parse node1, node2, weight
if weight = null ∨ weight > threshold then
    if undirected graph then
        source ← min(node1, node2)
        destination ← max(node1, node2)
    else
        source ← node1
        destination ← node2
    end if
    Output (source,destination),0
end if
```

The same mapper function is used for both intersection and union. It simply outputs all edges that have a weight greater than the threshold (currently, only greater than is supported). If the graph is undirected, the mapper ensures common edges are mapped to the same reducer by always listing the least vertex first. Note that if only filtering is required an empty reducer is used.

The reducer is responsible for determining if an edge appears in all of the graphs (in intersection) and for eliminating

Algorithm 2 Pseudocode for the analytic engine Reduce function for intersection.

```
Count number of instances
if Count = NumberGraphs then
    Output key
    if undirected graph then
        Output (destination,source)
    end if
end if
```

Algorithm 3 Pseudocode for the analytic engine Reduce function for union.

```
Output key
if undirected graph then
    Output (destination,source)
end if
```

duplicates (in union). The reducer also converts undirected graphs into directed graphs by duplicating undirected edges. This is needed for processing the graph with GraphLab. However, the intermediate file size would be smaller if the graphs had further MapReduce processing. This remains a topic for future work.

One issue to be noted is that the number of reducer instances is very high – on the order of the number of edges. We can reduce the number of instances by using the first node (labeled source in Algorithm 1) as the key and the second node as the value. This will map all edges incident on the node to the same reducer. For intersection, the reducer uses a map to determine the number of times each neighbor appears in the set of graphs. The edge is emitted if the neighbor appears in every graph. We are currently working to determine if this approach improves performance.

3.4 GraphLab Component

GraphLab is an alternative to Giraph, the Apache Hadoop graph computation engine, and is selected for this project due to the fact it can integrate with Hadoop 2.2 (as of this writing, Giraph could not) and the fact it was not part of the same development scheme. This allows us to show how components from unrelated projects can be combined into a single analytic execution plan.

In order to focus on the query processing aspects of the analytic engine, we restrict ourselves to the standard GraphLab toolkit provided with the installation. For our motivating example, we use the connected component tool. However, this program requires the graph to be in one of the supported formats – a directed graph represented as an edge list or an adjacency list. Although it is possible to modify the GraphLab tool to function with the different format, our goal is to integrate existing tools where possible.

Thus, Algorithm 2, requires no additional iterations of the MapReduce program to modify undirected graphs into directed ones and to remove the weights. Likewise, the MapReduce functions can be used to format graphs for all GraphLab tools, eliminating the need to modify every tool (and all future tools). Since very large data sets may need to be processed by MapReduce anyway, our approach is a strong alternative.

3.5 Access Plan

What remains is to integrate the components into a program which can answer the query and find the subgraphs of a set of graphs. Assume we are given the simple query:

```
FROM (
     AS weights=max_degree
     FROM g1 INTER g2 INTER g3 INTER g4
     WITH connected-components
)
WITH weight > 1
```

We use HDFS as the shared storage between the relational component GeneWeaver, the Hadoop MapReduce component and the GraphLab component. Currently, a shell script handles the actual integration. See Section 4 for potential improvements.

We assume every query is assigned a unique ID, and therefore we can use the query id to uniquely find the data. Assume such a directory is defined as $QUERYID.

Step One

Extract the set of microarray co-expression graphs into the HDFS directory $QUERYID/input and set the variable num_graphs to the number of graphs chosen

Step Two

Execute the MapReduce GraphIntersection program. This will output all of the edges common to all graphs, and adjust the data to an unweighted directed graph.

```
hadoop jar aql.jar GraphIntersect
  $QUERYID/input $QUERYID/inter0 num_graphs
```

Step Three

Execute the GraphLab program on the results to generate the subnetworks. Note that the number of cpus used should be parameterized based on the workload and the system resources. For this example we arbitrarily choose 4.

```
mpiexec -n 4 ./connected_component
        $QUERYID/inter0 $QUERYID/inter1
```

Step Four

Add a weight to each edge consisting of the highest degree of the nodes incident on the edge. This step can be accomplished easily by GraphLab, since reading the graph provides information about the nodes. However, a very large data set relative to the amount of memory available could suggest a MapReduce solution would be better. Fortunately, in this case the graph is the result of the connected_component GraphLab tool. Thus, it is known

that the data fits within available memory. Likewise, the output format of the connected_componet is easily read by GraphLab. A new GraphLab program was written to return this value.

```
mpiexec -n 4 ./max_degree $QUERYID/inter1
$QUERYID/inter2
```

Step Five

Filter the edges via MapReduce using Algorithm 1 and the null reducer.

```
hadoop jar aql.jar filter \$QUERYID/input
\$QUERYID/inter0 num\_graphs threshold
```

4. Future Work

Analytic optimization faces signficant challenges before it can be equivalent to the capability of query optimization for relational databases. However, RDBMS's faced many of the same challenges when they were first developed in the 1970's. Relational operations (especially joins) were considered too slow to be useful. However, the acceptance of SQL as the standard query language and improved algorithms for performing the operations required by SQL has provided sufficient performance for almost 30 years. Only with the recent advent of high velocity, volume, variety and veracity data has the need arisen for the NoSQL alternatives.

Clearly, the development of an analytical query language (AQL) will enable a wider variety of users to fashion ad-hoc queries on big data stores. Since the nature of analytical data is knowledge discovery, ad-hoc queries will be far more common in NoSQL environments than in traditional RDBMS. The simple AQL presented here is not sufficient for providing the robust operations possible for analytic queries. Even the motivating example demonstrates deficiencies with this AQL when the connected components have to be treated as an unusual graph in order to continue processing.

The variety of NoSQL systems provide an intense challenge to AQL. End users are unlikely to form AQL queries. Therefore, we will need programs to generate complex AQL queries from GUI input. SQL provides this functionality by allowing subqueries in several parts of the query. In order for AQL to provide the same functionality, the results of a query need to be used in other parts of the query. However, the high variety of the data in NoSQL applications work against the subquery structure. We are consdering different possibilities:

- XML-style schema definitions allow variety of data within a query, but significantly impacts performance. A query has to generate not only the data, but also the schema. Columnar data stores could be used as the shared storage.
- Hierarchical query languages similar to the query translation in heterogeneous databases can allow a query to be mapped to different data. For example, a join between a graph consisting of users and a RDBMS table of users can be performed by traversing the graph and scanning the table for each node in the graph. If no translation is available, the query could not be answered.
- Generalized data models consider data at its most primitive level and provide API access. These models are similar to how high-level programming languages access data in files. Although this solution would be able to answer the most queries, the end result could be no better than writing a program in Java or C++.
- Combination of the other solutions. Perhaps the best results can be achieved by using all of these ideas in concert. Data can be accessed in simple ways or with a schema. Queries can be written at one level of abstraction and executed at another.

Integrating diverse analytical tools is non-trivial. New and improved versions of tools are being developed constantly. For example, Hadoop MapReduce has only been available for eight years. As such, the successful analytic engine will provide an API that allows tools to integrate. In this paper, we used the simple mechanism of HDFS as the integration tool. However, that is analagous to using the hard drive as a communication mechanism between processes to implement relational operators. Pipelining with iterators is a far more efficient communication mechansim. At the moment, equivalent mechanisms for analytic processing are unclear. Furthermore, RDBMS must be integrated into NoSQL systems as well. This will require generating SQL that integrates with HDFS (as in the motivating example) or at some other point in the analytical data processing.

Additionally, the integration API must provide for a cost function for the analytic tools. The cost function must consider not only the memory and CPUs available, but also the potential for cost differences based on the computation model (i.e., Amazon EC2 costs per CPU unit). An interesing consideration is the tool that performs poor cost analysis may be used when it is inappropriate. Thus, a feedback system is required to adjust the access plan cost estimation.

Analytical query execution plans are more complicated than SQL query execution plans. For our motivating example, the number of different options is intentionally kept very low. For each step, there are only two choices and usually one is the obvious selection. However, the variety explodes quickly when other tools are included as well. Furthermore, with the high level tools used here, it is always possible to create new tools (as was done with MapReduce graph intersection and union). Tools should be kept as simple as possible, but with the greatest reuse possible. Thus, one mapper function that satisifies value filtering, graph intersection and graph union is a good tool, but would it be better to combine this functionality into other tools – such as modifying GraphLab's connected components to filter input based on threshold? Most likely, in some scenarios

one approah is better and in other scenarios another approach performs best.

5. Conclusion

This paper introduces the problem of query processing and optimization in analytic queries, specifically in the bioinformatics domain. It examines the problem space and the issues that need to be resolved. A motivating example is presented and solved using the primitives from Hadoop MapReduce and GraphLab.

GeneWeaver is an ongoing bioinformatics research project that attempts to perform large scale combinatorics on sets of associated biological entities, including elucidation of edge relationships between co-occurring observations. It provides an intuitive interface for investigating common and unique genes among thousands of potential candidate gene sets, but relies on a strictly-typed relational schema to return real time data queries. Even with a well-optimized relational model and supporting analysis tools, queries of very dense bi-partite data structures have bounded upper thresholds without the high performance computing. The GeneWeaver system is an example of an applied application hardened by the limitations of very large and potentialy dense data sets and a requisite *a priori* knowledge of the underlying data and associated analysis tools to achieve optimal results.

NoSQL systems provide analytic capabilities for data with high volume, high velocity, high variety and high veracity demands. These systems are likely to be used in more diverse domains. However, for the usage to be as ubiquitous as RDBMS systems today, some of the strengths of the RDBMS need to be incorporated into NoSQL. We consider the query processing and optimization issues of NoSQL, particularly with respect to the bioinformatics problem of finding the subgraphs common to a set of graphs.

In order to emphaisze the query processing issues, we limit ourselves to a simplified analytic query language, AQL, which uses Hadoop MapReduce to perform graph intersection, union and threshold filtering, and GraphLab tools provided in the default download. As such, the common subnetwork problem can be solved by finding the edges common to all graphs and then finding the connected components. In order to eliminate subnetworks consisting of a single edge, another very simple GraphLab tool is used.

References

[1] Erich J Baker, Jeremy J Jay, Jason A Bubier, Michael A Langston, and Elissa J Chesler. GeneWeaver: a web-based system for integrative functional genomics. *Nucleic Acids Res.*, 40:D1067–1076, 2012. PMID: 22080549.

[2] Erich J Baker, Jeremy J Jay, Vivek M Philip, Yun Zhang, Zuopan Li, Roumyana Kirova, Michael A Langston, and Elissa J Chesler. Ontological discovery environment: a system for integrating gene-phenotype associations. *Genomics*, 94(6):377–387, 2009. PMID: 19733230.

[3] Albert-LÃąszlÃş BarabÃąsi. Scale-free networks: a decade and beyond. *Science*, 325(5939):412–413, 2009. PMID: 19628854.

[4] Tanmoy Bhattacharyya, Sona Gregorova, Ondrej Mihola, Martin Anger, Jaroslava Sebestova, Paul Denny, Petr Simecek, and Jiri Forejt. Mechanistic basis of infertility of mouse intersubspecific hybrids. *Proc. Natl. Acad. Sci. U.S.A.*, 110(6):E468–477, 2013. PMID: 23329330 PMCID: PMC3568299.

[5] Fay et al Chang. Bigtable: A distributed storage system for structured data. *ACM Transactions on Computer Systems (TOCS)*, 26(2):4, 2008.

[6] Andrew et al Chatr-Aryamontri. The BioGRID interaction database: 2013 update. *Nucleic Acids Res.*, 41(Database issue):D816–823, January 2013. PMID: 23203989 PMCID: PMC3531226.

[7] Peter A Covitz, Frank Hartel, Carl Schaefer, Sherri De Coronado, Gilberto Fragoso, Himanso Sahni, Scott Gustafson, and Kenneth H Buetow. caCORE: a common infrastructure for cancer informatics. *Bioinformatics*, 19(18):2404–2412, December 2003. PMID: 14668224.

[8] Jr Dennis, Glynn, Brad T Sherman, Douglas A Hosack, Jun Yang, Wei Gao, H Clifford Lane, and Richard A Lempicki. DAVID: database for annotation, visualization, and integrated discovery. *Genome Biol.*, 4(5):P3, 2003. PMID: 12734009 PMCID: PMC3720094.

[9] John Ellson, Emden Gansner, Lefteris Koutsofios, Stephen C North, and Gordon Woodhull. GraphvizâĂŤopen source graph drawing tools. In *Graph Drawing*, pages 483–484. Springer, 2002.

[10] D. Guinness, P. Karbasi, and R Nazarov. Recommendations made easy, 2014. Unpublished manuscript.

[11] Amarnath Gupta, William Bug, Luis Marenco, Xufei Qian, Christopher Condit, Arun Rangarajan, Hans Michael MÃijller, Perry L Miller, Brian Sanders, Jeffrey S Grethe, Vadim Astakhov, Gordon Shepherd, Paul W Sternberg, and Maryann E Martone. Federated access to heterogeneous information resources in the neuroscience information framework (NIF). *Neuroinformatics*, 6(3):205–217, September 2008. PMID: 18958629 PMCID: PMC2689790.

[12] Florian Holzschuher and René Peinl. Performance of graph query languages: comparison of cypher, gremlin and native access in neo4j. In *Proceedings of the Joint EDBT/ICDT 2013 Workshops*, pages 195–204. ACM, 2013.

[13] Yannis E Ioannidis. Query optimization. *ACM Computing Surveys (CSUR)*, 28(1):121–123, 1996.

[14] Jeremy J Jay and Elissa J Chesler. Performing integrative functional genomics analysis in GeneWeaver.org. *Methods Mol. Biol.*, 1101:13–29, 2014. PMID: 24233775.

[15] John Levine. *Flex & Bison*. " O'Reilly Media, Inc.", 2009.

[16] Yucheng Low, Danny Bickson, Joseph Gonzalez, Carlos Guestrin, Aapo Kyrola, and Joseph M. Hellerstein. Distributed graphlab: A framework for machine learning and data mining in the cloud. *Proc. VLDB Endow.*, 5(8):716–727, April 2012.

[17] Tom Niemann. A compact guide to lex & yacc. 2003.

[18] Vibhor Rastogi, Ashwin Machanavajjhala, Laukik Chitnis, and Anish Das Sarma. Finding connected components on map-reduce in logarithmic rounds. *CoRR*, abs/1203.5387, 2012.

[19] Andrew et al Roth. Potential translational targets revealed by linking mouse grooming behavioral phenotypes to gene expression using public databases. *Prog. Neuropsychopharmacol. Biol. Psychiatry*, 40:312–325, 2013. PMID: 23123364.

[20] Konstantin Shvachko, Hairong Kuang, Sanjay Radia, and Robert Chansler. The hadoop distributed file system. In *Mass Storage Systems and Technologies (MSST), 2010 IEEE 26th Symposium on*, pages 1–10. IEEE, 2010.

[21] Srinath Srinivasa and M Harjinder Singh. Grace: A graph database system. *COMAD 2005b, Hyderabad, India*, 2005.

[22] Guozhang Wang, Wenlei Xie, Alan Demers, and Johannes Gehrke. Asynchronous large-scale graph processing made easy. In *6th Biennial Conference on Innovative Data Systems Research (CIDRâĂŹ13)*, 2013.

[23] Jintao Wang, Robert W Williams, and Kenneth F Manly. WebQTL: web-based complex trait analysis. *Neuroinformatics*, 1(4):299–308, 2003. PMID: 15043217.

[24] Matei Zaharia, Mosharaf Chowdhury, Michael J Franklin, Scott Shenker, and Ion Stoica. Spark: cluster computing with working sets. In *Proceedings of the 2nd USENIX conference on Hot topics in cloud computing*, pages 10–10, 2010.

Comparison and Review of Memory Allocation and File Access Techniques and Techniques preferred for Distributed Systems

Muazzam A. Khan, Nauman Nisar,
Department of Computer Engineering,
College of Electrical and Mechanical Engineering,
National University of Sciences and Technology,Islamabad, Pakistan.
khattakmuazzam@gmail.com, naumannisarghuri@gmail.com,

Abstract

Memory Allocation is a process in which operating system manages the Primary Memory and allocates user programs in spaces in the Main Memory. File Access is a process in which the files required to execute are accessed within in the Main Memory and brought to the CPU. Memory Allocation and file access methods play an important role in optimizing the CPU performance and primary memory performance. The paper focuses on the techniques that are applied for memory allocation and file retrieval and comparison of the techniques which performs better is Distributed Systems environment. This paper will also enlist some of the good features that should be included within techniques for memory allocation and file access within Distributed Systems.

Keywords: *Distributed Systems, Memory Allocation, File Access Methods, Memory Management, Memory Allocation Techniques.*

1. Introduction

Distributed systems are referred to as distributed computing systems; distributed system is a set of systems that acts like a one large computer. There are a lot of distributed computing systems projects on the Internet that helps in solving complex and difficult problems by sharing their resources by using different people's computers processing and other resources [1].

Memory management is the part of an operating system which manages primary and main memory. Memory management keeps complete record of all memory location in the main memory either it is allocated to a process or it is not allocated or free. It keeps record of memory, which is to be allocated to processes an processors. It decides which process should be delivered to memory at what time. It keeps record of the memory when it is freed or unallocated and maintains its status accordingly.

The main role of the memory management is to satisfy the requests made by processes in order to get executed by the processor [1].Whenever new processes in created it requests for memory or sometimes during execution some more memory is needed by the invoked processes. In both the cases the main memory should be capable to provide such memory space for the processes that made request. Main memory has two types of partitions. One is Low Memory in which the operating system is assigned and other one is High Memory: User processes are kept and allocated in that part of the memory.

Some systems use Pages for memory allocation and some use segments for memory allocation. Paged allocation partition the computer's main/primary memory into fixed-size frames or units called Page frames, and the program's allocated space into the pages of the one fixed

size [15]. Segmentation is the only memory allocation and management technique that does not entertain user's program while giving them a 'linear and contiguous address space. There are two types of memory allocation and that are Single Contiguous Allocation and Partitioned Allocation.
File access is a method of accessing files within the main memory. For the purpose two registers are used commonly, a base register and a limit register. The base register holds the smallest legal physical memory address and the limit register specifies the size of the range. For example, if the base register holds 300000 and the limit register is 1209000, then the program can legally access all addresses from 300000 through 411999 [8].

Section 2 will discuss literature review and related work in the paper. Section 3 will discuss various techniques of memory allocation in OS. Section 4 will compare different techniques of memory allocation and determine which technique is better for which types of systems. Section 5 will discuss about techniques for memory allocation and file access that should be used for distributed systems, recommendations and conclusions derived from this whole research and finally section 6 has references of resources from where paper is researched.

2. Related Work

This section will discuss about the search engines that were used for the research of the paper and the keywords that were used in it to make research and also will include the related work in which different techniques that were represented.

The search engines used for getting the relevant data are listed as follows:

- Google Scholars
- IEEE Explorer
- Springer Explorer
- Cite Seer Explorer

The keywords used to search relevant papers. All the citing is done using these keywords:

- Distributed Systems and Memory Management
- Memory Management.
- Memory Allocation in Distributed Systems
- Distributed Memory Management.

With conventional SMP systems, multiple processors execute instructions in a single address space. . It is possible to run parts of a program in parallel, generally by using threads to specify such parallelism and using synchronization primitives to prevent race conditions. Distributed shared memory is a technique for making multicomputer easier to program by simulating a shared address space on them. Different models are discussed in these notes for Distributed shared memory [15].

The methodology and the results of Malloc function which is one of necessities of memory management in distributed environment is presented in [1].
Modern software requires and adds an increasing reliance in dynamic memory allocation but the direct management is always error-prone. So Garbage Collection (part of Memory Management) eliminates many of these bugs [3][2].
Rainbow OS needs no locks for accessing memory or shared objects during executions. It uses an optimistic synchronization, allowing transactions to proceed locally and its validation at the end of it an example of distributed virtual memory [4].

3. Memory Allocation Techniques

In this section we will discuss techniques for user process memory management techniques

which are of two types one internal to process and others are external to process [12].

Internal to process techniques / schemes are

Segments. Process memory is divided into logical segments such as text, data, heap and stack etc. Some of them are read only, others are read-write. Some are known at compile time and some of them grow dynamically as process progress [7].

Static Allocations. These are done at compile time and done using heaps and stacks.

Dynamic Allocation. As static allocation is not enough for heaps and stacks. Dynamic storage need is necessary because user not need to know about the allocation of memory to a process when it is executed. It requires two fundamental operations: allocate dynamic storage and free it when no longer needed. Stacks are good when it is predictable when to remove process from memory and Heap is good when it is not predictable when to remove the process from the memory. Heap-based dynamic memory allocation techniques typically maintain a free list, which keeps track of all the holes. These are done using algorithms such as Best Fit, Worst Fit and First Fit [9].

Garbage Collection. Freeing memory from unused programs automatically. It can take up to 50% of time for cleaning unused programs from memory which was spent to allocate them to memory [2].

External to process techniques / schemes are
Static Relocation. Puts OS in the highest memory and assign each process a segment in which it fits perfectly and add address or its link with secondary memory logged [6]. Problems with static relocation are Safety of OS as Static relocation done process cannot take more size then when it was initialized and cannot move when it starts execution.

Dynamic Relocation. In dynamic relocation there are two address spaces one is physical address space and other is virtual or logical address space [5]. Virtual Address is the address which seen by user or process but physical address space is space which is visible to operating system only. Different virtual addresses can reside in one location in physical address but they are known to users as different address spaces [4].

Swapping. The process of removing or relocating a process from the main memory to disk and maintaining its complete state on the disk for further use [14]. OS can restore space in memory by swapping the blocked or ready processes from the memory.

Compaction. Dynamic relocation to the memory causes external fragmentation in the memory so for removing compaction is used in memory which moves the process so that memory allocation can be contiguous.

Paging. In paging memory allocation technique data is fetched from disk in fixed size blocks called pages [13]. Paging is an important part of virtual memory management. As data is fetched in fixed size block the fragmentation occurs.

Paging is a vital part of virtual memory implementation in most of the operating systems; it allows them to use disk storage for data that is not in physical random-access memory (RAM).

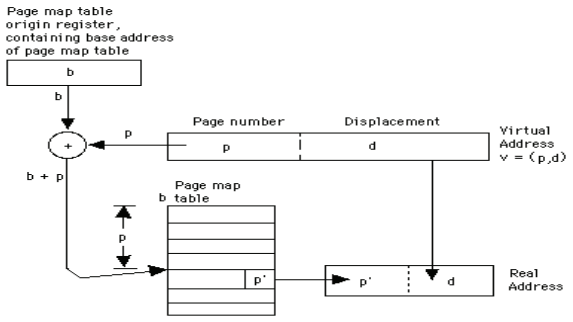


Fig 1: Page Address Translation

4. Comparison

This section of paper has results of comparison of memory allocation techniques that are used in main memory management. The parameters that are used for comparison are advantage; disadvantage and best for are used. Comparison is done in tabular form.

We will discuss about techniques and file systems that have higher impact on Distributed, Centralized and Standalone Systems. It is listed below in the table 1.

Techniques	Computation Required	Frames / Blocks Size	Allocation Time	Fragmentation	Performance	Best Performance For Systems
Segments	Low	Variable	Low	Internal	High	Centralized Systems, Distributed Systems
Static Allocation	Low	Fixed	Low	External	Low	----
Dynamic Allocation	High	Fixed, Variable	High	Internal, External	High	Distributed Systems, Centralized Systems
Static Relocation	Low	Variable	Low	External	Low	----
Dynamic Relocation	High	Variable, Fixed	High	Internal, External	High	Distributed Systems, Centralized Systems
Swapping	High	Fixed, Variable	High, Low	Internal, External	High	Distributed Systems, Centralized Systems
Compaction	High	----	High	Removes it	High	Distributed Systems, Modern Personal Systems
Paging	High	Fixed	High	Internal	High	Distributed Systems, Centralized Systems

Techniques	Computation Required	Frames / Blocks Size	Allocation Time	Fragmentation	Performance	Best Performance For Systems
TFS (Transparent File System)	Low	Fixed, Variable	Low	Internal	High (Low for Ordinary files)	Distributed Systems, Clouds
AFS (Andrew File System)	High	Variable	Low	----	High (Distributed Systems)	Distributed Systems, Standalone Systems
NFS (Sun Network File System)	High	Variable	Low	----	High (Distributed Systems)	Distributed Systems, Standalone Systems
GFS (Google File System)	High	Variable	High	----	High (Distributed Systems & Clusters)	Distributed Systems, Cloud Based & Clusters
HFS (HADOOP FILE SYSTEM)	High	Variable	High	----	High (Distributed Systems & Clusters)	Distributed Systems, Cloud Based & Clusters

Table 1: Memory Management & File Access Systems and Techniques Comparison.

5. Conclusion

It is concluded and recommended that dynamic allocation and relocation for internal process and external process respectively are better for distributed systems because in distributed systems a large number of processes can be present at a time with similar priorities in queue at different sites. So dynamic allocation and reallocation of processes in distributed memories is required. These techniques can be merged with other techniques such as swapping and compaction in removing their fragmentation and replacing them with priority. In dynamic relocation paging is one of the best method to address space in different memories with multiple paging algorithms [13].

Thus each technique has its unique pros and cons, their variations can be useful in distributed systems some of which are discussed in [1][2][3][13] and [14]

6. References

[1] Sam Toueg, **"***Unreliable failure detectors for reliable distributed systems*", Journal of the ASCM, 1996.

[2] Eliot J., Moss B., Richard L. Hudson, Ron Morrison and David S. Munro, "*Training Distributed Garbage: The DMOS Collector*", 1992.

[3] Viral Kapadia, "*Comparative Study and Implementation of Garbage collection for Distributed Environment Using Client Server Approach in Train Algorithm*" M.E. Thesis. 2009.

[4] M. Wende, "*Communcation model of a distributed virtual memory*". Ph.D. thesis, Ulm University, 2003.

[5] Huang Xian-ying , Wang Yue , Chen Yuan. "*Memory management strategy in embedded real-time system*", Computer Engineering and Design, 2004.

[6] S. Liang, R. Noronha, and D. K. Panda,, "*Swapping to Remote Memory over Infinite Band: An Approach using a High Performance Network Block Device*", IEEE Cluster Computing, Sept. 2005

[7] Xie Yinqiao, Li Guangjun, "*Memory management method of an embedded system based on Micro/OSII*". University of Electronic Science and Technology, 2006(In Chinese)

[8] H.Midorikawa, H.Koyama, M.Kurokawa, R.Himeno" *The Design of Distributed Large Memory System DLM and DLM Compiler*", IEICE Technical Report. Computer systems, Vol.107, No.398, pp. 29-34, 2007

[9] S. Tikar, and S. Vadhiyar, "*Efficient reuse of replicated parallel data segments in computational grids"*. Future Generation Computer Systems 24, (644-657) 2008.

[10] Paul Krzyzanowski , "*Distributed Shared Memory and Memory Consistency Models*". Rutgers University – CS 417: Distributed Systems Notes. 2009.

[11] George Coulouris, Jean Dollimore, Tim Kindberg, "*Distributed Systems: Concepts and Design, 4/e*", 2009.

[12] Stallings, "*Operating Systems*", Prentice Hall, 3rd edition, 1998.

[13] M´onica Serrano, Salvador Petit, Julio Sahuquillo, Rafael Ubal, Houcine Hassan, and Jos´e Duato, *Page-Based Memory Allocation Policies of Local and Remote Memory in Cluster Computers*", IEEE 18th International Conference on Parallel and Distributed Systems, 2012.

[14] Shogo Saito, Shuichi Oikawa, "*Exploration of non-volatile memory management in the OS kernel*", Third International Conference on Networking and Computing, 2012.

[15] KAPADIA V.V. & THAKORE D.G., "*Adaptive Distributed Memory Management for Distributed System by Distributed Memory Allocation and De-allocation*", Journal of Information Systems and Communications, 2012.

[16] JAMES CIPAR, MARK D. CORNER and EMERY D. BERGER. "*Contributing Storage using the Transparent File System*", ACM Transactions on Storage, Vol. V, No. N, Month 20YY.

[17] Sunita Suralkar, Ashwini Mujumdar, Gayatri Masiwal and Manasi Kulkarni **"***Review of Distributed File Systems: Case Studies***",** International Journal of Engineering Research and Applications (IJERA) 2013.

SESSION

BIG DATA SEARCH AND MINING METHODS

Chair(s)

TBA

Decomposition of Inverted Lists and Word Labeling: A New Index Structure for Text Search

Yangjun Chen[1], and Weixin Shen[2]
Dept. Applied Computer Science, University of Winnipeg, Winnipeg, Manitoba, Canada
[1]y.chen@uwinnipeg.ca, [2]wxshen1986@gmail.com

Abstract – *In a text database, a set of documents is maintained. To enquiry such a database, two kinds of queries are quite often used. One is the so-called conjunctive query, represented by a set of terms connected by conjunction (∧); and the other is the disjunctive query, which is also a set of terms, but connected by disjunction (∨). In this paper, we discuss an efficient and effective index mechanism to support the evaluation of both these two kinds of queries based on the inverted files. The main idea behind it is to associate each document word with an interval sequence based on a trie structure constructed over documents; and decompose an inverted list into a collection of disjoint sub-lists. In this way, both conjunctive and disjunctive queries can be conducted by performing a series of simple interval containment checkings. Experiments have been conducted, which shows that the new index is promising.*

Keywords: Search engine; Inverted files; queries

1 Introduction

Indexing the Web for fast keyword search is a key technology. In the past several decades, different indexing methods have been developed for this task, such as inverted files [1], signature files [5, 6] and signature trees [2] for indexing texts, and suffix trees and tries [7] for string matching. Especially, different variants of inverted files have been used by the Web search engines to find pages satisfying a query [8].

A text database can be roughly viewed as a collection of documents and each document is stored as a list of words. Over the documents, there are two kinds of Boolean queries, that is, queries that can be constructed from query terms by conjunction (∧) or disjunction (∨). A document D is an answer to a conjunctive query $w_1 \wedge w_2 \wedge \ldots \wedge w_k$ if it contains every w_i for $1 \le i \le k$ while D is an answer to a disjunctive query $w_1 \vee w_2 \vee \ldots \vee w_l$ if it contains any w_i for $1 \le i \le l$. Conjunction and disjunction can be nested to arbitrary depth, but can always be transformed to a conjunctive normal form:

$$(w_{11} \vee \ldots \vee w_{1l_1}) \wedge \ldots \wedge (w_{k1} \vee \ldots \vee w_{kl_k}).$$

In this paper, we discuss a new method to evaluate both conjunctive and disjunctive queries by decomposing an inverted list into a collection of disjoint sub-lists. The decomposition is based on the construction of a trie structure T over documents and then associating each document word with an interval sequence generated by labeling T by using a kind of tree encoding.

With this method, we can improve the efficiency of traditional methods by an order of magnitude or more.

2 New Index Structures

In this section, we mainly discuss our index structure, by which each word with high frequency will be assigned an interval sequence. We will then associate intervals, instead of words, with inverted sub-lists. To clarify this mechanism, we will first discuss interval sequences for words in 2.1. Then, in 2.2, how to associate inverted lists with intervals will be addressed.

2.1 Intervals assigned to words

Let $\boldsymbol{D} = \{D_1, ..., D_n\}$ be a set of documents. Let $W_i = \{w_{i1}, \ldots, w_{ij_1}\}$ $(i = 1,\ldots, n)$ be all of the words appearing in D_i, to be indexed. Denote $\boldsymbol{W} = \bigcup_{i=1}^{n} W_i$, called the *vocabulary*. We define the word appearance frequency by the following formula:

$$f(w) = \frac{num.of\ documents\ containing\ w}{num.\ of\ documents}, \qquad (w \in \boldsymbol{W}).$$

We then define a *frequency threshold* ζ. For any word w with $f(w) < \zeta$, we will associate it with an inverted list in a normal way, denoted as $\delta(w)$, exactly as in the method of inverted files. However, for all those with $f(w) \ge \zeta$, we will create a new index. For this, we will represent each D_i as a sequence containing all those words w with $f(w) \ge \zeta$, decreasingly sorted by $f(w)$. That is, in such a sequence, a word w precedes another w' if w is more frequent than w' in all documents. In addition, for any subset of words that have the same appearance frequency a *global ordering* is defined so that in each sorted word sequence this global ordering is followed. In addition, we maintain a hash table $\mathcal{H}$ that maps each word w to its inverted list $\delta(w)$ or to its new index.

Example 1 In Table 1, we show a set of four documents, their words w with $f(w) \ge \zeta = 0.4$, and the corresponding sorted word sequences, where we use a character to represent a word for simplicity.

Table 1: Documents and word sequences

DocID	words	sorted word sequence
1	*c, a, f, m, p*	*c, f, a, m, p*
2	*c, f, b, a*	*c, f, a, b*
3	*b, a, c, d*	*c, a, d, b*
4	*f, d, p, m*	*f, d, m, p*

Notice that the global order on {*f*, *a*, *c*} (with $f(w)$ = 0.75) is set to be $c \rightarrow f \rightarrow a$ while the global order on {*m*, *b*, *p*, *d*} (with $f(w)$ = 0.5) is $d \rightarrow b \rightarrow m \rightarrow p$.

For each document D_i (i = 1, …, n), we will use s_i to represent its sorted word sequence. Over all such sequences S = {s_1, …, s_n}, we will construct a digit tree, called a *trie*, as follows.

Assume that $\boldsymbol{W}$ = {w_1, …, w_m}. If $|S|$ = 0, the trie is, of course, empty. For $|S|$ = 1, *trie*(S) is a single node. If $|S| > 1$, S is split into m (possibly empty) subsets S_1, S_2, …, S_m so that a string is in S_j if its first word is w_j ($1 \le j \le m$). The tries *trie*(S_1), *trie*(S_2), …, *trie*(S_n) are constructed in the same way except that at the kth step, the splitting of sets is based on the kth words in the sequences. They are then connected from their respective roots to a single node to create *trie*(S). In Fig. 1, we show a trie T constructed over the sorted word sequences in Table 1.

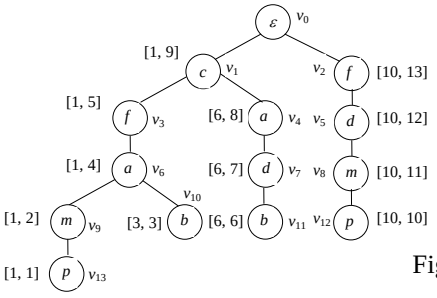


Figure 1. A trie

In the trie, v_0 is a *virtual* root, labeled with an *empty* word ε while any other node is labeled with a *real* word. Therefore, all the words on a path from the root to a leaf spell a sorted word sequence for a certain document. For instance, the path from v_0 to v_{13} corresponds to the sequence: *c*, *f*, *a*, *m*, *p*. Then, to check whether two words w_1 and w_2 are in the same document, we need only to check whether there exist two nodes v_1 and v_2 such that v_1 is labeled with w_1, v_2 with w_2, and v_1 and v_2 are on the same path. This shows that the *reachability* needs to be checked for this task, by which we ask whether a node v can reach another node u through a path. If it is the case, we denote it as $v \Rightarrow u$; otherwise, we denote it as $v \not\Rightarrow u$. The reachability problem on tries can be solved very efficiently by using a kind of tree encoding [3], which labels each node v in a trie with an interval $I_v = [\alpha_v, \beta_v]$, where β_v denotes the rank of v in a *post-order* traversal of the trie. Here the ranks are assumed to begin with 1, and all the children of a node are assumed to be ordered and fixed during the traversal. Furthermore, α_v denotes the lowest rank for any node u in $T[v]$ (the subtree rooted at v, including v). Thus, for any node u in $T[v]$, we have $I_u \subseteq I_v$ since the post-order traversal enters a node before all of its children, and leaves after having visited all of its children. In Fig. 1, we also show such a tree encoding on the trie, assuming that the children are ordered from left to right. It is easy to see that by interval containment we can check whether two nodes are on a same path. For example, $v_3 \Rightarrow v_{10}$, since $I_{v_3} = [1, 5]$, $I_{v_{10}} = [3, 3]$, and $[3, 3] \subset [1, 6]$; but $v_2 \not\Rightarrow v_9$, since $I_{v_2} = [10, 13]$, $I_{v_9} = [1, 2]$, and $[1, 2] \not\subset [10, 13]$.

Let $I = [\alpha, \beta]$ be an interval. We will refer to α and β as $I[1]$ and $I[2]$, respectively.

Lemma 1 For any two intervals I and I' generated for two nodes in a trie, one of four relations holds: $I \subset I'$, $I' \subset I$, $I[2] < I'[1]$, or $I'[2] < I[1]$. □

However, more than one node may be labeled with the same word, such as nodes v_9, and v_8 in Fig. 1. Both are labeled with word *m*. Therefore, a word may be associated with more than one node (or say, more than one node's interval). Thus, to know whether two words are in the same document, multiple checkings may be needed. For example, to check whether *p* and *d* are in the same document, we need to check v_{13} and v_{12} each against both v_7 and v_5, by using the node's intervals.

In order to minimize such checkings, we associate each word w with a word sequence of the form: $L_w = I_w^1, I_w^2, \ldots, I_w^k$, where k is the number of all those nodes labeled with w and each $I_w^i = [I_w^i[1], I_w^i[2]]$ ($1 \le i \le k$) is an interval associated with a certain node labeled with w. In addition, we can sort L_w by the interval's first value such that for $1 \le i < j \le k$ we have $L_w^i[1] < L_w^j[1]$, which will greatly reduce the time for the reachability checking. We illustrate this in Fig. 2, in which each word in Table 1 is associated with an interval sequence. From this figure, we can see that for any two intervals I and I' in L_w we must have $I \not\subset I'$, and $I' \not\subset I$ since in any trie no two nodes on a path are labeled with the same word.

c:	[1, 9]
f:	[1, 5][10, 13]
a:	[1, 4][6, 8]
d:	[6, 7][10, 12]
b:	[3, 3][6, 6]
m:	[1, 2][10, 11]
p:	[1, 1][10, 10]

Figure 2. Sorted interval sequences

As will be seen below, using such interval sequences, the checking of whether two words are in the same document can be done in a very efficient way.

Definition 1 (*word topological order*) Let S = {s_1, s_2, ..., s_n} be a set of n sorted word sequences. A word topological order over S is a sequence $\mathcal{G}$ = w_1, w_2, …, w_m, which contains all the words appearing in S such that for any two words w and w' if w appears before w' in some s_j ($1 \le j \le n$) then w appears before w' in $\mathcal{G}$, denoted as $w \prec w'$. □

In Fig. 2, the words are also listed (from top to bottom) in a word topological order with respect to the sorted word sequences given in Table 1. To find a word topological order over $S = \{s_1, s_2, ..., s_n\}$ with $\boldsymbol{W} = \{w_1, \ldots, w_m\}$, we will transform the corresponding trie T to an *acyclic directed graph* (DAG) G by splitting the node set of T (except for the virtual root) into m groups such that all the nodes in a group are labeled with the same word, and then collapsing each group g to a single node u. There is an edge in G from u (standing for a group g) to u' (for another group g') if T contains (x, y) with $x \in g$ and $y \in g'$. For example, the trie shown in Fig. 1 will be transformed to a DAG shown in Fig. 3(a). Using a hash function H on the words in $\boldsymbol{W}$, the transformation can be done in $O(|\boldsymbol{W}|)$ time, by which all those nodes labeled with the same word w will be mapped to a single node identified by $H(w)$.

Let $G(V, E)$ be such a DAG. It is well known that only $O(|V| + |E|)$ time is required to find a *topological order* of G, which is a linear ordering of all its nodes such that if $u \rightarrow v \in E$, then u appears before v in the ordering. Replacing each node in the ordering with the corresponding word, we will obtain a word topological sequence, as illustrated in Fig. 3(b).

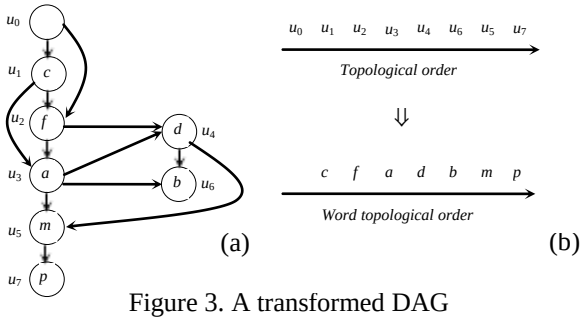


Figure 3. A transformed DAG

Now we consider two words w, w' with $w \prec w'$. It is easy to see that any interval in L_w cannot be contained in any interval in $L_{w'}$. Thus, to check whether w and w' are in the same document, we need only to check whether there exist $I \in L_w$ and $I' \in L_{w'}$ such that $I \supset I'$. This checking can be efficiently conducted as follows.

- Assume that $w \prec w'$. Let $L_w = I_w^1, I_w^2, \ldots, I_w^k$. Let $L_{w'} = I_{w'}^1, I_{w'}^2, \ldots, I_{w'}^{k'}$.
- Step through L_w and $L_{w'}$ from left to right. Let I_w^p and $I_{w'}^q$ be the intervals currently encountered. We will do one of the following operations:
  (1) If $I_w^p \supset I_{w'}^q$, report that w and w' are in the same document. Stop.
  (2) If $I_w^p[2] < I_{w'}^q[1]$, move to I_w^{p+1} if $p < k$ (then, in a next step, we will check I_w^{p+1} against $I_{w'}^q$.)
  (3) If $I_w^p[1] > I_{w'}^q[2]$, move to $I_{w'}^{q+1}$ if $q < k'$ (then, in a next step, we will check I_w^p against $I_{w'}^{q+1}$).
  (4) If $I_w^p \not\subset I_{w'}^q$ and $p = k$ or $q = k'$, report that w and w' are not in the same document. Stop.

The above process is referred to as a *two-word checking*, in which each interval in L_w and $L_{w'}$ is accessed only once. So only $O(|L_w| + |L_{w'}|)$ time is required. In Fig. 4, we illustrate the working process to check whether two words *d* and *m* are in a same document shown in Table 1.

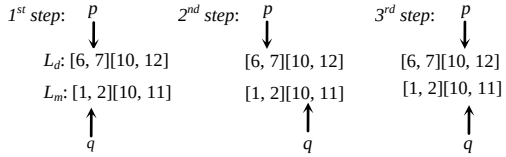


Figure 4. Illustration of two-word checking

In Fig. 4, we first notice that L_d = [6, 7][10, 12] and L_m = [1, 2][10, 11]. In the 1st step, we will check L_d^1 = [6, 7] against L_m^1 = [1, 2]. Since $L_d^1[1] = 6 > L_m^1[2] = 2$, we will check L_d^1 against L_m^2 = [10, 11] in a next step, and find $L_d^1[2] = 7 < L_m^2[1]$. So we will have to do the third step, in which we will check L_d^2 = [10, 12] against L_m^2. Since $L_d^2 \supset L_m^2$, we get to know that *d* and *m* are in the same document.

What we want is to extend this process to check whether a set of words are in the same document, based on which an efficient evaluation of conjunctive queries can be achieved. We will address this issue in Section 3.

2.2 Assignment of DocIds to Intervals

Another important component of our index is to assign document identifiers to intervals. An interval I can be considered as a representative of some words, i.e., all those words appearing on a *prefix* in the trie, which is a path P from the root to a certain node that is labeled with I. Then, the document identifiers assigned to I should be those containing all the words on P. For example, the words appearing on the prefix: $v_1 \rightarrow v_3 \rightarrow v_6$ in the trie shown in Fig. 1 are words *c*, *f*, and *a*, represented by the interval [1, 4] associated with v_6. So, the document identifiers assigned to [1, 4] should be {1, 2}, indicating that both documents D_1 and D_2 contain those three words. See the trie shown in Fig. 5 for illustration, in which each node v is assigned a set of document identifiers that is also considered to be the set assigned to the interval associated with v.

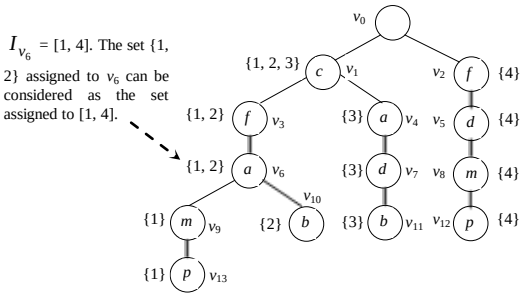


Figure 5. Illustration for assignment of document identifiers

Let v be the ending node of a prefix P, labeled with I. We will use $\delta(I)$, interchangeably $\delta(v)$, to represent the set of document

identifiers containing the words appearing on P. Thus, we have $\delta(v_6) = \delta([1, 4]) = \{1, 2\}$.

Lemma 2 Let u and v be two nodes in a trie T. If u and v are not on the same path in T, then $\delta(u)$ and $\delta(v)$ are disjoint, i.e., $\delta(u) \cap \delta(v) = \Phi$. □

Proposition 1 Assume that $v_1, v_2, \ldots, v_j$ be all the nodes labeled with the same word w in T. Then, $\delta(w)$, the inverted list of w (i.e., the list of all the documents identifiers containing w) is equal to $\delta(v_1) \uplus \delta(v_2) \uplus \ldots \uplus \delta(v_j)$, where $\uplus$ represents *disjoint union* over disjoint sets that have no elements in common.

Proof. Obviously, $\delta(w)$ is equal to $\delta(v_1) \cup \delta(v_2) \cup \ldots \cup \delta(v_j)$. Since $v_1, v_2, \ldots, v_j$ are labeled with the same word, they definitely appear on different paths as no nodes on a path are labeled with the same word. According to Lemma 2, $\delta(v_1) \cup \delta(v_2) \cup \ldots \cup \delta(v_j)$ is equal to $\delta(v_1) \uplus \delta(v_2) \uplus \ldots \uplus \delta(v_j)$. □

As an example, see the nodes v_2 and v_3 in Fig. 5. Both are labeled with word *f*. So the inverted list of *f* is $\delta(v_2) \uplus \delta(v_3) = \{4\} \uplus \{1, 2\} = \{1, 2, 4\}$.

3 Query Evaluation

Based on the interval sequences associated with words and the lists of document identifiers with intervals, we design our algorithm for evaluating queries.

3.1 Containment checking

Let $Q = \{w_1, w_2, \ldots, w_l\}$ be a set of words. Without loss of generality, assume that $w_1 \prec w_2 \prec \ldots \prec w_l$. We will check whether $w_1, w_2, \ldots, w_l$ are in the same document. For this purpose, we need to check whether there exists an interval sequence $\boldsymbol{I} = I_1, I_2, \ldots, I_l$ such that $I_j \in L_{w_j}$ and $I_j \supset I_{j+1}$ $(1 \le j \le l)$, where $I_{l+1} = \phi$, representing an empty interval. We call $\boldsymbol{I}$ a *containment sequence*.

Lemma 3 Let $Q = \{w_1, w_2, \ldots, w_l\}$ with $w_1 \prec w_2 \prec \ldots \prec w_l$. Denote by I_j an interval in L_{w_j} $(1 \le j \le l)$. If for some $1 \le i < j \le l$ we have $I_i \supset I_l$ and $I_j \supset I_l$, then $I_i \supset I_j$. □

As an example, consider $Q = \{f, a, p\}$ with $f \prec a \prec p$. From Fig. 2, we can see that $L_f = [1, 5][10, 13]$, $L_a = [1, 4][6, 8]$, and $L_p = [1, 1][10, 10]$. Obviously, $I_f^1 = [1, 5] \supset I_p^1 = [1, 1]$, and $I_a^1 = [1, 4] \supset I_p^1 = [1, 1]$. Then, we must have $I_f^1 \supset I_a^1$.

According to the above lemma, the checking of $I_{j+1} \subset I_j$ can be replaced by checking whether we have $I_{j+1} \supset I_l$ if we know $I_j \supset I_l$. Thus, the task to find a containment sequence can be done by slightly modifying step (1) in the two-word checking discussed in 2.1. That is, each time we find p, q $(1 \le p \le |L_{w_{l-1}}|, 1 \le q \le |L_{w_l}|)$ such that $I_{w_{l-1}}^p \supset I_{w_l}^q$, we need only to further check whether there exist $l - 2$ other intervals $I_1, I_2, \ldots, I_{l-2}$ such that each I_j is in L_{w_j} and $I_j \subset I_{w_l}^q$ for $1 \le j \le l - 2$. This will greatly simplify the process for finding a containment sequence.

For this purpose, we define an operation $con(w, I, j)$ to check whether an interval I is contained in some interval between jth and the last interval in L_w. If I is contained in an ith interval in L_w, the return value of $con(w, I, j)$ is a pair (*true*, i); otherwise, the return value is (*false*, i'), where i' is the least number such that $I_w^{i'}[1] > I[2]$. In addition, to simplify the control process, we place a *sentinel* at the end of L_w, whose value is set to be $[\infty, \infty]$ so that whenever we reach the sentinel of L_w, $con(w, I, j)$ returns (*false*, $|L_w| + 1$).

This operation will be used in the following algorithm, by which we will check, for a set $Q = \{w_1, w_2, \ldots, w_l\}$ with $w_1 \prec w_2 \prec \ldots \prec w_l$, whether each L_{w_j} $(1 \le j \le l)$ possesses an interval which contains a given interval I.

The input of this algorithm is a triplet (Q, I, b), where b is an integer array of length $|Q|$ with each $b[j]$ indicating the starting position to check L_{w_j} $(1 \le j \le l)$. For example, if $b[i] = 2$ for some i, we will check L_{w_i} starting from $I_{w_i}^2$. Initially, each entry in b is set to be 1. We also store Q as an array. Then, $Q[i]$ refers to w_i for $1 \le i \le l$.

ALGORITHM *interval-check*(Q, I, b)

begin

1. *mark* := *true*; $j := |Q|$; assume that $Q[1] \prec Q[2] \prec \ldots \prec Q[l]$;
2. **while** (*mark* = *true* and $j \ge 1$) **do** {
3. $(x, y) := con(Q[j], I, b[j])$; $b[j] := y$; /* $Q[j] = w_j$ */
4. **if** (x = *true*) **then** $j := j - 1$
5. **else** {*mark* := *false*;}
6. }
7. **if** (*mark* = *true*) **then** return (*true*, b)
8. **else** return (*false*, b);

end

The output of the algorithm is a pair (t, b'). If in each L_{w_j} $(1 \le j \le l)$ we can find an interval that contains I, t is *true*; otherwise, t is *false*. b' is an array satisfying the following properties:

(i) If t is *true*, each $b'[j]$ is an integer i showing that it is the ith interval in L_{w_j} that contains I.

(ii) If t is *false*, there exists j dividing b into three parts: $b'[1 .. j - 1]$, $b'[j]$, and $b'[j + 1 .. l]$ such that for any index k,

1. If $j + 1 \le k \le l$, then $b'[k]$ is an integer i such that ith interval in L_{w_k} contains I.
2. If $k = j$, then in L_{w_k} no interval is able to contain I and $b'[k]$ is $|L_{w_k}| + 1$ or a least number i such that $I_{w_k}^i[1] > I[2]$.
3. If $1 \le k \le j - 1$, then $b'[k]$ is the same as $b[k]$ (see line 5; the execution of this line will enforce the control to get out of the **while**-loop, and leave $b[1 .. j - 1]$ not updated.)

Lemma 4 Let (t, b') be the return value of *interval-check*(Q, I, b). Then, if t is *true*, b' satisfies property (i). Otherwise, b' satisfies (ii). □

The two properties (i) and (ii) described above are very important to the efficiency and correctness of our main algorithm to check whether $Q = \{w_1, w_2, \ldots, w_l\}$ is in the same document. Assume that $w_1 \prec w_2 \prec \ldots \prec w_l$. Its main idea is to find p, q such that $I^p_{w_{l-1}} \supset I^q_{w_l}$, and then use the above algorithm to check whether for each $w \in R = \{w_1, \ldots, w_{l-2}\}$ L_w has an interval containing $I^q_{w_l}$.

ALGORITHM *containment*(Q, b)

begin
2. let $|Q| = l$; assume that $Q[1] \prec Q[2] \prec \ldots \prec Q[l]$;
3. $R := \{Q[1], \ldots, Q[l - 2]\}$;
3. $p := b[l - 1]$; $q := b[l]$;
4. **while** $(p \le |L_{Q[l-1]}|)$ and $q \le |L_{Q[l]}|)$ **do** {
5. **if** $L^p_{Q[l-1]} \supset L^q_{Q[l]}$ **then** {
6. $(x, b) :=$ *interval-check*$(R, L^q_{w[l]}, b)$;
7. **if** $(x =$ *true*$)$ **then** {return (*true*, b);}
8. **else** $\{q := q + 1; b[l] := q;\}$
9. }
10. **else** {
11. **if** $(L^p_{Q[l-1]}[2] < L^q_{Q[l]}[1])$ **then** $\{p := p + 1; b[l - 1] := p;\}$
12. **else** $\{q := q + 1; b[l] := q;\}$
13. }
14. }
15. return (*false*, b);

end

The **while**-loop in the above algorithm is almost the same as the two-words checking (see 2.1). The only difference consists in lines 5 – 9. In the case of $L^p_{Q[l-1]} \supset L^q_{Q[l]}$, we will continually check whether there is an interval in each $L_{Q[j]}$ $(1 \le j \le l - 2)$ which contains $L^q_{Q[l]}$; but this is done simply by calling the algorithm *interval-check*( ) (see line 6.)

In addition, special attention should be paid to array b, whose values can also be utilized to indicate the checked intervals in every interval sequence. This enables us to avoid any redundancy when we want to find all the possible containment sequences by using this algorithm, which is required to evaluate conjunctive queries.

Example 2 Continued with Example 1. We will check two sets of words: $Q = \{f, a, p\}$ and $Q' = \{c, d, m, p\}$ to see whether each of them is in the same document.

For Q, we have $Q[1] = f \prec Q[2] = a \prec Q[3] = p$. Initially $b = \{1, 1, 1\}$ (i.e., b is an array containing three entries $b[1] = b[2] = b[3] = 1$). From Fig. 2, we see that $L_{Q[1]} = L_f = [1, 5][10, 13]$; $L_{Q[2]} = L_a = [1, 4][6, 8]$; and $L_{Q[3]} = L_p = [1, 1][10, 10]$.

In the 1st iteration of the **while**-loop, we will check $L^1_{Q[2]}$ against $L^1_{Q[3]}$. Since $L^1_{Q[2]} = [1, 4] \supset L^1_{Q[3]} = [1, 1]$, we will call *interval-check*(R, I, b), where $R = \{f\}$, $I = [1, 1]$, and $b = \{1, 1, 1\}$ (note that $b[2]$ and $b[3]$ will not be used in the execution of *interval-check*(R, I, b)). This call returns (*true*, $\{1, 1, 1\}$), which is used as the return value of *containment*(Q, b) (see line 7).

Now we consider $Q' = \{c, d, m, p\}$ with $c \prec d \prec m \prec p$. Again, initially $b = \{1, 1, 1, 1\}$; $L_{Q[1]} = L_c = [1, 9]$; $L_{Q[2]} = L_d = [6, 7][10, 12]$; $L_{Q[3]} = L_m = [1, 2][10, 11]$; and $L_{Q[4]} = L_p = [1, 1][10, 10]$. We will have the following working process.

1st iteration of the **while**-loop:
check $L^1_{Q[3]}$ against $L^1_{Q[4]}$. Since $L^1_{Q[3]} = [1, 2] \supset L^1_{Q[4]} = [1, 1]$, we will call *interval-check*$(R = \{c, d\}, I = [1, 1], b = \{1, 1, 1, 1\})$, which returns (*false*, $b = \{1, 1, 1, 1\}$). In this case, line 8 will be conducted (by which index q – index to scan $L_{Q[4]}$, will be increased by 1), and then in a next iteration we will check $L^2_{Q[4]}$.

2nd iteration of the **while**-loop:
check $L^1_{Q[3]}$ against $L^2_{Q[4]}$. Since $L^1_{Q[3]}[2] = 2 < L^2_{Q[4]}[1] = 10$, line 11 will be conducted (by which index p, - index to scan $L_{Q[3]}$, will be increased by 1), and in a next iteration we will check $L^2_{Q[3]}$.

3rd iteration of the **while**-loop:
check $L^2_{Q[3]}$ against $L^2_{Q[4]}$. Since $L^2_{Q[3]} = [10, 11] \supset L^2_{Q[4]} = [10, 10]$, we will call *interval-check*$(R = \{c, d\}, I = [10, 10], b = \{1, 1, 2, 2\})$, which returns (*false*, $b = \{3, 2, 2, 2\}$). In this case, line 8 will be conducted (by which index q will be increased by 1), which will get the execution out of the **while**-loop and *containment*(Q, b) returns (*false*, $\{3, 2, 2, 3\}$). □

Proposition 2 Algorithm *containment*( ) is correct. □

Proof. We only need to prove that values for b are correctly changed, since it guarantees that the return value of each call *interval-check*( ) is correct. We prove this by induction of the number k of *interval-check*( ) calls.

When $k = 1$, it is obviously correct since each entry $b[j]$ is set to 1.

Assume that when k it is correct, we will prove that by the $(k + 1)$th call b is also correctly changed. We first notice that if the return value of the kth call is (*true*, b) the $(k + 1)$th call will not be invoked. So we consider only the case that the return value of the kth call is (*false*, b). Assume that the kth call is of the form *interval-check*$(R, L^q_{Q[l]}, b)$. Then, the $(k + 1)$th call is of the form *interval-check*$(R, L^{q+1}_{Q[l]}, b')$, where b' is an array changed by the execution of *interval-check*$(R, L^q_{Q[l]}, b)$. In terms of the induction hypothesis, it is correct. Also, b' can be divided into three parts according to property (ii) shown above. From this, we can see that $L^{q+1}_{Q[l]}$ cannot be contained in the $(b'[j] - 1)$th interval in any $L_{Q[j]}$ $(1 \le j \le l - 2)$. From Lemma 3, we know that b' will be correctly changed by the execution of *interval-check*$(R, L^{q+1}_{Q[l]}, b')$. □

The above algorithm can be greatly improved as follows.

- *By checking sentinels*. Once the return value of a call *con*($R[j]$, $L^q_{w[l]}$, $b[j]$) is of the form (*false*, *y*) with *y* pointing to a sentinel, we can stop the whole process immediately as in this case, w_1, w_2, …, w_l cannot be in the same document.
- *By marking successful checkings*. Each time we find a containment sequence I_1, I_2, …, I_{l-1}, I_l such that $I_j \in L_{Q[j]}$ ($1 \le j \le l$) and $I_j \supset I_{j+1}$ ($1 \le j \le l$ - 1), we mark I_{l-1}. Then, we can find a next containment sequence I_1, I_2, …, I_{l-1}, I immediately, where I is an interval directly next to I_l in L_{w_l}, if $I_{l-1} \supset I$ and I_{l-1} is marked. In this way, each interval in all $L_{Q[j]}$'s can be visited at most two times by using the algorithm to find all the possible containment sequences.

We refer to the modified algorithm as *containment**(Q, b). However, due to space limitation, its formal description is omitted.

Proposition 3 The time complexity of *containment**(Q, b) is bounded by $O(\sum_{w \in Q} |L_w|)$. □

Finally, we notice that each L_w is sorted, and then we can use the binary or galloping search [5] to scan it. In this way, the average time complexity can be improved to $O(|L_{w_l}| + \sum_{w \in Q \setminus \{w_l\}} \log^2 |L_w|)$. We can also use the interpolation method to probe position in an interval sequence.

3.2 Evaluation of conjunctive queries

The containment-checking algorithm discussed in 3.1 can easily be adapted to evaluate conjunctive queries of the form $Q = w_1 \wedge w_2 \wedge \ldots \wedge w_l$ with $w_1 \prec w_2 \prec \ldots \prec w_l$. What needs to change is to find all the possible containment sequences for $\{w_1, w_2, \ldots, w_l\}$. This can simply be done by repeatedly calling the algorithm *containment**( ). Let $\boldsymbol{I}_1$, $\boldsymbol{I}_2$, …, $\boldsymbol{I}_m$ be all the found containment sequences. Let $\boldsymbol{I}_i = I_{i1}, I_{i2}, \ldots, I_{il_i}$ (i = 1, …, m). Then, the answer to Q should be $\delta(I_{1l_1}) \uplus \ldots \uplus \delta(I_{ml_m})$. Based on this analysis, we give the following algorithm for evaluating conjunctive queries.

ALGORITHM *con-evaluation*(Q)
begin
4. let $|Q| = l$; assume that $Q[1] \prec Q[2] \prec \ldots \prec Q[l]$;
5. **for** (j = 1 to l) **do** $b[j] := 1$;
6. $R := \Phi$; $i := 1$;
4. **while** ($i \le |L_{w[l]}|$) **do** { $(t, b) :=$ *containment**(Q, b);
5. **if** t = *true* **then** {
6. $R := R \uplus \delta(I^j_{w[l]})$; $b[l] := b[l] + 1$;
7. }
8. $i := b[l]$;
9. }
10. return R;
end

In the main **while**-loop (see line 4) of the above algorithm, we repeatedly call the algorithm *containment**( ) to find all the possible containment sequences. For each of them, a set of document identifiers can be determined and the disjoint union of all such sets makes up the result.

Obviously, the time complexity of the algorithm is bounded by $O(\sum_{w \in Q} |L_w|)$, but can be further improved by using the binary, or galloping search [5], as well as the interpolation probing [17].

Example 3 Continued with Example 1. Let $Q = f \wedge m \wedge p$. Then, the execution of *containment**( ) will find two containment sequences: $\boldsymbol{I}_1$ = [1, 5], [1, 2], [1, 1] and $\boldsymbol{I}_2$ = [10, 13], [10, 11], [10, 10]. The results is then $R = \delta([1, 1]) \uplus \delta([10, 10]) = \{1\} \uplus \{4\} = \{1, 4\}$. □

3.3 Evaluation of disjunctive queries

Based on the interval sequences associated with words, the disjunctive queries can also be evaluated efficiently and even more interesting. For ease of explanation, we first show how to evaluate a query of the form: $w \vee w'$. Then, the general case will be discussed.

Again, we assume that $w \prec w'$. Then, any interval in L_w cannot be contained in any interval in $L_{w'}$. However, some intervals in $L_{w'}$ may fall in some intervals in L_w. To find all the documents each containing either w or w', we need to merge any interval in $L_{w'}$ into L_w if it does not fall in any interval in L_w. As with the containment-checking algorithm, we will scan both L_w and $L_{w'}$ from left to right, but with some intervals in $L_{w'}$ possibly merged into L_w:

- Let $L_w = I^1_w, I^2_w, \ldots, I^k_w$. Let $L_{w'} = I^1_{w'}, I^2_{w'}, \ldots, I^{k'}_{w'}$.
- Step through L_w and $L_{w'}$ from left to right. Let I^p_w and $I^q_{w'}$ be the intervals currently encountered. We will do the following checkings:
  (1) If $I^p_w \supset I^q_{w'}$, move to $I^{q+1}_{w'}$ if $q < k'$. If $q = k'$, go to (4).
  (2) If $I^q_{w'}[2] < I^p_w[1]$, insert $I^q_{w'}$ into L_w just before I^p_w. If $q < k'$, move to $I^{q+1}_{w'}$; otherwise ($q = k'$), go to (4).
  (3) If $I^p_w[2] < I^q_{w'}[1]$, move to I^{p+1}_w if $p < k$. If $p = k$, append $I^q_{w'}, \ldots, I^{k'}_{w'}$ to the end of L_w and then go to (4).
  (4) Let $I_1, \ldots, I_{k''}$ be all the intervals in the changed L_w. Return $\delta(I_1) \uplus \ldots \uplus \delta(I_{k''})$.

We denote this procedure as $L = merge(L_w, L_{w'})$.

Example 4 Continued with Example 1. Let $Q = d \vee m$. We have $d \prec m$. By using the above procedure to merge L_m = [1, 2][10, 11] into L_d = [6, 7][10, 12], we will get a new sequence: [1, 2][6, 7][10, 11]. So, the result is $\delta([1, 2]) \uplus \delta([6, 7]) \uplus \delta([10, 12]) = \{1\} \uplus \{3\} \uplus \{4\} = \{1, 3, 4\}$. In the first step, we compare I^1_d = [6, 7] and I^1_m = [1, 2]. Since $I^1_d[1] = 6 > I^1_m[2] = 2$, I^1_m will be inserted into L_d just before I^1_d. Then, in the second step, we will compare I^1_d and I^2_m. Since $I^1_d[2] = 7 < I^2_m[1] = 10$, we will move to I^2_d. Next, in the third step, we compare I^2_d and I^2_m, and find $I^2_d \supset$

I_m^2. Since I_m^2 is the last interval in L_m, we terminate the merging process and return the result. □

Fig. 6 shows the entire merging process.

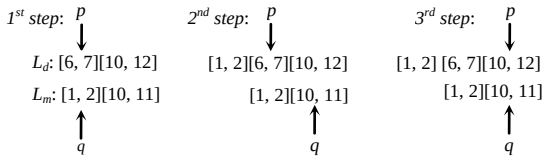


Figure 6: A merging process

This merging process can easily be extended to a general algorithm to evaluate disjunctive queries of the form $Q = w_1 \vee w_2 \vee \ldots \vee w_l$ with $w_1 \prec w_2 \prec \ldots \prec w_l$, as shown below.

ALGORITHM *dis-evaluation*(Q)

begin

1. let $|Q| = l$; assume that $Q[1] \prec Q[2] \prec \ldots \prec Q[l]$;
7. $L := L_{Q[1]}$;
8. **for** (i = 2 to l) **do** {
9. $L := merge(L, L_{Q[i]})$;
5. }
6. let $L = I_{1, \ldots, } I_k$;
7. return $\delta(I_1) \uplus \ldots \uplus \delta(I_{k''})$;

end

In the above algorithm, we use *merge*( ) to merge $L_{Q[i]}$ for i = 2, …, l into $L_{Q[1]}$ one by one. The running time is obviously bounded by $O(l \cdot r)$, where r is the largest number of intervals in all $L_{Q[i]}$'s which are not contained in each other. Again, the time requirement can be improved by using the binary search, the galloping search, and the interpolation probing.

4 Experiments

In the experiments, we have tested four methods:
Signature trees [2] (*ST* for short),
Inverted files [1] (*IF* for short),
Set intersection [4] (*SI* for short),
Interval based method (discussed in the paper; *IbM* for short).

All our experiments are performed on a 32-bit Windows operating system. The processor is Intel Core 2 Duo CPU with 4GB RAM. All index techniques are implemented by C++ and compiled by Microsoft Visual Studio 2010. We use the function *QueryPerformanceCounter()* from the *Kernel32.lib* library to measure the *CPU time*, which provides a high-precision timing (microsecond precision) on the Windows Platform.

- *Data sets*

To test the effectiveness of our index, we use a sample Web corpus, which contains one million text documents. We numbered the documents as they were stored, by assigning them a sequential number indicating their order in the indexing process. The characteristics of this collection are shown in Table 2.

Table 2: Characteristics of Web

	Web
Documents	1,000,000
Size (gigabytes)	7.5
Word occurrences (without markup)	3,603,556
Distinct words (after stemming)	285,344

- *Index construction time and sizes*

In Table 3, we show the time for constructing different indexes and their sizes. For this test, each document identifier and each interval occupy 4 bytes. For our method, the threshold ζ is set to be 1/1000. That is, only for those words w appearing in more than 100 documents an interval sequence will be established.

Table 3: Index construction time and size

	IF	*SI*	*ST*	*IbM*
Time (ms)	8,755	8,755	153,847	52,861
Size (MB)	14	14	20	14.4

From this table, we can see that the inverted file has the best time and space requirement than the other two methods. However, the space requirement by our method is just a little bit worse than the inverted method. For *SI*, they are exactly the same as *IF*.

- *Time of conjunctive queries*

In Fig. 6, we show the number of page access and the elapsed times for evaluating conjunctive queries containing different number of words. For this test, all the words are chosen randomly, but appear in more than 100 documents since only for such words the interval sequences are created. In addition, the page size is set to be 4KB. For the inverted file, a *melding algorithm* [5, 6] is used for doing the set intersection, which intersects the inverted lists two at a time in increasing order by size, starting with the two smallest. Also, it performs a binary search to determine whether a document identifier in the first list appears in the second list.

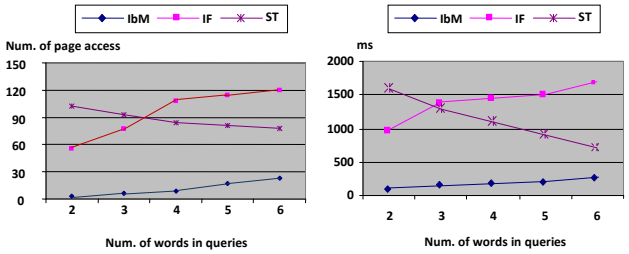


Figure 6. Test of conjunctive queries with page size 4KB

For each query, we average the running time over 20 executions.

From Fig. 6, we can see that our method is much better than both the inverted file and the signature file. Even the signature tree beats the inverted file. Especially, as the number of words in queries increases, both the number of

page access and the time of the signature tree decease. It is because a query signature is formed by superimposing (bit-wise OR) all the signatures of the words in a query. So, the more words in a query, the more 1's in a query signature, which will lead to less nodes to be explored in a signature tree. *SI* is an in-memory algorithm, not run for this test.

In Fig. 7, we show the results when the page size is set to be 12KB. From this, we can see that although the number of page access has been reduced, the time used is almost for all the three tested methods.

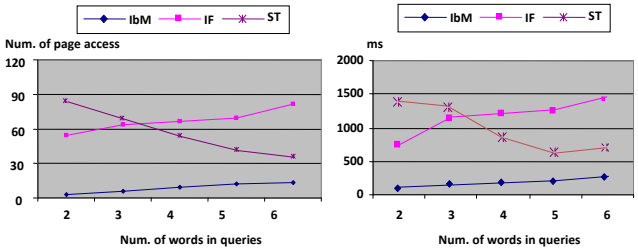


Figure 7. Test of conjunctive queries with page size 12KB

In Fig. 8, we show the test results when the whole index structure is accommodated in main memory for all four different methods.

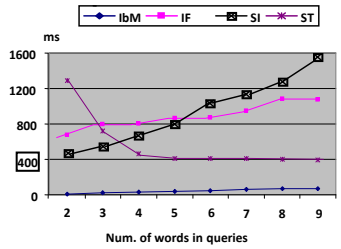


Figure 8. Conjunctive queries with whole index in main memory

- *Time of disjunctive queries*

In Fig. 9, 10, and 11, we show the test results for disjunctive queries, for which the signature file is not tested since it is totally not suitable for this task.

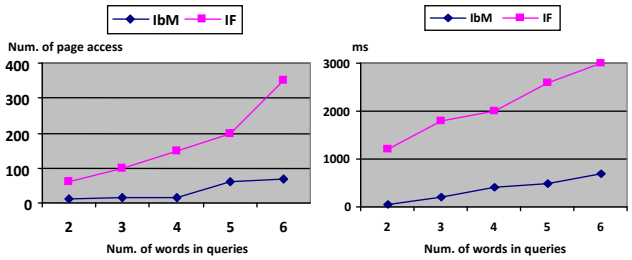


Figure 9. Test of disjunctive queries with page size 4KB

From these figures, we can see that more time is needed to evaluate a disjunctive query than a conjunctive for both the inverted file and ours. However, the discrepancy between these two kinds of queries for the inverted file is larger than for ours. It is because by the inverted file the normal set union is used with not much optimality being made. In the opposite, by ours the interval containment checking still works quite well even though the binary or galloping search has not been utilized.

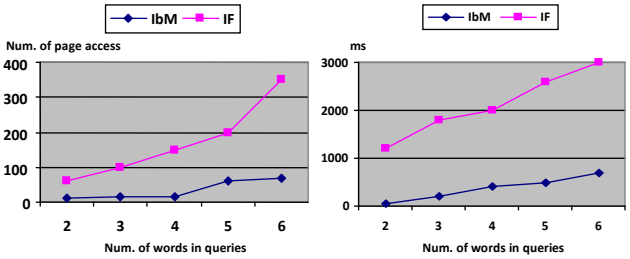


Figure 10. Test of disjunctive queries with page size 12KB

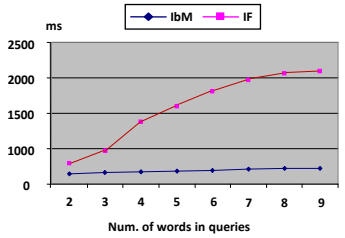


Figure 11. Test of disjunctive queries with whole index in main memory

5 Conclusion

In this paper, a new method is discussed to evaluate both conjunctive and disjunctive queries. The main idea is to transform an evaluation of queries to a series of reachability checkings, which improves the traditional method by an order of magnitude or more.

6. References

[1] Anh, V.N. and A. Moffat, A, 2005. Inverted index compression using word-alinged binary codes, *Kluwer Int. Journal of Information Retrieval 8, 1*, pp. 151-166.

[2] Chen, Y. and Chen, Y.B. 2006. On the Signature Tree Construction and Analysis, *IEEE TKDE*, Vol.18, No. 9, pp 1207 – 1224.

[3] Y. Chen and Y.B. Chen. An Efficient Algorithm for Answering Graph Reachability Queries, in *Proc. 24th Int. Conf. on Data Engineering (ICDE 2008)*, IEEE, April 2008, pp. 892-901.

[4] B. Ding, A.C. König, Fast set intersection in memory, *Proc. of the VLDB Endowment, v.4 n.4*, p.255-266, January 2011.

[5] Faloutsos, C. 1985. Access Methods for Text, *ACM Computing Surveys*, vol. 17, no. 1, pp. 49-74.

[6] Faloutsos, C. and Chan, R. 1988. Fast Text Access Methods for Optical and Large Magnetic Disks: Designs and Performance Comparison, *Proc. 14th Int'l Conf. Very Large Data Bases*, pp. 280-293.

[7] D.E. Knuth, *The Art of Computer Programming, Vol. 3*, Massachusetts, Addison-Wesley Publish Com., 1975.

[8] R. Lempel and S. Moran, Predictive caching and prefetching of query results in search engines, in *Proc. the World Wide Web Conf.*, Budapest, Hungary, ACM, 19-28, 2003.

MPGM: A Mixed Parallel Big Graph Mining Tool

Ma Pengjiang[1]
mpjr_2008@163.com
Liu Yang[1]
liuyang1984@bupt.edu.cn

Wu Bin[1]
wubin@bupt.edu.cn
Wang Hongxu[1]
513196584@qq.com

[1]School of Computer Science, Beijing University of Posts and Telecommunications, Beijing 100876, China

Abstract—*The design and implementation of a scalable parallel mining system for big graph analysis has proven to be challenging. In this study, we propose a parallel data mining system for analyzing big graph data generated on a bulk synchronous parallel (BSP) computing model and MapReduce computing model named mixed parallel graph mining (MPGM). This system has four sets of parallel graph mining algorithms programmed in the BSP parallel model and one set of data extraction-transformation-loading (ETL) algorithms implemented in MapReduce and a well-designed workflow engine optimized for Cloud computing to invoke these algorithms. Experimental show that the components of graph mining algorithm in MPGM are efficient and can make realistic application easy.*

Keywords- Cloud computing; parallel algorithms; graph data analysis; data mining; social network analysis

1 Introduction

Graphs are the most widely used abstract data structures in the field of computer science, and they offer a more complex and comprehensive presentation of data compared to link tables and tree structures. Many real application issues need to be described with graphical structure, and the processing of graph data is required in almost all cases, such as the optimization of railway paths, prediction of disease outbreaks, the analysis of technical literature citation networks, emerging applications such as social network analysis, semantic network analysis, and the analysis of biological information networks.

The graph mining theories and technique have been improved all the time. However, as the information time comes along, which has led to explosive growth of information, the scale of graph-based data has increased significantly. For example, in recent decades, with the popularity of the Internet and the promotion of Web 2.0, the number of webpages has undergone rapid growth. Based on statistics provided by the China Internet Network Information Center (CNNIC), at the end of December 2013, the number of webpages in China had reached 150.0 billion, 22.2% increase over last year. Simultaneously, the number of micro-blog users accounted for 54.7% of all Internet users, which is approximately 308 million. This phenomenon highlights the scale of big graph data come into being, and it is challenging job to perform efficient analysis of these data. To solve the large scale graph analysis task, we have built a system called MPGM which provides a series of parallel graph mining algorithms based on the BSP parallel computing model. While the process of computing, the data is always stored in memory of cluster, this mechanism helps BSP model achieved a high performance, but limited the scale of data that the system can handle. Therefore, MPGM adds a set of data extract-transformation-loading algorithms based on MapReduce to improve the data processing capacity when the cluster scale is limited. Tests on real mobile communication networks data show that our improvement is reliable and highly- efficient.

The remainder of this paper is structured as follows. Section 2 reviews related works. And then describes MPGM's system architecture in Section 3. The application example is presented in Section 4. Some performance measurements are reported in Section 5. Finally we will discuss future directions.

2 Related Work

MGPM is closely related to parallel computing platforms and graph mining tools. Here, we briefly summarize those related works.

Parallel computing platforms have been studied for a long time, and they can be roughly categorized into three types: (i) based on the MapReduce model, (ii) based on the message passing interface (MPI) model, and (iii) based on the BSP model. The MapReduce [1] model was proposed by Google, and the most famous and successful open-source implementation is Hadoop. The MapReduce model is extreme suitable for process large scale data, and algorithms that do not have many iterations, but it has a bad performance in high iterative algorithm, which means that they are not suitable for most graph algorithms.

The MPI [2] model provides a model for message passing, and many companies and universities have implemented jobs that can be run on almost any type of parallel computer, which support all existing graph algorithms. However, because the MPI model uses a communication method to integrate computing resources, this model has several drawbacks, for example, the low efficiency of parallel computing and the high consumption of memory makes it difficult to manage the resources and communication in detail.

BSP [3] is also a widely used parallel computing framework. BSP improved the weakness exhibited by MapReduce, and performs well when a program has a large number of iterations or requires a lot of communication. A BSP program can be divided into several super-steps, each of which consists of three ordered stages: local computation, communication, and barrier synchronization. A BSP system is composed of a number of computers with local memory and disks. Each computer can run several computing processes called peers. In the local computation stage, each peer is computed using locally stored data. After finishing local computation, each peer can communicate only necessary data to other peers. When a peer finishes the communication stage, it will wait until all the peers reach the barrier synchronization and a super-step is completed.

Popular parallel data mining tools include the following things. Mahout [4], which is supported by the Apache Foundation, supply classification, clustering, pattern mining, regression, and dimension reduction and other machine learning algorithms, but lacks the graph mining function. GraphLab [5] improves on the MapReduce abstraction by compactly expressing asynchronous iterative algorithms with sparse computational dependencies. However, there may be problems while implied a synchronous iterative graph algorithm. PEGASUS [6] is an open-source large graph mining system implemented on Hadoop. The key idea of PEGASUS is to convert graph mining operations into iterative matrix-vector multiplication. While it supports large-scale graph data, in practice, not all of the graph mining algorithms can be modeled by matrix-vector multiplications. Dryad [7] is a general parallel computing platform proposed by Microsoft Research, which abstracts the computing and communication in data mining operations into vertexes and edges to form a dataflow graph. The platform executes the vertexes on work nodes and refines the dataflow graph to optimize the running process. Big Cloud parallel data mining (BC-PDM) [8] was developed by China Mobile Research Institute (CMRI), and it provides visualization operations for data mining and the analysis of graph data. However, it is based on Hadoop, and the graph mining algorithms therefore cannot achieve a high level of performance. Pregel [9], which was motivated by BSP and implemented by Google, provides a complete solution for large-scale graph computing, but it has not been published in the public domain. BC-BSP [10] is another implementation of the BSP parallel platform. While most BSP platforms use memory to exchange the temporary data, BC-BSP designed a mechanism of spill data (including static data and dynamical data) on the local disk to improve the data processing capacity when the cluster scale is limited, but the management and updating of this data spill mechanism requires extra communication and system resources, while introducing new defects to the platform.

3 System Architecture

This system focuses on big graph management and graph mining. We noticed that, while the original data is huge, but most graph mining application using part of the original data. In response to this feature, we use MapReduce to extract the graph data from original data, and construct the graph. To manage graph data and original data, we designed a data I/O management component in the parallel platform layer and a data management component in the logical layer. The algorithm layer divided into two parts, the ETL algorithm set and graph mining algorithm set. In the graph mining field, graph pattern mining, graph clustering mining, graph classification mining, and dynamic graph mining are the most popular topics. We built the graph mining algorithm set to implement graph clustering mining and graph classification mining algorithms, and the graph attribute analysis as the foundation of the graph analysis. Finally, we made this system extendable to enable the addition of other algorithm components.

An overview of the architecture of MPGM is presented in Figure 1. The system consists of four layers. The function of each layer is described as follows:

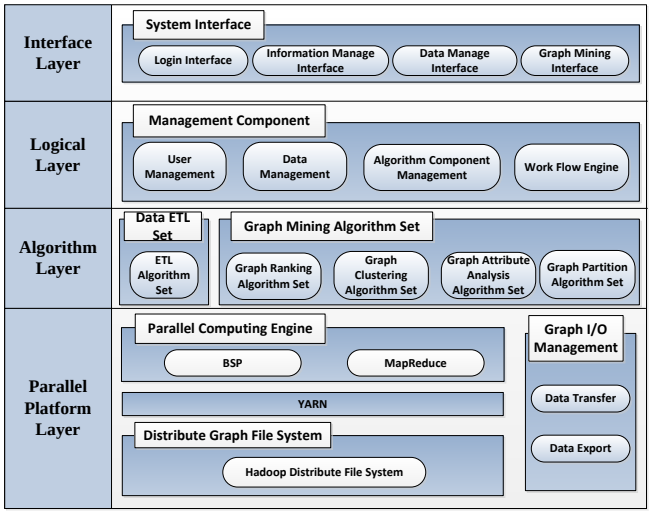


Figure 1: Architecture of PGM

3.1 The Parallel Platform Layer

The Parallel Platform Layer comprises four components: a Distributed Graph File System, YARN, Parallel Computing Engine, and Graph I/O Management component. We used the Hadoop Distributed File System (HDFS) to construct the Distributed Graph File System, enabling the storage of big graph data. YARN, a framework for cluster resource management and job scheduling, comprises a ApplicationMaster (AM) which should integrates multiple computing frameworks, e.g. MR, BSP. Because the BSP model achieves a high performance in graph mining algorithms, we chose Hama BSP [11] as the parallel computing engine, and used it to handle message communication, data distribution, and fault tolerance, and use MapReduce as the graph data pre-processing engine for extract graph information from original data. The Graph I/O Management component is responsible for the transfer of data from the database into the graph data form that the MPGM can handle, and it then exports the resulting MPGM data.

3.2 The algorithm layer

The algorithm layer is the main layer of MPGM. This layer can be roughly divided into 2 parts. The ETL algorithm set and graph mining algorithm set. In the graph mining algorithm set, we implemented four sets of 20 graph mining algorithm components in the BSP parallel model and four group of data ETL algorithm for transform original into graph data. The ETL algorithm set is composed of data cleaning set for detect and remove error value, data transform set for transform value into the format we need, data extract set and data update set. Those graph mining algorithm components can be divided into four sets. The graph ranking set comprises PageRank [12], HITS [13], and RWR [14] algorithms components, the graph clustering set comprises GN [15], CNM [16], CPM [17], and LPA [18] algorithm components, and the K-means algorithm is used for the processing of general data. The graph attribute analysis set contains the graph diameter, closeness centrality, clustering coefficient, network density, betweenness centrality, and five other algorithm components, while the graph partition set contains components that are based on the MSP [19] algorithm component and the Metis [20] algorithm component. Those algorithm components can be run either in the console, or can be invoked in a user-defined workflow from the user interface.

3.3 The logical layer

The logical layer is based on Open Service Gateway Initiative(OSGi), to implement a stable and efficient scalable system, which is service platform and manager all kinds of services that are supported by this layer, such as User Management Service, Data(HDFS) Management Service , Algorithm Service (Each algorithm is a kind of service), and the workflow engine Service.

Figure 2 shows the operation of the logical layer. Getting the start command from tomcat servlet which is a container in the interface layer, OSGi container will execute a train of operations, starting the life cycle, activating and registering the each service that hosted in bundle-services management. With the tomcat servlet invoking one of services, Service Register queries and gets service that is requested by the tomcat from the services pool, and then Execution Environment (EE) calls the workflow engine service to perform the requested service which is ultimately implemented in the algorithm layer. There are two important features:

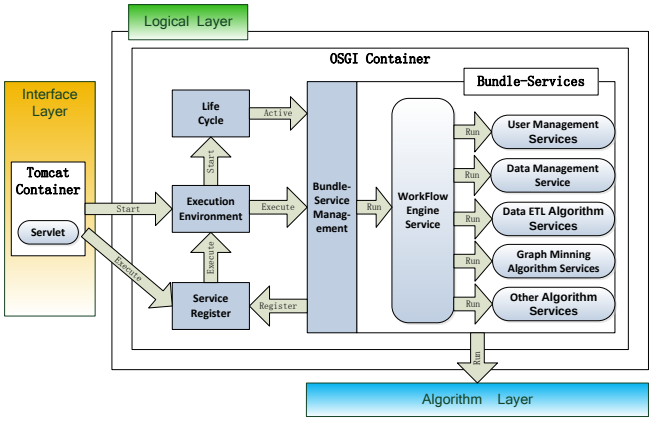


Figure 2: Operation flow of the Logical Layer

3.3.1 Hot- plugging of Bundle-Services.

Each service that provided by Bundle-Services Management is a plug-in or bundle with highly cohesion, low coupling features. With the independent class loader, OSGi prevents the external system accessing to detail of the bundle, just exposing externally callable interfaces, and provides a dynamic service management strategy. In other words, receiving the operation from the interface layer, Execution-Environment can dynamically install and uninstall the bundle-services, without shutting down the system.

3.3.2 High scalability and flexibility of Workflow Engine.

The workflow engine implemented in this system abstracts data-intensive computing into an orderly and plain workflow instance. Meanwhile, the workflow engine defining a set of unified interfaces can be seamless integrated with multiple computing frameworks, e.g. MR, BSP. Figure 3 shows an example workflow instance which includes three algorithms that can be divided into two categories, based on

MR and based on BSP. When the interface layer sends a command to execute, Workflow Engine automatically analyzes the workflow and orderly executes the each component. Workflow Engine also achieve computing operation monitoring and provides an flexible configuration .

3.4 The interface layer

The interface layer is built in HTML, and flex provides an interactive interface with which the user can login to the MPGM, management information, and most importantly, use the graph algorithms' mining graph data.

4 Application Example

Here we use the key user of mobile communication discovery as an example to demonstrate our plat-form.

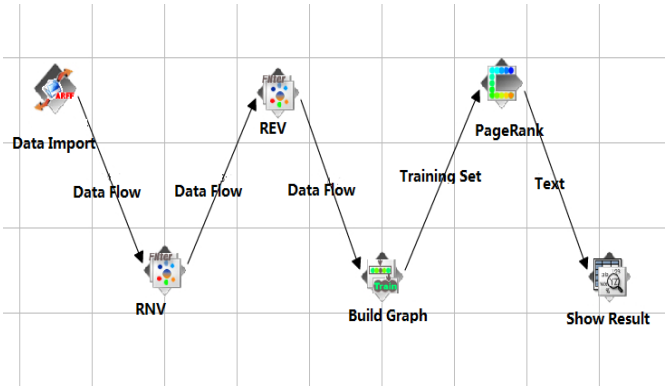


Figure 3: Operation flow of the Key User Discovery application

The discovery of key person in mobile communication is an important and valuable application. It is trying to find a number of users that influence people around them called the key user from communication information data with graph mining algorithm. The experience proven that delivery advertisement or making market strategy direct to those key users is more effective. The graph can be constructed from user mobile phone call record, and apply PageRank algorithm on the graph data, then the user with higher rank value has the higher influence to other users which means a key user.

However, in realistic application there is no purely graph data to make PageRank running on it, and the scale may be too large for our BSP platform on the cluster. The call information record has number of call and called user, the phone duration the phone happening time and data, some of user number cloud be null as they belong to other mobile operator. So we need to clean the data, select the call duration longer than 5 seconds, the less may be spam call, and get the purely graph data. The operation flow of this application in our platform is showed in Figure 3. The flow starts with data import, and removes null value (RNV) to ignore the other mobile operator users, then removes extreme value (REV) to eliminate the spam call, and then builds graph and runs PageRank on the graph, finally list the users in order of decrease PageRank value.

The running result proved that the ETL operation in MapReduce can handle 1.3GB call information record and the extracted graph data is about 150MB, and the whole operation takes 2633seconds on our 4 computing nodes cluster. The single BSP platform can't handle the original size of data, and the single MapReduce platform can't finish computing so fast.

5 Performance

We have tested the MPGM for its functionality, reliability, usability, efficiency, maintainability, and portability. The evaluation was performed on clusters having 9 nodes, where each node consists of 2 Intel(R) Xeon(R) CPU E5530, 48 GB main memory and 1024 GB hard drive. The evaluation data is a randomly generated graph data set scale ranging from 10,000 edges to 2000,000 edges. We also deployed a BC-PDM on the same cluster and run some social network analysis algorithms using Google web data. Some of the results are presented in Figure 4. Finally, we compared MPGM and BC-BSP with the PageRank algorithm on a 4-node cluster, but where the nodes have the same hardware. The results are recorded in Figure 5. The characteristics of these graphs' data are shown in Table 1.

Table 1. Networks Basic Structural Properties

Name	|N|	|E|	Type
data_set_1	17500	100000	Random
data_set_2	72000	500000	Random
data_set_3	175000	1000000	Random
data_set_4	720000	5000000	Random
data_set_5	1750000	10000000	Random
data_set_6	3500000	20000000	Random
GoogleWeb	875713	5105039	Web

Table 2 show that most graph mining jobs can be accomplished in a short time and benefit from well-designed architecture. Also, the MPGM has a higher performance than BC-PDM and BC-BSP.

Table 2. Runtime of some Graph Mining Algorithm Components(Second)

Graph Data Set	Eigenvector Centrality Measure	InDegree Count	MSP	PageRank	Closeness Centrality	Personal Centrality	Clustering Coefficient	RWR
data_set_1	16.2	13.2	166.1	25.2	31.2	13.1	13.1	22.2
data_set_2	25.1	16.2	310.1	40.2	70.5	16.1	19.1	28.1
data_set_3	28.2	22.1	343.0	61.5	31.4	19.1	19.2	37.2
data_set_4	88.2	64.2	696.1	202.4	25.4	22.0	31.1	169.2
data_set_5	173.6	73.2	995.2	439.6	32.1	31.2	55.2	313.4
data_set_6	643.7	199.3	1278.3	1241.6	55.7	52.2	151.2	688.7
GoogleWeb	199.2	79.2	721.3	304.6	31.7	34.2	52.2	331.5

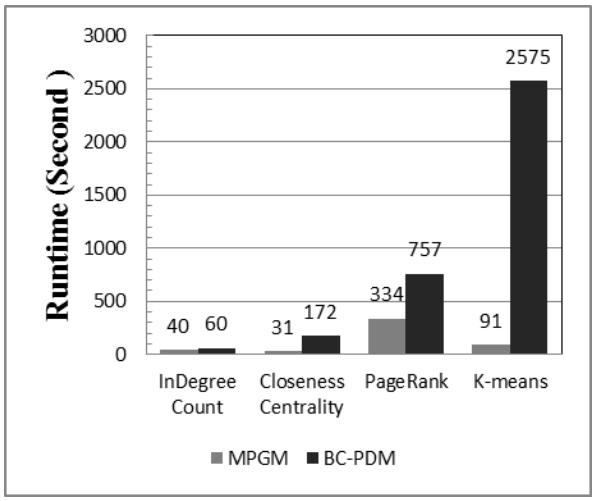


Figure 4: Comparison of MPGM and BC-PDM on data_set_5

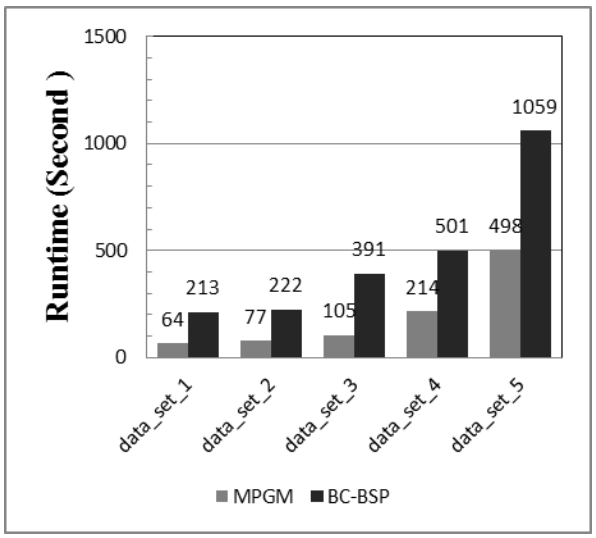


Figure 5: PageRank Performance of MPGM and BC-BSP

6 CONCLUSION

In this study, we introduced MPGM based on Cloud computing. It has the ability to analyze big graph data and achieved a better performance than the Hadoop-based data mining tools BC-PDM and BSP-based parallel platform BC-BSP. We expected to mix more parallel computing model to achieve a higher performance of graph mining both in data scale and computing speed.

7 Acknowledgment

This work is supported by the National Key Basic Research and Department (973) Program of China (No.2013CB329603) and the National Science Foundation of China (Nos.61375058,and 71231002). This work is also supported by the Special Coconstruction Project of Beijing Municipal Commission of Education.

8 References

[1] J. Dean, and G. Sanjay, MapReduce: Simplified data processing on large clusters, Communications of the ACM., vol. 51, no. 1, pp. 107-113,2008.

[2] S. Marc, S. W. Otto, D. W. Walker, J. Dongarra, and S. Huss-Lederman, MPI: The Complete Reference. MIT press, 1995.

[3] L. G. Valiant, A bridging model for parallel computation, Communications of the ACM., vol. 33, no. 8, pp. 103-111, 1990.

[4] S. Owen, A. Robin, T. Dunning, and E. Friedman, Mahout in Action. Manning, 2011.

[5] Y. Low, J. Gonzalez, A. Kyrola, D. Bickson, C. Guestrin, and J. M. Hellerstein, Graphlab: A new framework for parallel machine learning, arXiv preprint., arXiv:1006.4990, 2010.

[6] U. Kang, C. E. Tsourakakis, and C. Faloutsos, Pegasus:A peta-scale graph mining system implementation and observations, in Data Mining, 2009. ICDM'09. Ninth IEEE International Conference on, 2009.

[7] M. Isard, M. Budiu, Y. Yu, A. Birrell, and D. Fetterly, Dryad: Distributed data-parallel programs from sequential building blocks, ACM SIGOPS Operating Systems Review, vol. 41, no. 3, pp. 59-72, 2007.

[8] L. Yu, J. Zheng, W. Shen, B. Wu, B. Wang, L. Qian, and B. Zhang, BC-PDM: Data mining, social network analysis and text mining system based on cloud computing, presented at the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2012.

[9] G. Malewicz, M. H. Austern, A. J. C. Bik, J. C. Dehnert, I.Horn, N. Leiser, and G. Czajkowski, Pregel: A system forlarge-scale graph processing, in Proceedings of the 2010ACM SIGMOD International Conference on Management of Data, 2010.

[10] Y. Bao, Z. Wang, Y. Gu, G. Yu, F. Leng, H. Zhang, B. Chen, C. Deng, and L. Guo, BC-BSP: A BSP-based parallel iterative processing system for big data on cloud architecture, in Proc.Database Systems for Advanced Applications, Springer Berlin Heidelberg, 2013.

[11] S. Seo, E. J. Yoon, J. Kim, S. Jin, J. Kim, and S. Maeng, Hama: An efficient matrix computation with the mapreduce framework, in Cloud Computing Technology and Science (CloudCom), 2010 IEEE Second International Conference on, 2010.

[12] L. Page, S. Brin, R. Motwani, and T. Winograd, The PageRank citation ranking: Bringing order to the web, 1999.

[13] http://malt.ml.cmu.edu/mw/index.php/Random walk with restart.

[14] J. M. Kleinberg, Authoritative sources in a hyperlinked environment, Journal of the ACM (JACM), vol. 46, no. 5, pp. 604-632, 1999.

[15] M. Girvan and M. E. J. Newman, Community structurein social and biological networks, Proceedings of theNational Academy of Sciences, vol. 99, no. 12, pp. 7821-7826, 2002.

[16] A. Clauset, M. E. J. Newman, and C. Moore, Finding community structure in very large networks, Physical Review E, vol. 70, no. 6, 2004.

[17] M. E. J. Newman and M. Girvan, Finding and evaluating community structure in networks, Physical Review E, vol. 69, no. 2, 2004.

[18] U. N. Raghavan, R. Albert, and S. Kumara, Near linear time algorithm to detect community structures in largescale networks, Physical Review E, vol. 76, no. 3, 2007.

[19] Z. Zeng, B. Wu, and H. Wang, A parallel graph partitioning algorithm to speed up the largescale distributed graph mining, in The 1st International Workshop on Big Data, Streams and Heterogeneous Source Mining: Algorithms, Systems, Programming Models and Applications, 2012.

[20] G. Karypis and V. Kumar, Metis-unstructured graph partitioning and sparse matrix ordering system, version 2.0, 1995.

[21] J. Yang and D. Zhang, Lightweight workflow engine based on Hadoop and OSGI, presented at the 5th IEEE International Conference on Broadband Network & Multimedia Technology, Beijing, China, 2013.

Spatial Association Mining of Focal Events in Cloud Computing

Sang Jun Park and Jin Soung Yoo†
Department of Computer Science
Indiana University-Purdue University, Fort Wayne, Indiana, USA
{parks07,yooj@ipfw.edu}

***Abstract*—Spatial association mining is the process of discovering interesting relationship and correlation patterns from spatial and spatiotemporal data. This work concerns finding the spatial association patterns of a certain focal event from massive spatial data. Explosive growths in geospatial data, followed by the emergence of social media and location sensing technologies, have emphasized the need to develop new and computationally efficient methods for analyzing big spatial data. To carry out computationally expensive spatial association mining tasks, we used modern computational frameworks facilitating the distributed executions of massive tasks. This work presents a MapReduced-based parallel and distributed algorithm to discover the association patterns of a focal spatial event in a cloud computing environment. The performance of the proposed algorithm was extensively evaluated on real clusters with Hadoop run-time environment. The experimental result shows that the proposed method is scalable with respect to various workload factors including data size and neighborhood size, and achieves a significant improvement in computational speed with an increase of cluster nodes.**

***Index Terms*—association mining, spatial relationship patterns, cloud computing, MapReduce**

I. Introduction

"Big data" is defined by three characteristics: volume, in terms of large-scale data storage and processing; variety, the availability of data in different types and formats; and velocity, the fast rate of new data acquisition [1]. This so-called "Big data" is a reality of today's world and brings not only huge amounts of data but also a variety of data types to organize and analyze. Collected from numerous sources including GPS tracking systems, mobile phones, social media, environment observation sensors, outbreaks of disease, and crime logs, rich spatial data with geo-location and time tags are considered invaluable nuggets of information [2]. Finding the solution that is able to translate the plentiful amount of spatial data that surrounds us into meaningful and useful information has led to the rise of spatial data mining. Spatial data mining is a process to search interesting and previously unknown, yet potentially useful patterns in large spatial data.

Spatial association mining is one of core spatial data mining tasks and aims to discover correlations and interesting relationships among spatial features and/or events in a large spatial database. A spatial association pattern can be represented in terms of spatial and non-spatial predicates. For instance, is_a(x, robbery) $\wedge$ within(x, zip code 46807) $\wedge$ close_to(x, school) (0.65), implies "*There is high chance (65%) of the occurrence of robbery in the nearby area of schools in a region of zip code 46807*." The association patterns with spatial information is especially useful and beneficial to data analysts and decision makers as they attempt to understand the underlying spatial relationships of their data, and has broad applications to numerous fields including location-based services, criminology, public health and climatology. This work focuses on the discovery of spatial association patterns of a certain focal event.

Spatial association pattern mining gives several challenges with big data. A spatial relationship between two objects is represented with a spatial predicate (e.g., close_to) in the spatial association pattern. The computation of spatial relationships is inherently demanding of both the computational processing time and memory requirements. Furthermore, the large data volumes have outgrown the processing capabilities of a single host. Therefore, this work proposes to distribute and parallelize the spatial association mining process in order to deal with spatial data at a massive scale.

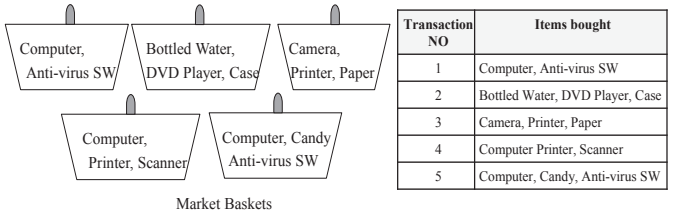
In the general data mining area, researchers have considered high-performance parallel and distributed computing in order to speed up the process of mining frequent itemset patterns in the ever-increasing transactional databases [3]. The distributed/parallelized mining process attempts to divide the mining problem into smaller ones and to solve these sub-problems using homogeneous computing nodes that may work independently and simultaneously. Modern frameworks that currently facilitate the distributed executions of massive tasks have become increasingly popular since the advent of the MapReduce programming model, the Hadoop execution framework and distributed file systems [4]. Thus this study presents an algorithmic work for spatial association pattern discovery on the modern framework.

The remainder of this paper is organized as follows. Section II describes the basic concept of frequent itemsets mining, spatial association pattern, and the MapReduce paradigm. Section III describes the problem statement and related work. Section IV presents the proposed algorithm based on the MapReduce. The experimental results of the proposed work are reported in Section V. The last Section VI includes conclusion.

II. Background Concepts

The association pattern mining problem was originally developed to find interesting relationships among items in the

The authors are listed in the alphabetical order of last name. † represents the corresponding author.



Transaction NO	Items bought
1	Computer, Anti-virus SW
2	Bottled Water, DVD Player, Case
3	Camera, Printer, Paper
4	Computer Printer, Scanner
5	Computer, Candy, Anti-virus SW

Fig. 1. Market-basket transaction data

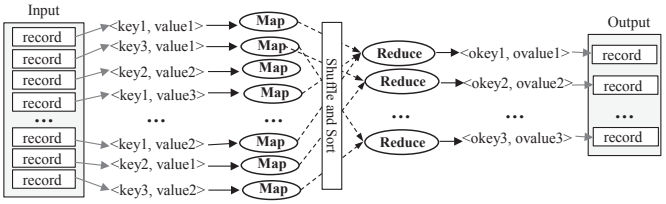


Fig. 2. MapReduce programming model

transaction records of businesses [5]. To begin, we describe the market-basket model of data to explain the innate problem of not only frequent itemsets, but also the spatial association patterns. Afterward, the MapReduce paradigm will be introduced.

A. Frequent Itemsets Mining

In frequent itemset mining, the problem that needs to be solved is that of discovering sets of items that appear in (and are related to) many of the same market baskets [6]. For example, Figure 1 shows market-basket data, often referred to as *transactions*. Each basket consists of a set of items. A set of items that appears in many baskets is said to be 'frequent'. To be formal, we assume that there is a number s, called the *support threshold*. So if I is a set of items (an itemset), the support count of I is the number of baskets for which I is a subset. Thus, we say I is frequent if its support count is s or more. In the example of Figure 1, among the single sets, {Computer} and {Anti-virus SW} are quite frequent. 'Computer' appears in baskets 1, 4 and 5, therefore its support count is 3. 'Anti-virus SW' on the other hand, appears in baskets 1 and 5, so its support count is 2. Next, we see a doubleton {Computer, Anti-virus SW}. It appears in basket transactions 1 and 5. Therefore, its support count is 2. Last, we see a triple {Computer, Anti-virus SW, Candy}, which appears only in transaction 5, so its support count is 1. Suppose that we set our threshold at s=2. Then among the example itemsets above, {Computer}, {Anti-virus SW} and {Computer, Anti-virus SW} are frequent itemsets.

B. Spatial Association Pattern

The spatial association mining discovers certain association relationships among a set of spatial attributes (such as geographic locations) and possibly some non-spatial attributes [7]. The main difference between the spatial association pattern and the traditional association pattern is that the spatial aspect of analysis data must be included in the pattern. Thus we define the spatial association pattern as the following.

Definition 1: A frequent spatial event/feature pattern is defined as a set of spatial predicates and non-spatial predicates, $\{P_1 \wedge \ldots \wedge P_m\}$ where at least one of the predicates is a spatial predicate, and the frequency of the pattern is greater than a given minimum threshold.

The spatial predicate presents the spatial relationship between objects. There are three categories of spatial relationships to consider: distance, directional, and topological [8]. Distance relationships are based on a distance metric between two objects. For example, close_to(x, y) is a spatial predicate based on the distance relationship. When the distance between two spatial objects x and y is less than a given distance threshold d, we say that x and y are close to one another, i.e. close_to(x, y) $\Leftrightarrow$ distance (x, y) $\leq d$. Directional relationships deal with the order of objects as they are located in space in relation to one another. Relationships such as left, right, north, and east are included in this category. Topological relationships characterize the types of intersection between two spatial objects. The intersection models by Egenhofer [5] provide 8 binary topological relations: crosses, contains, within, covers, covered_by, equals, disjoint, and overlaps. For example, {is_a(x, city) $\wedge$ within(x, BC) $\wedge$ adjacent_to(x, water) $\wedge$ close_to(x, US) } represents that cities within British Columbia (BC) and adjacent to water are close to U.S.A [7]. Here, x represents a feature (or an event) which is the focus on the analysis.

C. MapReduce

MapReduce [9], [10] is a programming model for processing large data sets with a parallel, distributed algorithm on a cluster. Computations in the MapReduce framework are described by map tasks and reduce tasks. Programmers address the required computation to be executed in parallel by designing the map function and $reduce$ function. Figure 2 shows the MapReduce model abstraction. In the first map phase, the key-$value$ pairs of each input block are processed by a mapper running independently on the storage node of the input block. The output of the map function is another set of intermediate key-$value$ pairs. The values associated with the same key across all nodes are grouped together and provided as input to the reduce function in the second phase. All values associated with the same key, key-$valuelist$, are located at a single reduce task. The reduce task then performs the operations specified in the reduce function and finally outputs the result to a file. Apach Hadoop [4] is one of the implementations of the MapReduce and an open-source software framework for storage and large-scale processing of data sets on clusters of commodity hardware, i.e., shared nothing architecture. In a Hadoop cluster, one node is designated as the $master$ and that main node schedules tasks for execution to the other worker nodes. Hadoop first splits data into physical blocks and distributes the blocks to the distributed file systems (such as HDFS [4]) automatically so that users do not need to worry about the locations and the distributions of data and thus have

to only focus on algorithms and programming. Furthermore Hadoop re-executes a crashed task without the re-executions of the other ongoing tasks and achieves good fault-tolerances.

III. Problem Statement and Related Work

A. Problem Statement

This study concerns the discovery of spatial association patterns based on the spatial proximity (i.e., distance) relationship of a focal event. The problem statement can be described as the followings.

Given:

1) An analysis focus event e and its instance object set S_e
2) A set of task-relevant features $F = \{f_1, \ldots, f_n\}$ and their instance objects S_f
3) A neighbor distance threshold d
4) A minimum frequent threshold min_freq

Objective:

To develop a parallel and distributed algorithm to discover spatial feature sets that are frequently observed in a nearby area around the focal event.

Constraints:

The algorithm follows the MapReduce programming model so that it is able to enable the distributed computation on a cluster of commodity servers.

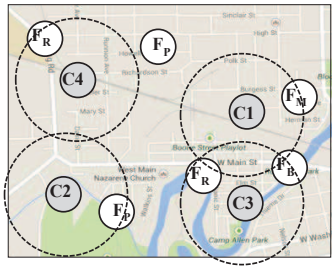
B. Related Work

Spatial data mining is the process of discovering interesting and previously unknown, but potentially useful patterns from large spatial data sets [8]. Many algorithms have been proposed for spatial data mining tasks, for example, spatial clustering [11], [12], spatial characterization [13], spatial association pattern mining [14], [7], [15], and spatial outlier detection [16]. Koperski et al. [7] first introduced the problem with mining association rules with spatial relationships (e.g., proximity, adjacency). Shekhar et al. [14] defined a clique-based neighbor relationship pattern that is called co-location. Morimoto [17] studies the same problem to find sets of (mobile) services located close to each other but proposes a different mining algorithm using spatial partitioning and non-overlapping counting schemes. Munro et al. [18] introduced complex relationship patterns of spatial data and addressed the pattern mining strategies from spatial databases. Zhang et al. [19] studied the techniques to find star-like and clique topological patterns. Yoo et al. [20], [21] presented the mining of variant co-location patterns like top-k closed co-locations and maximal co-locations. Vatsavai et al. [2] described the need of spatiotemporal data mining in the era of big spatial data. However, there are only a few works in the parallelization of spatial data mining. Xu [22] presented a parallel clustering algorithm for large spatial databases, and Kazar et al. [23] discussed the parallel formulation of the spatial auto-regression model for spatial classification.

IV. Proposed Methodology

Since our analysis focuses on finding the spatial association patterns of a specific focal event, the input spatial data may boil down to a set of neighborhood records (called *neighborhood transactions*) once all the neighbor relationships between the focal event and the task-relevant features have been extracted. The neighborhood transaction of a focal object is defined as a set of neighboring features of the focal object. Figure 3 shows the example of focal event-centric neighborhood and the neighbor list. To measure the strength of spatial association patterns of the focal event, the $support$ interest measure is used.

The proposed algorithmic framework uses three MapReduce



(a) Neighborhoods of a focal event

Focal event	Neighboring features
C1	F_R, F_B, F_M
C2	F_P
C3	F_R, F_B
C4	F_P, F_R

(b) Neighbor list

Fig. 3. Focal event-centric neighborhoods and the neighborhood transaction

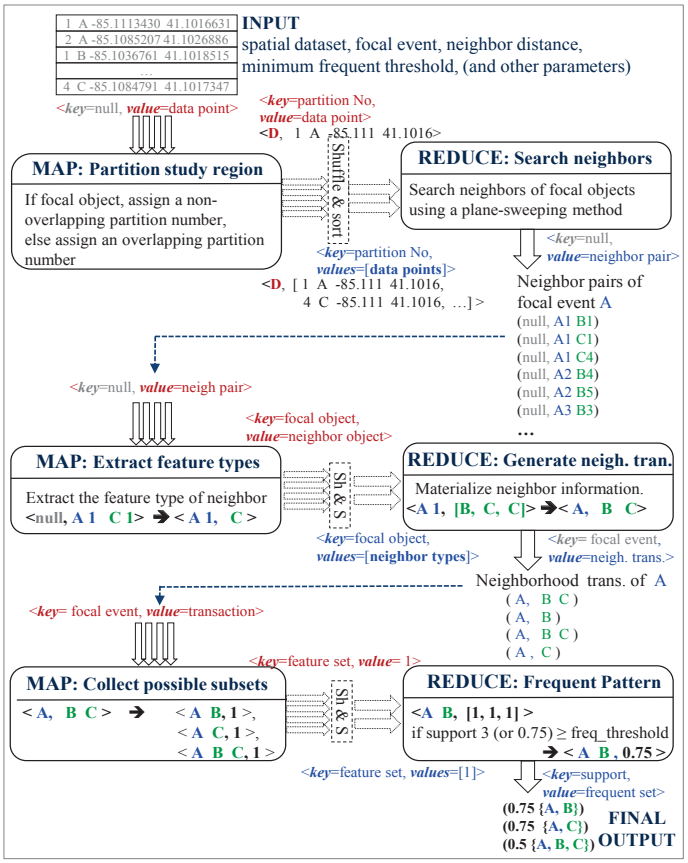
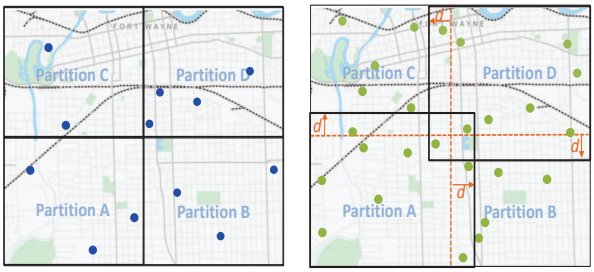


Fig. 4. An algorithmic framework for mining spatial association patterns of focal events on MapReduce



(a) Non-overlapping partition for focal event objects

(b) Overlapping partition for other objects

Fig. 5. Space partitioning for parallel neighbor search

job phases. The phase-1 searches all neighboring features of focal events, phase-2 generates neighborhood transactions from the output of phase-1, and phase-3 searches all frequent spatial feature sets from the neighborhood transactions. Figure 4 shows a running example of the proposed work.

Algorithm 1 Focal event's neighbor search (Phase-1)

Mapper (key, value=o)
```
1) if o is a focal event then ;
2)   partitionNo=byNonOverlapping(o);
3)   Emit(partitionNo, o);
4) else assignPartitionNoByOverlapping(o);
```

Reducer (key=$partitionNo$, value=$[o]$)
```
1) objectSet=sortBy_xCoordidate([o]);
2) neighVicinity=∅;
3) m=length(objectSet);
4) for i in 1 to m do
5)  candiNeighSet=subset(neighVicinity,
        objectSet[i], dist);
6)  foreach obj ∈ candiNeighSet do
7)   if(objectSet[i] is a focal event &&
        distance(objectSet[i],obj)≤dist) then
8)     oe=objectSet[i];
9)     of=obj;
10)  end if
11)  if(objectSet[i] is not a focal event &&
        distance(objectSet[i],obj)≤dist) then
12)    oe=obj;
13)    of=objectSet[i];
14)  end if
15)  Emit(oe, of);
16) end do
17) add(objectSet[i], neighborVicinity);
18)end do
```

Algorithm 2 Neighborhood transaction generation (Phase-2)

Mapper (key=o_e, value=o_f)
```
1) Emit(oe, of's type);
```

Reducer (key=o_e, value=$[feature\ type]$)
```
1) neighTrans=distinct [feature type]
2) Emit(oe, neighTrans);
```

Phase-1: The MapReduce framework splits the input data records into physical blocks without considering the geolocation of the data point. However, through the map function of the phase-1, the data records are rearranged for parallel neighbor search. Space partitioning is the process of dividing a space into non-overlapping or overlapping regions. We use two space partitioning strategies so that we would not lose any neighbor relations of focal event objects while also minimizing duplicate information. A non-overlapping partition strategy is used for the focal event. The space is divided into disjoint regions. For example, Figure 5 (a) shows four distinct regions using grid partitioning: A, B, C and D. The mapper function assigns one partition number to each focal object according to its geographic location. On the other hand, for task relevant feature objects, an overlapping partition strategy is used. As shown in Figure 5 (b), each partition region is overlapped with other regions with the neighbor distance size. Thus feature objects in the overlapping area are included in multiple partition regions, having one partition number per each region. The mapper outputs a key-value pair $\langle key' = partitionNo, value' = o\rangle$ where o is a data object. Since all values associated with the same key (i.e., partition number) are located at a single reducer by the MapReduce framework, the reduce can find all neighboring features of focal events in the partition without any missing. The reduce function uses a plane sweep algorithm [24] for the neighbor search. The plane sweep algorithm (or sweep line algorithm) is a type of algorithm that uses a conceptual sweep line to solve various problems in space. The idea of this algorithm is to imagine that a line is moved across the plane, stopping at some points. In our case, when the sweep line stops at a focal object, the search operation is restricted to feature objects in the neighbor vicinity of the sweep line. The reducer then outputs a key-value pair $\langle key' = o_e, value' = o_f\rangle$ where o_e is a focal event object and o_f is its neighboring feature object. Algorithm 1 shows the pseudo code of the phase-1 job.

Phase-2: The second phase job generates neighborhood transactions with the output of the first phase. The mapper is fed with the shards of neighbor pairs, $\langle key = o_e, value = o_f\rangle$ and outputs $\langle key = o_e, value = o_f$'s feature type $\rangle$ because we consider the neighbor's type, and not each neighbor object for the frequent pattern mining. The reducer is fed with $\langle key' = o_e, [value' = feature\ type]\rangle$. The reducer function constructs the neighborhood transaction of a focal object o_e, $neighTrans(o_e)$, with distinct neighbor feature types in the value list. Algorithm 2 describes the pseudo code of this second job.

Phase-3: The third job finds all frequent spatial feature patterns. When each mapper instance starts, the mapper is fed with the shards of neighborhood transactions, $\langle key = o_e, value = neighTrans(o_e)\rangle$. The map function collects all possible feature sets from $neighTrans(o_e)$ where each set includes the focal event, that is, $\{e, f_1, \ldots, f_{k-1}\}$. The proposed algorithm can limit the search of possible feature sets with specified pattern sizes, e.g., pattern size k=2 to 5. The map function outputs $\langle key' = feature\ set, value' = 1\rangle$

Algorithm 3 Frequent spatial feature set search (Phase-3)

Mapper (key=o_e, value=$neighTrans$)
```
1) candiSets=getPassibleSets(o_e, neighTrans);
2) foreach set ∈ candiSets do
3)   Emit(set, 1);
4) end do
```

Reducer (key=$feature\ set$, value=$[1]$)
```
1) support=sum([1]);
2) if support ≥ θ then
3)   Emit(feature set, support);
4) end if
```

pairs for each set. The MapReduce feeds the reducers with $\langle key' = feature\ set, [value' = 1]\rangle$, where $[value']$ is a list of occurrence of the key'. The reduce function sums the value list. When the computed value (i.e., support) is no less than the given frequent threshold, the reducer outputs $\langle key'' = feature\ set, value'' = support\rangle$ pair as the frequent association pattern. Algorithm 3 shows the pseudo code of the frequent set search.

V. Experimental Evaluation

We evaluated the performance of the proposed algorithm through the experiment.

A. Experiment Setting

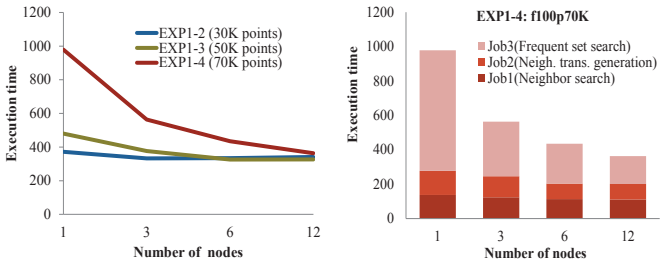
The algorithm was implemented in Java and MapReduce library functions. The performance evaluation was conducted on Amazon Web Services' (AWS) Elastic MapReduce platform [25], which provides resizable computing capacity in the cloud. For this experiment, we used instance type $m1.small$ in the AWS. The version of Hadoop used was 1.0.3.

The evaluation of the proposed algorithm was conducted under various experimental settings. Table I shows the detail of each experiment. Before each experiment, we counted the number of data points of a selected focal feature to know the number of possible neighborhood transactions. We estimated the region of each dataset with the minimum bounding rectangle that contains every data points in the dataset. The region was divided to four sub regions. The sub region size was different depending on the space partitioning method, i.e., the non-overlapping partition or overlapping partition.

We used real world data as well as synthetic data. The synthetic data was generated using a spatial data generator used in [15]. The focal event (feature) was randomly selected with considering the number of its data points. For the real data, crime incident records and Points Of Interest (POI) in the area of Fort Wayne, Indiana, were used. The purpose of this real data experiment was to discover interesting relationship patterns between crime incidents and nearby facilities (POI). The POI data was collected from two public data website, the GeoDeg website [26] and the ExpertGPS website [27]. The total number of data points was 765. The number of distinct types was 16. The POI types are hotel, school, bar, restaurant, church, shopping mall, office, and so on. The crime incident data was gathered from the 'Daily Activity Logs' of the Fort Wayne Indiana Police Department [28]. We collected 185,127 incident records from November 1, 2012 to October, 2013. The raw incident record was composed of six fields: incident no, date, time, nature, address and community. The 'nature' field represents incident type such as theft and burglary. After cleaning the nature value, we ended with 142 incident types. In this experimental evaluation, we chose 'theft' as the focal event. The number of theft incidents was 5,218. We preprocessed the theft incident records to get necessary property attribute values. The day of crime was derived from the incident 'date' data. The incident 'time' value was converted to a time zone, i.e. 0-3, 3-6, and so on. Since the raw 'address' field had only house number and street name, geocoding was used to obtain the geographic location of crime place. The complete address was obtained from the generated geographic coordinates using reverse geocoding, and then the zip code was extracted from the text address. The POI data and theft data were finally

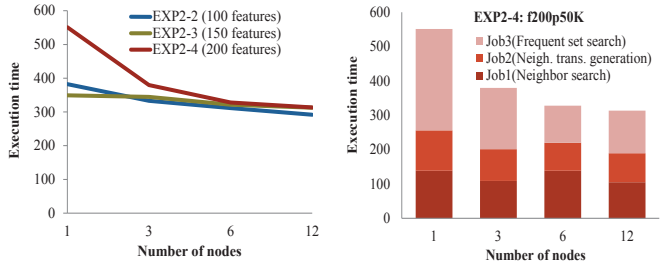
No	input dataset	# of feature types with a focal	# of data points	# of focal data points	neighbor distance threshold	frequent threshold
EXP1-2	f100p30K	100	30000	450	10	0.3
EXP1-3	f100p50K	100	50000	750	10	0.3
EXP1-4	f100p70K	100	70000	1050	10	0.3
EXP2-2	f100p50K	100	50000	420	10	0.3
EXP2-3	f150p50K	150	50000	415	10	0.3
EXP2-4	f200p50K	200	50000	399	10	0.3
EXP3-1	f50p50K	50	50000	420	10	0.3
EXP3-2	f50p50K	50	50000	420	15	0.3
EXP3-3	f50p50K	50	50000	420	20	0.3
EXP4-1	f50p50K	50	50000	420	10	0.3
EXP4-2	f50p50K	50	50000	830	10	0.3
EXP4-3	f50p50K	50	50000	1239	10	0.3
EXP4-4	f50p50K	50	50000	1595	10	0.3
EXP5-1-1 EXP5-1-2	FW_THEFT_POI FW_THEFT_property	17	5983	5218	1 Km	0.2

TABLE I
Experimental setting



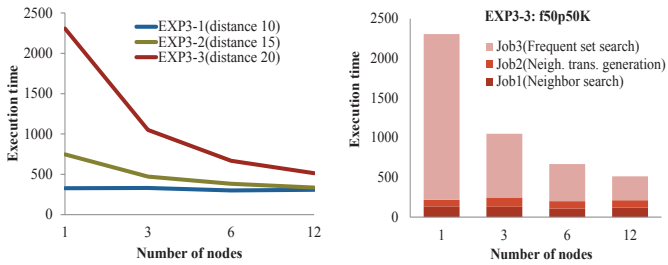
(a) (b)

Fig. 6. By number of data points: (a) Total execution time, (b) Execution time per job



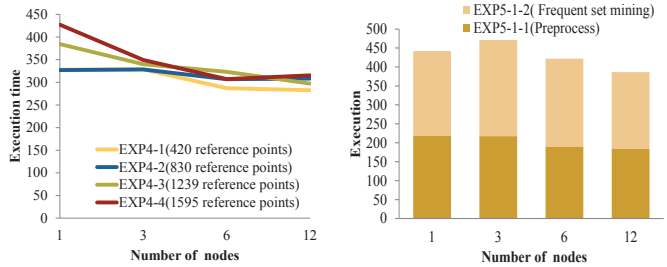
(a) (b)

Fig. 7. By number of feature types: (a) Total execution time, (b) Execution time per job



(a) (b)

Fig. 8. By neighbor distance: (a) Total execution time, (b) Execution time per job



(a) (b)

Fig. 9. Experiment result: (a) By number of focal points in synthetic data (b) With Real-world data

combined for one input dataset.

B. Experimental Result

Effect of number of data points: In the first set of experiments (EXP1-2, EXP1-3 and EXP1-4), we used three synthetic data sets which are different in the number of data points. Figure 6 (a) represents the result. With an increase of cluster nodes, the total execution time is decreased in all the cases. The EXP1-4 with larger number of data points shows higher execution time than others, but the execution time is effectively decreased with increase of nodes. The frequent set search of Job 3 (Phase-3) had the benefit as shown in Figure 6 (b). When the data size is not big enough for the parallel processing (EXP1-2 case), the benefit of increase of computing nodes was not significant.

Effect of number of features: In the second set of experiments (EXP2-2, EXP2-3 and EXP2-4), we used three synthetic data sets different in the number of distinct features. As shown in Figure 7 (a), the total execution time is decreased with increase of cluster nodes. The EXP2-4 with larger number of feature types shows higher execution time than others. In the EXP2-2 and EXP2-3 experimental settings, there was no significant difference in their execution time although the time of EXP2-2 shows slightly lower execution time. Figure 7 (b) shows the execution time of each job phase.

Effect of neighbor distance : In the next experiment, we used one data set (f50p50K) but used different neighbor distance thresholds. With increase of the neighbor distance, the execution time is dramatically increased in the single node cluster. However, with an increase of nodes, the execution time in the EXP3-3 (using neighbor distance 20) was significantly reduced as shown Figure 8 (a). Figure 8 (b) shows the time of frequent set search is mostly affected with increase of neighbor distance because larger neighborhood may include more neighbor objects. A majority time of frequent event set search (Job 3) was devoted to the enumeration of event sets from the neighborhood transaction. When the neighbor distance is relatively small, there was no much benefit with increase of cluster nodes (EXP3-1).

Effect of number of focal data points: In the last experiment with synthetic data (EXP4-1, EXP4-2, EXP4-3 and EXP4-4), we used four different sizes of focal data points. The total number of data points was the same with 50K. Figure 9 (a) shows the result. When the mining work was processed in parallel, the total execution time is decreased in all the cases. However, in this experimental setting with a small increase of focal data points, the performance effectiveness was not like linear with increase of nodes.

Result with real data: In this experiment, we used real-world data to evaluate the performance of the proposed algo-

rithm, and to find interesting association pattern results. The number of distinct feature types was 17 including the focal event 'theft'. The total number of data points was 5,983. The incident data points took 87% of them. We used 1 km for the neighbor distance. The support threshold was fixed to 0.2. This experiment had two runs in order to generate the neighborhood transaction (EXP5-1-1) and to find frequent event sets from the neighborhood transaction and additional property attributes of the theft incident (EXP5-1-2). Figure 9 (b) represents the result. Although the execution time is slightly increased in a cluster with three nodes, overall execution time was decreased with increase of nodes. We could find some simple patterns with high support: {theft, to_close(restaurant)} (0.63), {theft, to_close(school)} (0.64), {theft, to_close(bank)} (0.44), {theft, to_close(park), to_close(school)}(0.47), {theft, 12-15h} (0.25), {theft, 15-18h} (0.23), and so on. Any high frequent patterns related with specific streets or zipcode area are not found in this experimental setting.

VI. Conclusion

Most spatial association rule mining algorithms [7], [14], [17], [15] are Apriori-like algorithms. They use a multiple-pass generation-and-test framework, with a pruning phase to reduce the number of candidate sets before searching spatially associated objects. However, our algorithm differs from the traditional method in that it finds frequent feature sets without candidate generation. We have developed the parallel and distributed spatial association mining algorithm on Hadoop MapReduce framework. This proposed algorithm scans the entire input spatial data one time to find all the neighbors of focal event objects. Each worker then conducts the spatial association mining process with a shard of neighborhood records that includes all the necessary neighbor information. One MapReduce job is used for finding all the frequent association patterns. The experimental results show that our algorithmic design approach is parallelizable and follows a significant increase in speed, with respect to an increase in cluster nodes.

References

[1] *Big Data: Techniques and Technologies in Geoinformatics, Edited by Hassan A. Karimi*. CRC Press, 2014.

[2] R. R. Vatsavai, A. Ganguly, V. Chandola, A. Stefanidis, S. Klasky, and S. Shekhar, " Spatiotemporal Data Mining in the Era of Big Spatial Data: Algorithms and Applications," in *Proceedings of ACM SIGSPATIAL International Workshop on Analytics for Big Geospatial Data*, 2012, pp. 1–10.

[3] M. Zaki, "Parallel and Distributed Association Mining: A Survey," *Concurrency, IEEE*, vol. 7, no. 4, pp. 14–25, 1999.

[4] "Apache Hadoop," http://hadoop.apache.org/.

[5] M. Egenhofer, "A formal Definition of Binary Topological Relationships," *Foundations of Data Organi-zation and Algorithms*, pp. 457–472, 1989.

[6] R. Agarwal and R. Srikant, "Fast algorithms for Mining association rules in large databases," in *Proceedings of International Conference on Very Large Data Bases)*, 1994, pp. 487–499.

[7] K. Koperski and J. Han, "Discovery of Spatial Association Rules in Geographic Information Databases," in *Proceedings of International Symposium on Large Spatial Data bases*, 1995, pp. 47–66.

[8] S. Shekhar and S. Chawla, *Spatial Databases: A Tour*. Prentice Hall, 2003.

[9] J. Dean and S. Ghemawat, "MapReduce: Simplified Data Processing on Large Clusters," *Communications of the ACM*, vol. 51, no. 1, pp. 107–113, 2008.

[10] J. Dean and S. Ghemawat, "MapReduce: A Flexible Data Professing Tool," *Communications of the ACM*, vol. 53, no. 1, pp. 72–77, 2010.

[11] R. Ng and J. Han, " Efficient and Effective Clustering Methods for Spatial Data Mining," in *Proceedings of International Conference on Very Large Data Bases*, 1994, pp. 144–155.

[12] J. Sander, M. Easter, H. P. Kriedgel, and X. Xu, " Density-based Clustering in Spatial Databases," *Data Mining and Knowledge Discovery*, pp. 169–194, 1998.

[13] M. Easter, H. Kriegel, and J. Sander, "Knowledge Discovery in Spatial Databases," in *Proceedings of International Conference on Artificial Intelligence*, 1999, pp. 1–14.

[14] S. Shekhar and Y. Huang, "Co-location Rules Mining: A Summary of Results," in *Proceedings of International Symposium on Spatio and Temporal Database*, 2001, pp. 236–256.

[15] J. S. Yoo and S. Shekhar, "A Join-less Approach for Mining Spatial Co-location Patterns," *IEEE Transactions on Knowledge and Data Engineering*, vol. 18, no. 10, pp. 1323–1337, 2006.

[16] S. Shekhar, C. T. Lu, and P. Zhang, "A Unified Approach to Detecting Spatial Outliers," *GeoInformatica*, pp. 139–166, 2003.

[17] Y. Morimoto, "Mining Frequent Neighboring Class Sets in Spatial Databases," in *Proceedings ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2001, pp. 353–358.

[18] R. Munro, S. Chawla, and P. Sun, " Complex Spatial Relationships," in *Proceedings of IEEE International Conference on Data Mining*, 2003, pp. 227–234.

[19] X. Zhang, N. Mamoulis, D. Cheung, and Y. Shou, "Fast Mining of Spatial Collocations," in *Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2004, pp. 384–393.

[20] J. S. Yoo and M. Bow, "Mining Top-k Closed Co-location Patterns," in *Proceedings of IEEE International Conference on Spatial Data Mining and Geographical Knowledge Services*, 2011, pp. 100–105.

[21] J. S. Yoo and M. Bow, "Mining Maximal Co-located Event Sets," in *Proceedings of Pacific-Asia International Conference on Knowledge Discovery and Data Mining*, 2011, pp. 351–362.

[22] X. Xu, " A Fast Parallel Clustering Algorithm for Large Spatial Databases," *Data Mining and Knowledge Discovery*, pp. 263–290, 1999.

[23] B. M. Kazar, S. Shekhar, D. J. Lilja, and D. Boley, " A Parallel Formulation of the Spatial Autoregression Model for Mining Large geospatial datasets," in *Proceedings of SIAM International Conf. on Data Mining Workshop on High Performance and Distributed Mining*, 2004, pp. 58–73.

[24] M. I. Shamos, " Geometric intersection problems," in *Proceedings of Annual Symposium on Foundations of Computer Science*, 1976, pp. 208–215.

[25] "Amazon Elastic MapReduce (Amazon EMR)," http://aws.amazon.com/elasticmapreduce/.

[26] "Geodeg," http://geodeg.com.

[27] "Expertgps," http://www.expertgps.com/data/in/.

[28] "Fort Wayne Indiana Police Department," http://www.fwpd.org/.

Spectrum Recognition in Large-Scale Cognitive Radio Networks With Spectral Data Mining

Adam L. Anderson, C. Brett Witherspoon, and Behnaz Papari
Electrical and Computer Engineering Department, Tennessee Tech University, Cookeville, TN, USA

Abstract— *This work proposes a big data algorithmic examination of spectrum recognition in large cognitive radio networks (CRN) where the user spectrum demands and capabilities are orders of magnitude smaller than the available bandwidth. The proverbial big data "needle in a haystack" problem is particularly meaningful to this type of CRN where users are looking for a relatively tiny kilo- or megahertz channel in a many gigahertz spectrum. A large body of work focuses on sensing the spectrum in large-scale CRN; this proposed work instead focuses on recognizing the spectrum that is meaningful to the user without any knowledge of PHY-layer protocols. Data mining algorithmic approaches are considered for finding this spectral "needle" while a distributed heuristic solution is presented to show the power of distributed data in large-scale cognitive radio. The proposed method tags each user spectrum making it unique amongst all users and allows for clustered users to participate in making decisions on where to transmit data and how to receive it. Software-defined radios and computer simulation are used to demonstrate feasibility of the proposed approach to spectrum recognition.*

Keywords: cognitive radio, big data algorithms, spectrum sensing, large-scale networks, spectrum recognition

1. Introduction

Big data is a concept where massive datasets are analyzed and decisions made in a distributed manner that would otherwise be impossible with centralized computation and traditional data, including: aggregation, processing tools, and technologies. Furthermore, as the technologies used to analyze big data advance, more options will emerge for viewing the data from different angles [1], or to identify hidden patterns. In such analyses, data needs to be collected, processed and analyzed in an extremely large manner for a variety of applications [2]. However, data analysis as a main bottleneck impedes progress in many applications due to the lack of scalability, algorithms, and complexity of data during data acquisition [3]. During the coming decade, making sense of data in real time is more important not only for macro-scale big data sources like social media outlets and vendors but also for micro-scale big data such as telecommunications systems required for next-generation wireless.

This concept of big data on a micro-scale in telecommunications is especially important at the physical (PHY) layer where data rates and spectrum use are increasing dramatically. A common, oft repeated saying with regards to big data is that large systems and businesses generate petabytes of information each day [4], [5]. Likewise, software radios utilizing the entire radio spectrum are capable of generating peta*samples* per day and can certainly benefit from application of big data algorithms. For example, in cognitive radio networks (CRN), the spectrum is made available to more users especially when said spectrum lies underutilized and this spectrum can span many gigahertz of frequency. Cognitive radio secondary users (SU) are permitted to access spectrum that is not currently used by primary users (PU). The primary users transmit based on some coordinated transmission schemes, independent of the secondary users. Meanwhile, secondary users transmit when communication channels are idle from PUs [6], [7]. This approach to spectrum management can certainly be improved with decision making capabilities of big data algorithms.

With the dramatic developmental increases of wireless communications, and the almost insatiable ability for humans to consume data, up to many gigahertz of wireless bandwidth are part of the scarcity issue of limited wireless spectrum resource [8]. In order to mitigate the spectrum scarcity, there exists a necessity for a new communication pattern to exploit the existing wireless spectrum opportunistically without creating new, regulated radio-frequency (RF) spectrum bands [9]. Dynamic spectrum access (DSA) is a possible solution to eradicate the spectrum inefficiency problems. DSA techniques can select available communication channels through the use of free spectrum (spectrum holes) [10] which remain idle otherwise until used by the PUs. More specifically, spectrum holes can be exploited by SU that are able to independently detect the presence of PUs through continuous spectrum sensing [11], [12].

This spectrum sensing is a critical component of CRN and can be classified as either noncooperative or cooperative detection. Cooperative detection can be considered as a feasible solution to realize more accurate sensing and

This work was supported in part by grand prize winnings from the 2014 DARPA Spectrum Challenge.

improve the agility of sensing processes in wideband CRN. This approach to spectrum sensing is possible from decentralized fusion to cooperative compressive sensing (CS) to obtain global optimality and make a consensus decision for large-scale CRN [13]. In large-scale CRN, such as Wireless Regional Area Network (WRAN), the SUs may have different sensing SNRs due to environmental parameters; consequently, appropriate sensing modalities need methods based on cluster-based cooperative spectrum sensing (CSS) to transmit and fuse the sensing data efficiently [6]. By separating all the SUs into a few clusters and selecting the most favorable user in each cluster allows users to participate in making decisions on where to transmit data and how to receive it. Based on these features, high dimensional spectral data mining is essential in design of large-scale CRNs for low-cost spectrum sensing [14].

Since data mining has found success in CRN, specifically with the problem of spectrum sensing, this work suggests looking at spectrum recognition in a similar distributed manner. Spectrum recognition is an import aspect in CRN such as with regards to recognizing commercial standards [15] or unintended interference [16]. This work, instead, focuses on recognizing desired spectrum in a CRN where source and destination nodes have no method to coordinate on which band communication will take place inside a spectrum that is vastly larger than a single node is capable of sensing alone. Once a source node chooses a frequency to transmit on it is up to the destination cluster to use available data to correctly determine which spectral energy is intended for which user - spectrum recognition. This recognition is accomplished by "tagging" each spectrum in such a way that it can be uniquely identified. Software-defined radios (SDR) are used to confirm the feasibility of spectrum recognition in this manner while computer simulation demonstrates the network utilization increase when many nodes are able to process the spectral data jointly.

2. Network System Model

As the focus of this work is the concept of distributed algorithmic approaches to spectrum recognition in CRN a simplified network model is used to focus on certain key aspects. Consider the network modeled in Fig. 1. Source nodes are separated and grouped into clusters with a one-to-one mapping between a node in the source cluster wanting to transmit data to a node in the destination cluster. Non-data communication is only allowed within a cluster; thus, data communication can only take place if a source node can accurately choose an opening in the spectrum and the corresponding destination node can successfully determine both that a channel is being used *and* which channel corresponds to itself. To match hardware constraints in the experimental setup, the available bandwidth is roughly 4GHz while each user only has hardware capable of a 20MHz bandwidth though the center frequency of each user can span the entire band. This is a realistic assumption when node hardware are SDR enabled.

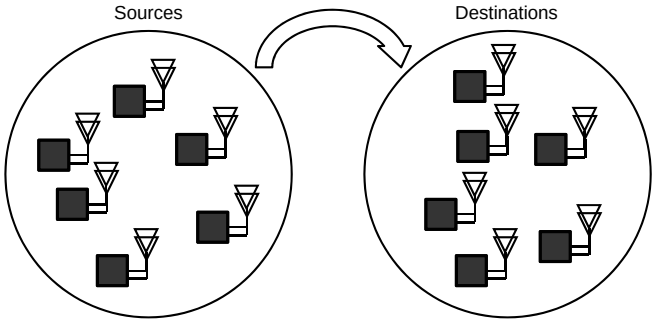


Fig. 1: A simple cognitive radio network where no spectrum is preallocated for any of the users. For simplicity, source and destination nodes are separated into clusters. Clusters can work distributedly to enhance network performance but no inter-cluster "handshaking" is allowed between source and destination nodes.

Spectrum recognition is considered on a frame-by-frame basis as depicted in Fig. 2. At the start of a frame, source nodes will sense the spectrum and try to find a spectral hole adequate for their transmission. Since the focus of this work is not on spectrum sensing itself, it is assumed that each node can always find an open channel once the entire band has been scanned. Since the band in question will be much larger than the capabilities of a particular node, cooperation within the cluster is assumed to get an accurate read on the spectrum. For example, each user has limited bandwidth capabilities due to hardware constraints. The total available spectrum is evenly distributed amongst users. If the distributed spectrum is still too large for a single user then that user must sweep its frequency range by changing its center frequency accordingly. In this manner, the entire spectrum can be sensed with the time required to scan the entire spectrum given as

$$\tau = \left\lceil \frac{B}{N_u W} \right\rceil T_o \tag{1}$$

where $\lceil \cdot \rceil$ is the ceiling operator, T_o is the settling time for the local oscillator on the SDR, B is the bandwidth of the entire spectrum (on the order of many gigahertz), N_u is the number of users participating in the cluster, and W is assumed the realistic bandwidth capabilities of each node (on the order of megahertz).

Once the entire spectrum, B, has been swept and spectral holes found, intra-cluster communication takes place and each user tunes its own center frequency to one of the openings. At this point, the source nodes could start transmitting data; however, the destination nodes do not know where to tune their own frequencies for reception. Thus in the network frame, some preamble or training time is inserted to allow the destination cluster to also sweep the spectrum.

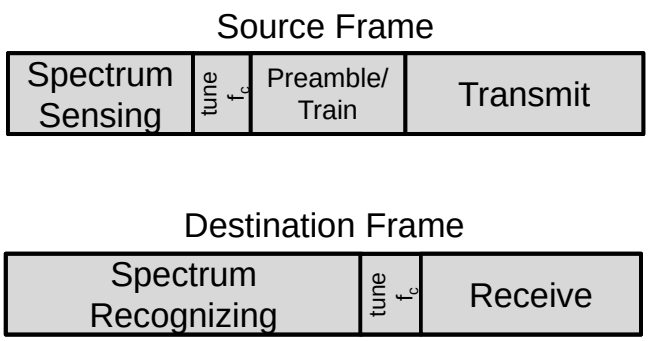


Fig. 2: A frame-based look at the spectrum sensing/recognizing cycle. Source nodes find spectral holes before transmitting data while destination nodes find spectrum intended for themselves before receiving. The duration (width) of each portion of the frame will depend on the number of users in the CRN.

Where the source cluster sensed the spectrum to find spectral holes, the destination cluster scans the spectrum to recognize spectrum that is intended for themselves. The method of individual spectrum recognition will be described shortly. Once the spectra have been recognized, those that recognize successfully will be able to receive the intended data once their own center frequencies have been tuned. This pattern of: spectrum sensing, transmit frequency tuning, spectrum recognition, receive frequency tuning is repeated every T seconds - the duration of a frame. It is assumed that a source user will change its transmit frequency every frame to keep with the behavior of cognitive radio with no preallocated coordination among SUs.

A key aspect of the particular CRN being examined in this work is that, in order to adequately recognize the spectrum, a node most be able to make a decision without actually demodulating the signal. This is an interesting situation since it ensures a wide variety of hardware can share the same spectrum with minimal changes. To this end, tagging must occur in the frequency domain, with the requirement that such tags: 1) Give the spectrum a unique signature against all other spectra and 2) Does not degrade the performance of individual links. Similar to code-division multiple access (CDMA), we choose to give each user's spectra a specific "chip pattern" to make it unique. The difficulty with coding in the frequency domain is that the magnitude spectrum is always positive thus no true orthogonality is possible; however, placing the pattern into the envelop of the spectrum can produce the desired results of spectrum recognition.

For example, Fig. 3 shows the spectrum tagging used. A user's spectrum (shown occupying the entire local band) is filtered by a spectrally biased pattern unique in the network. This results in a spectrum with peaks and valleys in the envelop of the magnitude spectrum similar to a time-domain direct-sequence spread spectrum (DSSS) given a DC offset.

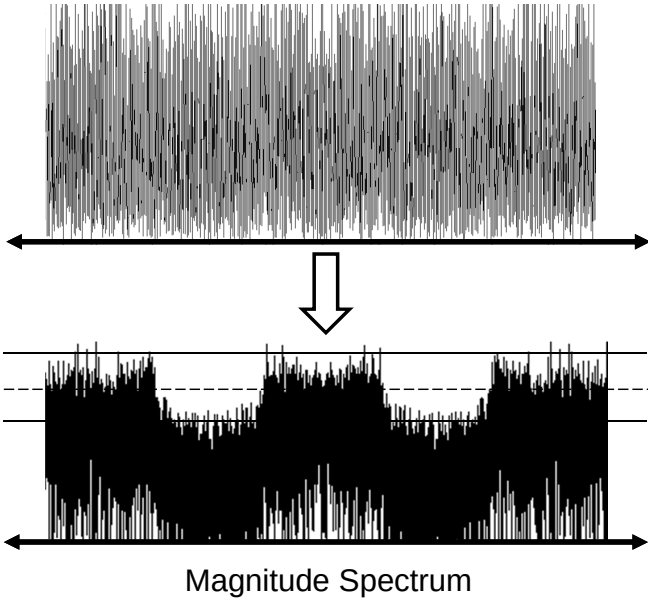


Fig. 3: Frequency tagging for spectrum recognition in CRN. Shown is the original user spectrum passed through a filter that crenelates the magnitude spectrum. Depth and frequency of the crenels gives each user's spectrum a unique signature.

The demarcation shown in Fig. 3 is to help show the analogy of DSSS used in the magnitude spectrum envelope. For ease in explanation, and since the resulting patterns look like a castle wall, the valleys that result from the spectrum tagging are referred to as "crenels". The number of, and depth of, these crenels will have impact on network and link performance. For example, deep crenels will make spectrum recognition easier on the network but may adversely affect the bit-rate performance of the individual links.

3. Preliminary Experimental Results

To demonstrate the spectrum recognition algorithm an experiment is performed using software-defined radio (SDR). The objective of the experiment is to transmit several signals, each tagged using spectral shaping, in non-overlapping frequency sub-bands and be able to successfully detect the user of each band. The demonstration attempts to emulate the spectrum recognition that would be performed by each distributed node of the large-scale network. The hardware used in the experiment is Ettus Research's N210 Universal Software Radio Peripheral (USRP) [17]. The radio includes dual 100 Msps analog-to-digital converters, dual 400 Msps digital-to-analog converters, and a Xilinx Spartan 3A-DSP FPGA. The RF chain is provided by Ettus Research's SBX daughterboard which operates in the 400 MHz to 4.4 GHz range with 40 MHz of bandwidth.

After digital down-conversion or up-conversion in the FPGA, the baseband samples are transmitted to and from

a host computer using a Gigabit Ethernet interface. The Gigabit Ethernet interface limits the sampling rate to 25 Msps using 16-bit samples or 50 Msps with 8-bit samples. For this experiment a moderate 5 Msps sampling rate is chosen and samples were transported using the 16-bit format. The N210 features a large FPGA fabric with extra space to implement low-latency signal processing functions and the capability to process 100 Msps. The focus of this work is algorithm development, so the experiment is implemented entirely in software on the host computer and transmitted through the wireless medium using the SDR interface.

3.1 Experimental Software Platform

To facilitate a quick development cycle the GNU Radio open-source software development toolkit is utilized. GNU Radio provides signal processing blocks for software defined radios which can be connected in a flowgraph creating a dataflow programming type architecture [18]. Blocks for most common signal processing tasks are included in the toolkit and implementing new signal processing blocks with the GNU Radio framework is made easy. Additionally, GNU Radio allows for custom signal processing blocks to be easily added to the flowgraph which allows for rapid prototyping of new wireless protocols and designs.

The toolkit also features a scheduler where each processing block is run in an individual thread for parralelism and tasks such as thread scheduling, buffer sizing, and message passing are handled transparently by the scheduler. For better performance, many of the processing blocks provided by GNU Radio take advantage of SIMD instructions of the host processor and custom blocks can also take advantage of SIMD instructions using the provided VOLK library. The GNU Radio framework allows *ad hoc* experiments and complex software defined radio systems to be implemented quickly and efficiently. All these features suggest SDR will be able to play an important role in big data algorithms used in CRN testbeds.

3.2 Experimental Setup

As a feasibility test and to emulate the spectrum recognizing performed by a single distributed node in the subset of a larger network, signals from four users with spectral shaping are transmitted from a USRP and received from another. These signals are each spectrally tagged, as explained previously, with a crenel depth of 50% chosen mostly to ensure a clean visualization of the spectrum tagging. Each user is then detected using the proposed algorithm.

At the transmitter, four random signals of 625 kHz bandwidth are generated. Each signal is then filtered with the coefficients of a spectral shaping filter $g_i(t)$ which is generated from a unique non-orthogonal spectral shaping code c_i of length five which is added with a spectral bias so that the code resides on the envelope of the magnitude spectrum. Each signal is then resampled to the target sampling rate of 5 Msps and frequency shifted into non-overlapping sub-bands. The signals are then summed before being transmitted. It is important to reiterate here that the purpose of spectrum tagging is not to achieve orthogonality. Frequency-division multiple access (FDMA) is the key orthogonality modality in CRN; the proposed spectrum tagging is simply to add uniqueness to each band so that it can be identified by simply looking at the spectral content. Demodulation/detection of individual symbols/bits is not necessary at the PHY layer to recognize spectrum.

The complex baseband representation of the transmitted signal then becomes

$$v(t) = \sum_{i=1}^{N} g_i(t) * u_i(t) e^{j2\pi f_i t} \tag{2}$$

where N is the number of user signals, $*$ is the convolution operator, $u_i(t)$ is the ith message signal and f_i is the frequency offset for the ith sub-band. For purposes of this experiment, $N = 4$ and the message signals are generated randomly with a Gaussian distribution so the spectra remain somewhat flat prior to tagging. The conglomerate signal, $v(t)$, is then upconverted to some high frequency for wireless transmission.

At the receiver, the N-point FFT of the received signal is computed. Assuming an AWGN channel, the frequency domain representation of the received baseband signal $R(f)$ is given by

$$R(f) = \sum_{i=1}^{N} G_i(f - f_i) U_i(f - f_i) + N(f) \tag{3}$$

where $G_i(f)$ and $U_i(f)$ are the Fourier transforms of the tagging filter and message signal, respectively, and $N(f)$ is spectral content of the AWGN.

Each user spectrum is then identified by cross-correlating the received spectrum with the spectral shape $G_i(f)$ determined by the unique shaping code c_i. The frequency bin containing the peak of the cross-correlation identifies the sub-band offset of the user. This detector is described by

$$\hat{f}_i = \arg\max_k \sum_{m=-\infty}^{\infty} R^*(m) G_i(m + k). \tag{4}$$

If the user can be recognized correctly the original signal $U_i(f)$ can be recovered by multiplying by the inverse function $G_i^{-1}(f)$ which recovers the original signal perfectly but with some noise amplification depending on the depths of the crenels. Thresholding can also be applied to prevent the detection of an inactive user.

For experimentation, the preceding signal is implemented on a software radio. The magnitude spectrum using the 4096-point FFT of the actual received signal is shown in Fig. 4. For easier visualization the result of the FFT is averaged with a single pole IIR filter. The destination node assigned to this particular slice of the spectrum correlates each band

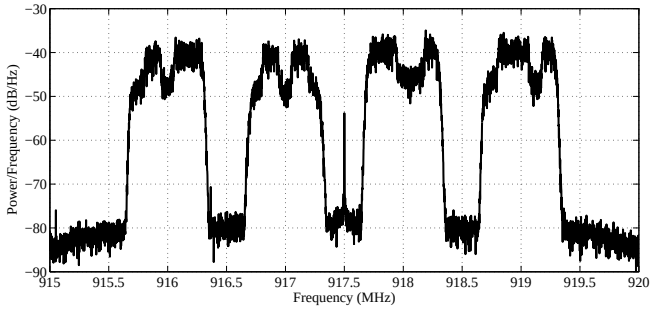


Fig. 4: The averaged 4096-point FFT of the spectrally shaped sub-bands. Envelop crenels correspond to a matching code; the spectrum on the left is associated with [0 1 0 1 1] for the length five code.

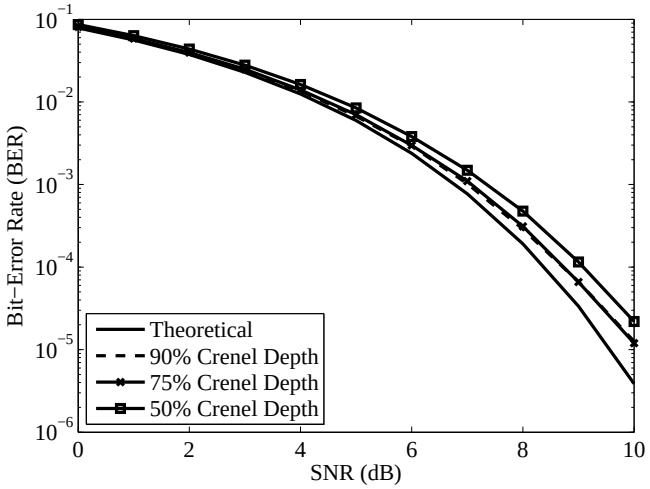


Fig. 5: Bit-error rate (BER) versus signal-to-noise ratio (SNR) for BPSK modulation. Shown is the theoretical BER and the BER at various crenel depths in the spectrum shaping.

against some known codebook of spectrum shapes. In this preliminary experiment the code used is easily recognized using the above detector without disrupting the underlying time-domain content.

4. Simulation Results

Given practical confirmation in the previous section, this current section attempts to demonstrate some of the behavior of spectrum recognition when bandwidth capabilities are far below available spectrum. Such a scenario would result in a massive amount of data that is not able to be processed through an individual node; thus, some form of mining technique is needed to draw conclusions about spectrum use, not limited to: center frequencies that users utilize to send information, traffic patterns of specific users, bandwidth and rate requirements and usages, and so. The following simulations will focus on performance degradation when spectrum tagging is used as well as recognition probability from these tagged spectra.

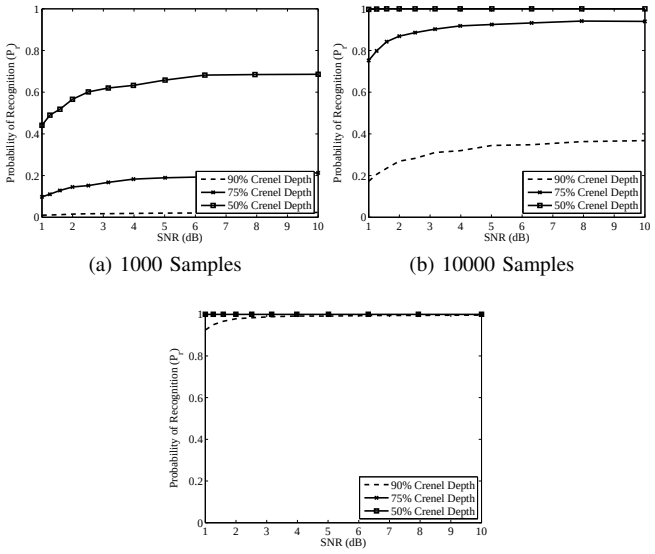
Consider tagging a single-user band with various levels of crenel depth in a channel that uses BPSK modulation. Bit-error rate (BER) performance at various depths is shown in Fig. 5. When the depth of the crenel is at 100% (no depth loss) then the peformance of the link should match theoretical values but no spectrum recognition would be possible. As the crenel depth increases, noise amplification occurs at the receiver with some loss in performance; however, this loss is marginal and does not appear to adversely affect the link in a significant way.

The next simulations demonstrate the ability for the proposed spectrum tagging to effectively distinguish the spectra such that recognition of individual user's bandwidths is possible. In Fig. 6(a)-(c) the spectrum recognition algorithm is performed when 1000, 10000, or 100000 samples are taken, respectively, in the FFT. There are two key behaviors revealed in these simulation results. First, the larger number of samples the greater the accuracy of spectrum recognition. This appeals to reason since more samples will average out the noisy channel and spectrum shapes will be more unique. Second, deeper crenel depths result in facility in spectrum recognition. This result is particularly interesting since this increase in recognition will result in some performance loss as we saw in Fig. 5 but increase network throughput. Thus, a CRN designer would need to carefully take into account the effects of shaping on overall network performance; a lower BER may not be detrimental if users can more often connect during appropriate frame times.

The final simulation considers network performance when the number of users is increased. Consider a metric termed "nominal utilization" which attempts to capture the percentage of time users in the CRN are utilizing the spectrum when they want to. The nominal utilization, μ, can be determined by looking at certain parameters of the CRN

$$\mu = \frac{(1 - P_r)(T - \tau)}{T} \tag{5}$$

where τ is the time required to scan the entire spectrum, T is the duration of a frame, P_r is the probability of recognizing the spectrum, which, as shown previously, is itself a function of the number of user and spectrum tagging filters. There are a couple behaviors captured by the nominal utilization. When T_o, the local oscillator settling time, is zero, and when spectrum recognition is perfect, then the CRN is assumed to be utilized at 100% - meaning all users can use the channel in every frame. However, this metric does not take into account throughput of the CRN since two networks with vastly different number of nodes can both have the same utilization but certainly a different throughput. Since this work is focused on spectrum recognition, throughput is not included in the network utilization metric. This sepration between utilization and throughput is important in the following results.



(a) 1000 Samples (b) 10000 Samples

(c) 100000 Samples

Fig. 6: Probability of spectrum recognition as the number of data samples and crenel depth increases. Performance curves are plotted against the single link SNR.

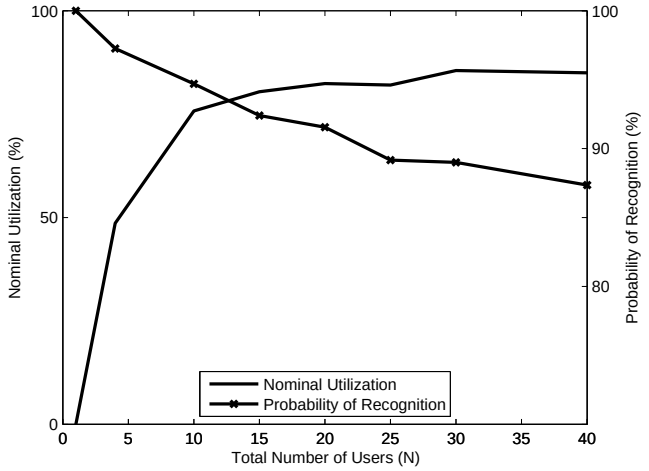


Fig. 7: Relationship between probability of recognition, nominal utilization and number of users. Though P_r may decrease as the number of users increases, utilization is increased since clusters can mine the entire spectrum more easily.

Fig. 7 shows the nominal utilization and recognition probabilities as the number of users in the CRN increases. As demonstrated previously, as the number of users increases the probability of spectrum recognition decreases since spectrum tags are more similar and are more difficult to distinguish. On the contrary, network utilization (and not just throughput) actually increases with the number of nodes. This is somewhat surprising since the probability of recognition drops and some links may not happen during each frame. The reason for this behavior is that more nodes allows for a higher distribution of user bandwidths over the entire spectrum. When the product of user bandwidth and number of users is equal to available spectrum then no frequency tuning is required to sense the entire spectrum and local oscillator settling times are not a problem. The trade-off is that more distribution results in more data aggregation and optimized spectral data mining techniques will be needed to fully exploit the potential of CRN using big data analytics.

5. Conclusion

Cognitive radio networks, especially when the spectrum is open and nodal bandwidth capabilities are far smaller than available spectrum, are a prime repository for big data mining techniques. Network traffic, per-user bandwidths, hardware/software availability all contribute to the CRN robustness and reliability. The amount of information required, however, is far beyond the capability of a single node to make meaningful decisions itself. This work has demonstrated a simple example of spectrum tagging to allow all nodes in a CRN to jointly possess the entire bandwidth by distributing the computational burden of spectrum sensing and recognition to a cluster of nodes with small bandwidths. With the capabilities of SDR, each node has a large center frequency range which allows spectrally mined information to be gathered and appropriate decisions made on spectral reuse.

References

[1] A. Bhatia and G. Vaswani, "Big data–a review," *IEEE International Journal of Engineering Sciences & Research Technology IJESRT*, Aug. 2013.

[2] R. C. Joseph and N. A. Johnson, "Big data and transformational government," *IEEE ITPro*, pp. 43–48, Nov. 2013.

[3] J. Manyika, M. Chui, B. Brown, J. Bughin, R. Dobbs, C. Roxburgh, and A. H. Byers, "Big data: The next frontier for innovation, competition," and productivity. Technical report, McKinsey Global Institute, Tech. Rep., June. 2011.

[4] A. McAfee, E. Brynjolfsson, *et al.*, "Big data: the management revolution," *Harvard business review*, vol. 90, no. 10, pp. 60–68, 2012.

[5] V. Mayer-Schönberger and K. Cukier, *Big data: A revolution that will transform how we live, work, and think*. Houghton Mifflin Harcourt, 2013.

[6] Y. Rao, W. Chen, and Z. Cao, "A sequential sensing data transmission and fusion approach for large scale cognitive radios," in *Communications (ICC), 2010 IEEE International Conference on*. IEEE, May. 2010, pp. 1–5.

[7] L. Fu, L. Qian, X. Tian, H. Tang, N. Liu, G. Zhang, and X. Wang, "Percolation degree of secondary users in cognitive networks," *Selected Areas in Communications, IEEE Journal on*, vol. 30, no. 10, pp. 1994–2005, Nov. 2012.

[8] X.-L. Huang, G. Wang, and F. Hu, "Multitask spectrum sensing in cognitive radio networks via spatiotemporal data mining," *Vehicular Technology, IEEE Transactions on*, vol. 62, no. 2, pp. 809–823, Feb. 2013.

[9] D. B. Rawat and G. Yan, "Spectrum sensing methods and dynamic spectrum sharing in cognitive radio networks: A survey," *International Journal of Research and Reviews in Wireless Sensor Networks*, vol. 1, no. 1, pp. 1–13, March. 2011.

[10] I. F. Akyildiz, W.-Y. Lee, M. C. Vuran, and S. Mohanty, "Next generation/dynamic spectrum access/cognitive radio wireless networks: a survey," *Computer Networks*, vol. 50, no. 13, pp. 2127–2159, Sep. 2006.
[11] S. S. S. T. Mehta, N. Kumar, "Performance improvement in spectrum sensing for cognitive radio networks with optimized weighted threshold principle," *Computer Networks*, Oct. 2013.
[12] D. Cabric, S. M. Mishra, and R. W. Brodersen, "Implementation issues in spectrum sensing for cognitive radios," in *Signals, systems and computers, 2004. Conference record of the thirty-eighth Asilomar conference on*, vol. 1. IEEE, Sep. 2004, pp. 772–776.
[13] Z. Fanzi, C. Li, and Z. Tian, "Distributed compressive spectrum sensing in cooperative multihop cognitive networks," *Selected Topics in Signal Processing, IEEE Journal of*, vol. 5, no. 1, pp. 37–48, March. 2011.
[14] L. Freitas, Y. Pires, J. Morais, J. Costa, and A. Klautau, "Data mining applied to cognitive radio systems," 2012.
[15] M. Lopez-Benitez, F. Casadevall, A. Umbert, J. Perez-Romero, R. Hachemani, J. Palicot, and C. Moy, "Spectral occupation measurements and blind standard recognition sensor for cognitive radio networks," in *Cognitive Radio Oriented Wireless Networks and Communications, 2009. CROWNCOM '09. 4th International Conference on*, 2009.
[16] L. Kong, Z. Xu, J. Wang, and K. Pan, "A novel algorithm for jamming recognition in wireless communication," in *Image and Signal Processing (CISP), 2013 6th International Congress on*, 2013.
[17] E. R. Website, accessed April 2014. [Online]. Available: http://home.ettus.com
[18] G. R. Website, accessed April 2014. [Online]. Available: http://www.gnuradio.org

SESSION

SECURITY AND PRIVACY IN THE ERA OF BIG DATA AND RISK ANALYSIS

Chair(s)

TBA

Some risk measures and their applications in financial data analysis

Q. Tang, L. Zhang
Dept. of Mathematics, University of Sussex, Brighton BN1 9QH, UK

Abstract *- The quantitative definitions of risks are becoming ever more important in the big data era when we try to decode messages behind numbers exhibited by social, natural, and financial phenomena. In this report, we study some "risk functions" that we derived while investigating issues related to hedge fund performance and data mining of financial options' implied volatility. This leads us to further statistical investigation of real world implied volatility for S&P500 related asset to derive some very interesting formulas and observations.*

Keywords: *Risk measure, financial crisis, turning point, distribution pattern, negative risk.*

1 Introduction

Using random walk theory to derive valuation formula for financial options (with possibly some added features) is a popular pursuit.

In recent years, many risk measures have been introduced. The first category is mainly based on random walk thory. Typical example of this type is the standard deviation of asset returns, coming out of Black-Scholes option value equation. The second category is very intuitive, for example, measures like drawdown and winning runs. The third category is mostly based on statistical descriptions of loss distribution, for example VaR, Omega etc.. The fourth category is combinations of some of the above, falling more or less into the utility function concept, a typical example is Sharpe ratio.

These risk measures are widely used in various contexts in assessing the impact of recent financial market volatilities and in helping to shape regulations. A typical example is the papers assembled in [4], where formidable effort has been put into reconsidering the impact of the 2008-2009 financial crisis on funds of hedge funds using various risk benchmarks and management criteria.

The category that draws us into data investigation in this report is the first category, the risk measures that are derived from random walk theory. The initial motivation is that in the derivation process, various authors make different assumptions, that ultimately leads to the Black-Scholes formula. Very little questions were asked about if such assumptions should be made, if they were not made, what are the consequences.

We aim to follow this approach by relaxing/adjusting some of these assumptions in deriving risk measures. We are aware that by adjusting additional assumptions in some cases, we may no longer follow the strict random walk theory. The derived formulas will be of non-standard feature from classical theory. However, our point is to think provacatively about these formulas, an also, to look at if applied to real world financial data, what would these derived risk measures imply.

The first finding is the so-called S risk measure σ^2 / μ, it used assumption that asset prices will not decline (break the normal distribution pattern of returns). It is interesting to find that this measure has, if applied to hedge fund returns data, the ability to detect the arrival of financial crisis on three occasions in 1998, 2001 and 2008 in advance (cf. [7]).

The second finding is on the so-called price movement exhaustion phenomenon, we applied the same process as binomial derivation of Black-Scholes formula, but without imposing any conditions on the dynamic probability of asset price moving up or down, we derived a new risk measure $\sigma^2 - \omega\sigma$, which can catch local maximum of asset prices for the assets at the period of tests chosen (the chosen time period is entirely arbitrary) (cf. [5]).

In the end, following observations from [1]. [2], [3], [5] [6] and [7], we look at the whole set of financial options for S&P 500 and get some interesting statistical observations from the data between 2005-2013 regarding 'nagative' risks. (current work by the authors).

Thanks to the arrival of the big data era and the increasing tools available for data exploration, we have the opportunity to observe quantitatively many phenomena that could alter our views about many traditional theories on risk measures.

2 S-risk measure and prediction of financial crisis

When investigating hedge funds, one of the terms came to attention is absolute return. It is well-known that the selling point of the hedge fund industry is to produce absolute returns. Hence assuming that asset prices cannot go down seems to be an interesting proposition.

In binomial tree setting, this looks like

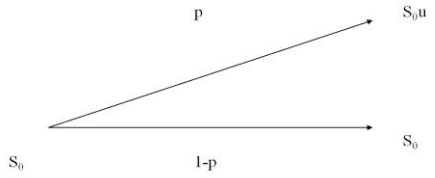


Figure 2.1

It does not follow the normally distributed returns pattern, but let's just formally go down the route of B-S derivation (follow [7]):

$$E(S_{\Delta t}) = pS_0u + (1-p)S_0$$

$$Var(S_{\Delta t}) = S_0^2 p(1-p)(u-1)^2$$

Let μ be the expected unit time period return, σ be the one time period risk of the asset. We get

$$E\left(\frac{S_{\Delta t}-S_0}{S_0}\right) \approx \mu\Delta t$$

$$Var\left(\frac{S_{\Delta t}-S_0}{S_0}\right) \approx \sigma^2\Delta t$$

And further

$$E(S_{\Delta t}) \approx S_0(1+\mu\Delta t)$$

$$Var(S_{\Delta t}) \approx S_0^2\sigma^2\Delta t$$

Follow standard BS derivation approach:

$$pS_0u + (1-p)S_0 \approx S_0(1+\mu\Delta t)$$

$$S_0^2 p(1-p)(u-1)^2 \approx S_0^2\sigma^2\Delta t$$

Comparing them, we get

$$u = 1+\frac{\sigma^2}{\mu}+\mu\Delta t$$

$$\mu\Delta t = p(u-1)$$

$$\sigma^2\Delta t = p(1-p)(u-1)^2$$

$$\frac{\sigma^2}{\mu} = (1-p)(u-1)$$

Add assumption that $\sigma^2 << \mu$(since variance is of order retrun^2, this seems reasonable) we obtain

$$(u-1)^2 = (\frac{\sigma^2}{\mu}+\mu\Delta t)^2 << u-1 = \frac{\sigma^2}{\mu}+\mu\Delta t$$

Follow standard derivation of Black-Scholes formula, we arrive at

$$\frac{\partial P}{\partial S}(S_0,t_0)S_0r + \frac{1}{2}\frac{\partial^2 P}{\partial S^2}(S_0,t_0)rS_0^2\frac{\sigma^2}{\mu} - P(S_0,t_0) + \frac{\partial P}{\partial t}(S_0,t_0) = 0 \quad (2.1)$$

So we have the S-risk measure $\frac{\sigma^2}{\mu}$ standing in the place of σ^2 in the Black-Scholes equation.

We take a pool of 60 hedge funds (chosen by a fund of hedge funds on the grounds that these are quality funds with good reputation and return history) and use Monte-Carlo method to generate random nonnegative portfolio weights ($w_1;w_2;$ … $;w_{60}$) twenty thousand times such that w_1 + … + w_{60} = 1. Using 1 year historic data to calculate and plot the scatter of

$$\left(\frac{\sigma^2}{\mu}, \mu\right)$$

we obtain the following:

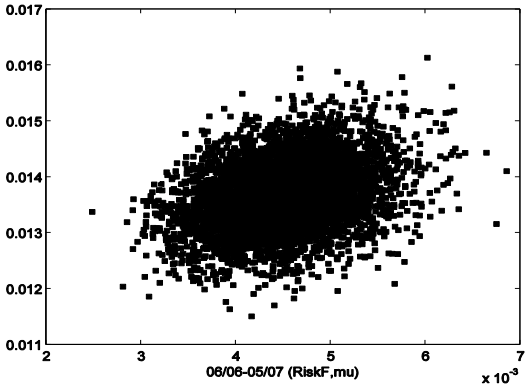


Figure 2.2

in May 2007. By Jan 2008, the picture becomes (we omitted many pictures in the middle, but it is a gradual transition from Figure 2.2 above)

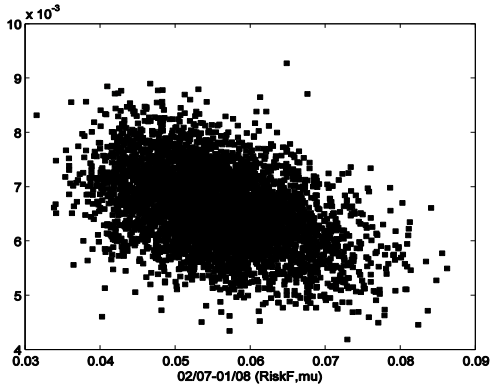


Figure 2.3

It is clear from Figure 2.2 that for higher risks, we get statistically higher returns. For Figure 2.3, that for high risks, we get statistically lower returns. Beware that Jan 2008 is shortly before the time when dramatic plunge in stock market took place. When the real stock market plunge have taken place, due to so many negative returns generated by hedge funds, the picture now looks like

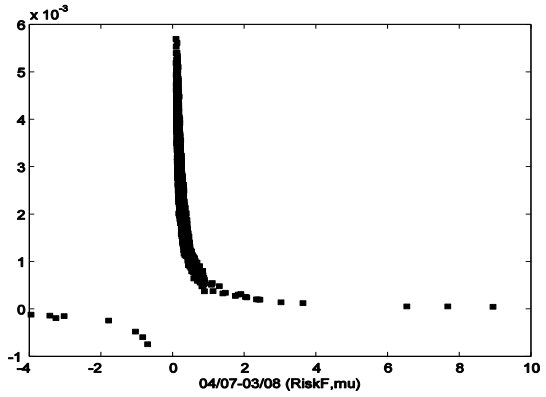


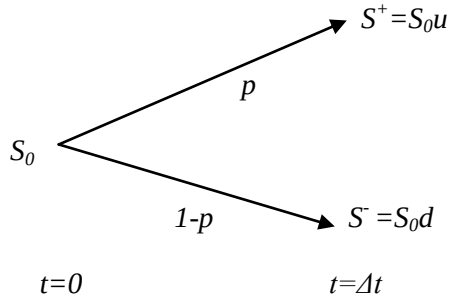
Figure 2.4

We also checked that during the 1998 and 2001 financial crisis, the same behavior happens 2-3 months before the big stock market crash. In these cases, we have chosen an arbitrarily large number of hedge funds' returns data. But as hedge funds die in large quantities after each financial crisis, we did not have the same sets of hedge funds for verification, so we did not include the results here.

This implies that hedge fund managers, as a collective set of people, exhibit some (maybe unconscientious) stressed behavior before the arrival of large scale stock market falls.

3 Price movement exhaustion

We again look at the binomial tree in option price derivation process:



If we simply use *p* to denote the probability of price going up, we have, without any assumptions with respect to *u*, *d* and *p*

$$u = 1 + \mu\Delta t + \frac{\sigma\sqrt{(1-p)\Delta t}}{\sqrt{p}} = 1 + \mu\Delta t + \sigma\eta\sqrt{\Delta t}$$
$$d = 1 + \mu\Delta t - \sigma\sqrt{\Delta t}/\eta \qquad (3.1)$$
$$\eta = \sqrt{1-p}/\sqrt{p}$$

It is worth mentioning that what we have written above is in the most general setting. In the literature, there is Cox-Ross-Rubinstein formula assuming (cf. [2])

$$p = \frac{e^{r\Delta t} - d}{u - d}, \quad u = e^{\sigma\sqrt{\Delta t}}, \quad d = e^{-\sigma\sqrt{\Delta t}} \qquad (3.2)$$

or even simpler by Higham (cf. [3]), assuming that

$$p = 0.5$$

to derive the Black Scholes option formula, resulting in σ, the standard deviation, representing the risk.

We follow the standard derivation process of Black-Scholes formula without pre-assuming any forms of *u*, *d* and *p*, we obtain a different formula (cf. [5]):

$$rC = C_s S_0 r + C_t + \frac{1}{2} C_{ss} S_0^2 (\sigma^2 - \omega\sigma) \qquad (3.3)$$
$$\omega = -r\sqrt{\Delta t}(\eta - 1/\eta)$$

where

$$\eta = \sqrt{1-p}/\sqrt{p}, \qquad (3.4)$$

Remark: This is almost Black-Scholes with only the term $\sigma^2 - \omega\sigma$ replacing σ^2. Note that if *p = 1/2,* we have $\omega = 0$, this goes back to the Black-Scholes equation.

The interesting point is that asymptotically if *p* is a quantity that changes as asset price changes, than if $p \to 1$, we might have the quantity

$$\sigma^2 - \omega\sigma < 0$$

resulting in negative risk. The option valuation equation then appears to become a mixed type partial differential equation (the original Black-Scholes equation is a backward parabolic equation).

This issue of negative risk (cf. [1]) or negative variance (cf. [6]) have been observed in real world data by various authors, the general attitude has been to avoid them in the mathematical modelling.

Here we (follow [5]) attempt to give a first guess on what could be the reason for negative risk to arise:

The challenge now is if the "new" risk function $\sigma^2 - \omega\sigma$ could turn negative sometimes. We first make some reasonable choices: Δt *=1/255,* $\sqrt{\Delta t}$ = 0.0626, and let

$$Day\,\mathrm{Re}turn_+ = \begin{cases} DayReturn & if\ DayReturn > 0 \\ 0 & if\ DayReturn \le 0 \end{cases}$$

$$p = \frac{\sum_{DaysInInvestigation} DayReturn_+}{\sum_{DaysInInvestigation} |DayReturn|} \qquad (3.5)$$

Then it is easy to see that by suitably adjusting "DaysInInvestigation", we could make $\sigma^2 - \omega\sigma$ negative. More importantly, they have some price momentum implications:

Example: We take all FTSE 100 shares between the dates of 1st July 2008 and 1st July 2009 and look for occasions where "risk function" turns negative for no less than 2 days (if risk function turns negative for just one day, it could just be noise). Interest rate is chosen to be constant:

a) If we change interest rate in the calculation of *p*:

Interest rate	No. of days in investigation	Number of occasions where "risk function" turns negative for no less than 2 days	Number of local maximums captured
0.03	5	146	146
0.05	5	152	152
0.08	5	175	175

b) If we change the number of days used in calculating *p:*

Interest rate	No. of days in investigation	Number of occasions where "risk function" turns negative for no less than 2 days	Number of local maximums captured
0.05	3	717	717
0.05	5	152	152
0.05	8	7	7

4 Statistical observations of the BS implied volatility

First, a word about the choice between SPX and SPY:

Period	SPX Volume	SPY Volume	Volume Ratio SPY/SPX
Pre-Crisis(2005-Bear Stein)	332,599,157	279,524,781	0.84
During Crisis (Bear Stein – end of 2009)	233,942,487	547,829,314	2.34
After Crisis (2010 – 2013)	527,382,841	1,705,122,316	3.23
Overall	1,093,924,485	2,532,476,411	2.32

From the table above, it is clear that the volume in SPY has picked up dramatically after the financial crisis. Hence we decide to take SPY related European options as our main subject of study.

We make the following additional choices:

a) Use last price to calculate implied volatility. Special cases:
   a.1) If last = 0, ask < asset price, last is replaced by (bid+ask)/2
   a.2) If last =0, ask > asset price happened over the history of the option, the whole option is deleted (we deleted 43 call and 24 put options in a database containing more than 80,000 options)
b) Use USD 1 year LIBOR rate as interest rate.

The price data we use in this article come from www.deltaneutral.com, whom according the corporate website, is the data provider for Wall Street Journal.

All other data, for example, SPY end of day share price, SPY dividend, USD I year LIBOR end of day rates are all downloaded from Thompson Reuters Data Service.

The time period chosen is Jan 2005 - Dec 2013, we aim to include the leading to, during and aftermath recovery of the 2008-2009 financial crisis.

We start with a typical volatility surface (date chosen arbitrarily on June/17/2011) and we have (a coloured contour film of this is available on www.dataaction.co.uk)

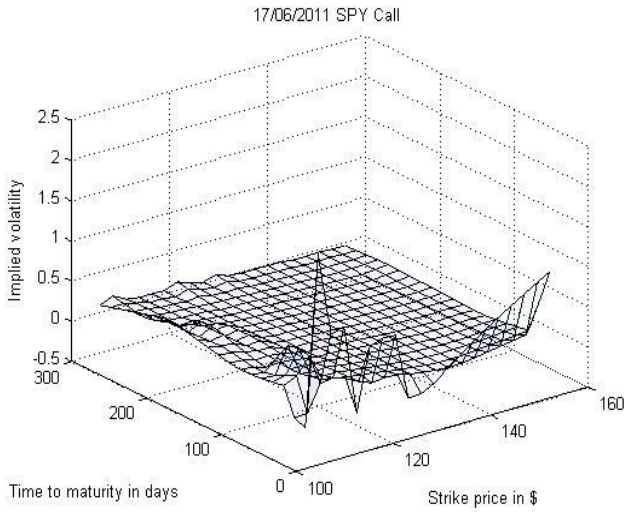


Figure 4.1

Conclusion: we can see that there is clearly a complicated structure of volatility surface across the strikes and the time to maturity. Some local sneer and smile across strikes is visible

but the time structure does not look close to a solution of a PDE of (reverse) parabolic type.

4.1 Correlation between implied volatility and returns

The intuitive relationship between implied volatility and underlying asset returns should be

a) Negative for call options. The real world data demonstrates that

Call	Total frequency % for correlation < 0		Total frequency % for correlation > 0	
	Delay 0	Delay -1	Delay 0	Delay -1
2-months 5-day	92.99%	20.51%	7.01%	79.49%
2-months 10-day	99.29%	19.35%	0.71%	80.65%
3-months 5-day	94.19%	18.60%	5.81%	81.40%
3-months 10-day	99.29%	18.28%	0.71%	81.72%
4-months 5-day	94.19%	17.80%	5.81%	82.20%
4-months 10-day	99.51%	17.35%	0.49%	82.65%

b) Positive for put options. The real world data demonstrates that

Put	Total frequency % for correlation < 0		Total frequency % for correlation > 0	
	Delay 0	Delay -1	Delay 0	Delay -1
2-month 5-day	15.14%	75.72%	84.86%	24.28%
2-month 10-day	7.25%	72.38%	92.75%	27.62%
3-month 5-day	13.49%	76.43%	86.51%	23.57%
3-month 10-day	6.14%	73.31%	93.86%	26.69%
4-month 5-day	13.23%	76.83%	86.77%	23.17%
4-month 10-day	5.96%	73.89%	94.04%	26.11%

Remark 4.1.1: Here X-month Y-day means we use all options with time-to-maturity up to X months, to calculate every day's algebraic average risk, and take Y days of risk to correlate with Y days of returns of the underlying asset.

Remark 4.1.2: The case delay = 0 means risk and underlying returns are calculated simultaneously. The case delay = -1 means risk is taken one day earlier than underlying return. So delay = -1 means to investigate predictive ability of risk on underlying returns..

Conclusion: the real world average risk behaves as we expected in relation to returns of the underlying asset.

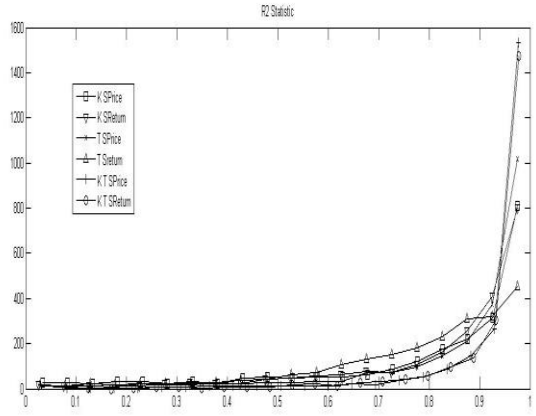
4.2 Variance capture

Although correlation confirms the relationship between risk and underlying return, it is clear that, if we carry out regression analysis between risk and return, the R2 statistic is very poor. Hence correlation may not reflect the true relationship between risk and return.

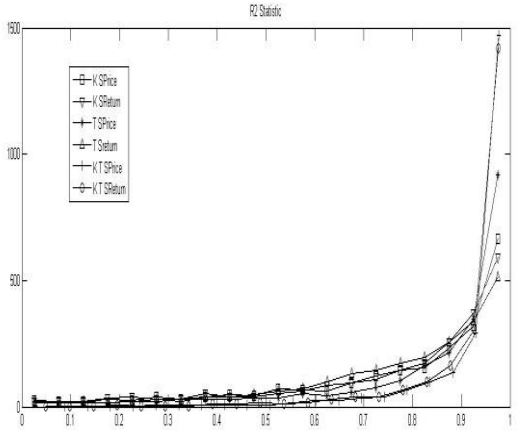
In order to overcome this, we carry out regression of the simple average risk against various combinations of

underlying price (return), time to maturity and strike.

We obtain very good R2 by using 5-day historical data rolling statistics. In particular, for call options:



for put options:



Remark 4.2.1: Here the horizontal axis represents the R2 value of regression, the vertical axis represents the frequency that such R2 has been achieved.

Remark 4.2.2: As to notations such as

K SPrice

we mean regression of risk against

K – strike price and SPrice – underlying share price.

In addition, T represents time to maturity, SReturn represents underlying share return.

In particular, the regression of risk against K, T and SPrice and against K, T and SReturn all produced better R2 statistic in comparison to other combinations. This means that the variance of risk can almost be perfectly taken up by the variances of underlying share prices (or returns), time to maturity and strike combined. The following tables are the probabilities of the corresponding situations by differentiating the signs of betas:

a) For call options

K+,T+,SPrice+	0.026	K+,T+,SReturn+	0.008
K+,T+,SPrice-	0.017	K+,T+,SReturn-	0.035
K+,T-,SPrice+	0.025	K+,T-,SReturn+	0.008
K+,T-,SPrice-	0.017	K+,T-,SReturn-	0.034
K-,T+,SPrice+	0.001	K-,T+,SReturn+	0.023
K-,T+,SPrice-	0.462	K-,T+,SReturn-	0.440
K-,T-,SPrice+	0.008	K-,T-,SReturn+	0.015
K-,T-,SPrice-	0.441	K-,T-,SReturn-	0.434

Table 4.1

b) For put options

K+,T+,SPrice+	0.291	K+,T+,SReturn+	0.287
K+,T+,SPrice-	0.001	K+,T+,SReturn-	0.005
K+,T-,SPrice+	0.55	K+,T-,SReturn+	0.491
K+,T-,SPrice-	0.016	K+,T-,SReturn-	0.076
K-,T+,SPrice+	0.027	K-,T+,SReturn+	0.045
K-,T+,SPrice-	0.027	K-,T+,SReturn-	0.009
K-,T-,SPrice+	0.022	K-,T-,SReturn+	0.057
K-,T-,SPrice-	0.064	K-,T-,SReturn-	0.029

Table 4.2

Remark 4.2.3: The notation in the third row, first column, for example,

K+,T-,SPrice+

means that the regression coefficient is positive regarding K (strike), negative regarding T (time to maturity), and positive regarding SPrice (share price). In the third column, SReturn stands for the underlying share return. The numbers in the 2nd and 4th columns are the probability of the corresponding situations.

Remark 4.2.4: The highlighted areas in Tables 4.1 and 4.2 are the dominating phenomena in terms of probability, as it can be seen easily that they take up more than 70% of the possibilities.

Conclusion: The call option risk is negatively related to underlying price or returns. The put option risk is positively related to underlying price or returns just as we expected.

4.3 Are negative risks really related to local maximums and local minimums?

Following the work of Section 3, we use real world data to justify the relationship between negative risk and local extreme values of the underlying asset price.

First, let us investigate the mechanism of negative risk formation from Black-Scholes formula. In fact, zero risk value of an option is defined mathematically to be, for call option, where S stands for asset price, K the strike, r the risk free interest rate, T the time to maturity,

$$\max(S-K*\exp(-r*(T-t)),0)$$

This formula gives only non-zero values when the call option strike is below or very near asset price, hence only in-the-money options can really enjoy this. In the case of in-the-money options,

$$\max(S-K*\exp(-r*(T-t)),0) == S-K*\exp(-r*(T-t)).$$

A similar argument run for put options, only in-the-money options can really enjoy having negative risk.

Intuitively, options priced below zero risk value should be endowed with "negative risk". While this has been observed in various literature when exploring real world data, people have avoided going further into the details.

We have run a number of numerical tests on the database and we observe that the assumption that the options being priced below zero risk value for being "cheap" is not obvious, ie, such phenomenon does not lead to straightforward arbitrage opportunities. However, options priced below zero risk value do have some interesting properties linked to the discussions in Section 3. That is, they are closely linked to price movement momentum.

In the following table, we track every single option with strikes within 20% of the underlying asset price on the market. We observe when the value of the option falls below the zero risk value between Jan 2005 and Dec 2013, we obtain:

Call			
Isolated negative days	max	Continuous negative intervals	Max
9507	5572	12367	10198

Table 4.3

Put			
Isolated negative days	Min	Continuous negative intervals	Min
9403	5371	10865	8397

Table 4.4

Conclusion: statistically, when an option whose value falls below the zero risk value on the financial market, for a call option , at that time, a local maximum value of the underlying asset will appear with high probability during that period (Table 4.3). For a put option, a local minimum value of the underlying asset will appear with high probability during that period (Table 4.4).

5 Conclusions

The study of implied volatility is of great interest in helping people to really understand the world of financial derivatives. In this report, we do not limit ourselves to the formulas. We use the formidable tools of data analysis developed in recent years to look at what has happened in real life using traded historical data. It is very interesting to discover some obscure corners of financial options valuation and attach significant links to underlying asset price movements to them.

We successfully confirmed that the real world data does conform to many of the well-established mathematical theories or standard financial concepts. But real world data also brings surprises. For example, when price of options fall below zero risk value, the underlying asset price exhibit local extreme values which may induce further research and investigation.

All these conclusion relies, one way or another, on the details embedded in the derivation of random walk model of Black-Scholes financial option valuation formula. They imply that we should not be satisfied with imposing many assumptions, the random walk phenomenon has a lot more to it embedded in the real world data and worth studying in a much more detailed way.

6 References

[1] Bernard Dumas, Jeff Fleming, and Robert E. Whaley, Implied Volatility Functions: Empirical Tests, The Journal of Finance, Vol. LIII, NO.6 Dec, 1998.

[2] John C. Cox, Stephen A. Ross and Mark Rubinstein, Option pricing: a simplified approach, Journal of Financial Economics, 1979,9(7): 229-263.

[3] Desmond J. Higham, *An Introduction to Financial Option Valuation*, Cambridge University Press, 2004

[4] Greg N. Gregoriou, Reconsidering Funds of Hedge Funds, the financial crisis and best practices in UCITS, tail risk, performance and due diligence, ELSEVIER, 2013.

[5] Qi Tang, Danni YAN, Autoregressive trending risk function and exhaustion in random asset price movement. Journal of Time Series Analysis. 2010, 31 465-470.

[6] Jim Gatheral, *The Volatility Surface*, John Wiley & Sons, Inc. 2006.

[7] Qi Tang, Haidar Haidar, Bernard Minsky and Rishi Thapar, The S-risk function in option valuation and in predicting performance of hedge funds, Journal 34: Cass-Capco Institute Paper Series on Risk, https://capco.com/sites/all/files/restricted/journal34-article-17.pdf

The Use of Fully Homomorphic Encryption in Data Mining with Privacy Preserving

A. Laécio A. Costa[1], **B. Ruy J. G. B. de Queiroz**[2]
[1]College of Informatics, Instituto Federal do Sertão Pernambucano - IFSertão, Petrolina, Pernambuco, Brazil
[2]Center of Informatics, Universidade Federal do Pernambuco - UFPE, Recife, Pernambuco, Brazil

Abstract - *Data mining is a computational tool widely used today which aims to extract useful information from various databases. Nowadays with the large volume of information that is produced, stored in a remote database (using cloud computing), concerns about confidentiality and privacy of information are arising due to lack of guaranteed security by storage service and the mining algorithm. A new approach of modern cryptography, defined as the Fully Homomorphic Encryption (FHE), allows for the encrypted data to be arbitrarily computed which is a solution that aims to preserve the security, confidentiality and data privacy.*
This article aims to present a study of the literature to identify research that proposes methods that ensure the confidentiality and privacy in the mining of databases based on fully homomorphic encryption.

Keywords: Data Mining, Privacy Preserving, Fully Homomorphic Encryption

1 Introduction

Many companies need to explore sensitive data derived from data mining of multiple databases in order to obtain useful information. To preserve the integrity and privacy in this exploration, they need to make the ciphertext without disclosing or having knowledge of the content of the data operations. For example, two competing companies need to exchange sensitive information, this information is important for assessing a given market scenario, so each company "X" and "Y" encodes and forwards its information to a platform of homomorphic processing. The processing platform performs operations with the encrypted data of the two companies without knowledge of the original message and returns the result to the two companies "X" and "Y" for analysis. At no time did either company have access to the data of the competitor, but managed to make a more accurate analysis of the market using the output data. Therefore the proposed homomorphic encryption aims to ensure this scenario is safe as well as practical.

The homomorphic encryption is an area of modern cryptography, enabling the completion of computation of arbitrary computations on a ciphertext and still achieves the encrypted result that corresponds to the sequence of operations performed in the original text. For example, one could add two encrypted numbers and then another person could decipher the result, without being able to find the initial value of the computed numbers.

According to Valeria Nikolaenko and Dan Boneh [18], the current mining algorithms data should have access to data in clear text for proper handling. Thus preserving privacy and confidentiality in the current mining process is a vulnerability that needs to be resolved. Thus, several studies are being directed towards the development of schemes and techniques that allow the manipulation and computation of encrypted data, with the prospect that the homomorphic encryption can solve this problem effectively and efficiently [16].

One solution to this problem was first defined in 1978 where Rivest, Adleman and Dertouzos [21] suggested the construction of secret homomorphisms - privacy homomorphisms - as a way of providing a technique that meets this demand. However, the technique presented in their proposal [21] called for unfavorable conditions that would make the practice technique a fully homomorphic cryptosystem.

The homomorphic encryption has two forms, partially homomorphic encryption and fully homomorphic encryption. The partially homomorphic encryption is defined when it has limits on the amount of transactions with encrypted [23] data. Nevertheless, the fully homomorphic encryption is a cryptographic system that allows you to make a set of arbitrary mathematical operations (without limitation) in the ciphertext, which should result in another ciphertext corresponding to the result of the operation in plain text. In this research we focus on fully homomorphic encryption in order to meet the requirements of mining encrypted data.

Fully homomorphic encryption with any circuit can be evaluated homomorficly, allowing the construction of programs that can run with the encodings of its inputs to produce an encryption of its output. Programs such as homomorphic never decode their inputs, they can be used by untrusted third parties, therefore it is impossible to reveal its entry data and internal processes.

The existence of a fully homomorphic and efficient cryptographic system would have great practical implications for the outsourcing of private computing, as in the context of cloud computing. The example of a cloud-based model is shown in Figure 1, which illustrates three different hospitals which provide sensitive data to the cloud. The cloud computing

platform analyzes and extracts useful information (data mining operation) from data entry and delivers it to recipients.

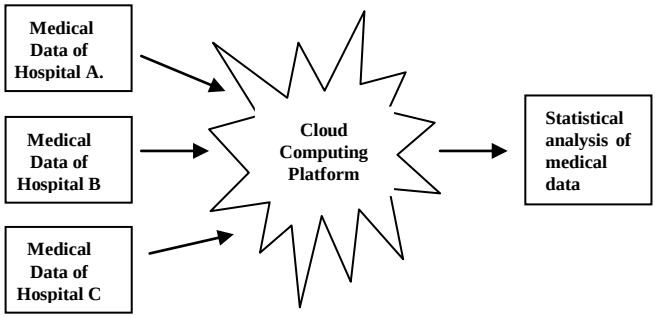


Figure 1 : FHE model based in Cloud Computing

In order to preserve the privacy and confidentiality of patient information and using fully homomorphic encryption, the Platform for Cloud Computing will perform operations using only encoded data and deliver the results to the recipients. Thus, no information can be leaked during calculation or delivery stages of the communication.

In this context, we intend to present a study to identify methods and proposals that use completely homomorphic encryption to maintain the privacy and confidentiality of information during the process of data mining.

This paper is organized as follows: section two presents the fundamentals and concepts related to the fully homomorphic encryption and Privacy in Data Mining. Section three presents the methodology applied in the study, section four the results are presented in a survey and analysis of studies cataloged, and in section five final remarks are made in conclusion.

2 Preliminary

2.1 Fully Homommorphic Encryption

The concept of homomorphic encryption was initially defined by Ronald Rivest, Len Adleman, and Michael Dertouzos [21], after verifying that the RSA cryptographic system developed by Ronald Rivest, Shamir Adl, Len Adleman [22], had multiplicative homomorphism. Thus, Rivest, Adleman and Dertouzos, defined special encryption functions called "privacy homomorphisms", which are a subset of arbitrary encryption schemes.

The homomorphic public key encryption is a cryptographic system that allows the performance of a set of operations on the data when they are encoded, resulting in its data appearing in plain text. The public key homomorphic encryption allows it to perform various calculations on the data without revealing any information of the encoded message, preserving the privacy and confidentiality of data.

The fully homomorphic cryptographic models remained as speculation for more than 30 years, because researchers have failed to develop a fully homomorphic encryption method that could compute arbitrary numbers of operations. But in 2009, the researcher Craig Gentry [12] proposed a system which has a valid encryption scheme with a fully homomorphic public key that can arbitrarily compute on encrypted data, based on ideal lattices.

Ideal lattices is an area that studies mathematical structures, and has applications in cryptography because of the complexity involved in solving difficult problems. It is defined as a set of points in n-dimensional space with a periodic structure. In other words, a lattice vector is a discretized space, using the concept of standard dimension orthogonal linear transformation, among others [20].

As defined by Gentry, the homomorphic encryption schemes completely preserve the operations of addition and multiplication on encrypted blocks, ie:

Definition: Given that *E(m)* is the application of the encryption algorithm to a message m, a cryptographic scheme is fully homomorphic if:

$$E(m_1 + m_2) = E(m_1) + E(m_2),$$

$$E(m \bullet m_2) = E(m_1) \bullet E(m_2), \text{ where}$$

- For any m1 and m2 block of the message to be encrypted; and
- The same applies to any number of consecutive operations performed on a single block.

The research of Craig Gentry [12] is based on polynomial ideals for obtaining a scheme of restricted homomorphic encryption restricted (SHE - Somewhat homomorphic encryption). This scheme is able to add and multiply encrypted texts in a homomorphic way, but as transactions are conducted noise is added to the ciphertext. According to Gentry the decoding algorithm works provided that such noise does not exceed a certain threshold.

Using the concept that is called bootstrapping, Craig Gentry proposes the construction of a new scheme that can decode and reduce noise homomorficly. However, this adaptation leads directly to increasing the size of the parameters, making it impossible to implement the scheme. From the Gentry scheme other schemes have been proposed in an attempt to be more practical and efficient, an example based on the Learning with Errors (LWE) proposed by Zvika Brakerski and Vinod Vaikuntanathan [4].

2.2 Privacy in Data Mining

Data mining helps in extracting useful knowledge from large data sets, but the process of data collection and data dissemination may, however, result in an inherent risk of threats to confidentiality and data privacy. Some personal information about individuals, companies and organizations must be deleted before it is shared or published, unless such information is encoded. Thus, preserving privacy in data mining has become a very important issue in the last decade.

The problem of learning something without revealing the data itself has not been defined recently. This problem was proposed in 1982 by Andrew C. Yao [24] in his article "Protocols for Secure Computations", where Yao defines a model that can ensure privacy between the parties. To [24], a

protocol between two parties is considered safe to perform operations if participants do not learn anything beyond what is revealed by the output of the circuit.

In this context, the term originated in Privacy Preserving Data Mining (PPDM - Privacy-Preserving Data Mining) that refers to the area of data mining for protecting sensitive information from unsolicited disclosure. Mining techniques over traditional data analyze and model the data set statistically, while the preservation of privacy is primarily concerned with the protection against disclosure of individual data records. This separation of domain points to the technical feasibility of PPDM.

The term Privacy-Preserving in Data Mining - PPDM was introduced by [11] and [17]. These researchers considered two fundamental problems in PPDM: i) preservation of privacy in data collection, and ii) privacy during the mining process of a partitioned data set from several private companies.

The Agrawal and Srikant [11] researchers developed an algorithm of randomness that allows a large number of users to contribute with their private records for centralized data mining, limiting disclosure of their records; Lindell and Pinkas [17] designed a cryptographic protocol for the construction of a decision on a data set horizontally partitioned between two parties tree. These methods were later refined and extended by many researchers.

The goal of preserving privacy in data mining (PPDM) is to extract relevant knowledge from large amounts of data, while protecting sensitive information [9].

Thus, the process to preserve the privacy and confidentiality of data during the data mining requires new technologies and advancements, especially in the area of modern cryptography.

3 Applied Method

Conducting a survey of relevant studies, using the mechanisms of systematic mapping, this mechanism establishes a formal investigation in the literature, in order to give credibility to the ongoing research in the area.

The steps of the systematic literature mapping process were defined in order to explore and find the most relevant primary studies able to respond to the problem: "It is computationally feasible to explore sensitive data in data mining when they are encrypted using fully homomorphic encryption in order to ensure security, privacy and confidentiality efficiently?".

With the implementation of the systematic mapping protocol, studies dealing with Fully Homomorphic Encryption, Data Mining, privacy and confidentiality of data were identified.

Thus, with the implementation of the search strings in the mills: ACM Digital Library, Engineering Village, IEEE Xplore, Springer Link, Scopus and ScienceDirect, a total of 213 trials, distributed according to Figure 2 were collected.

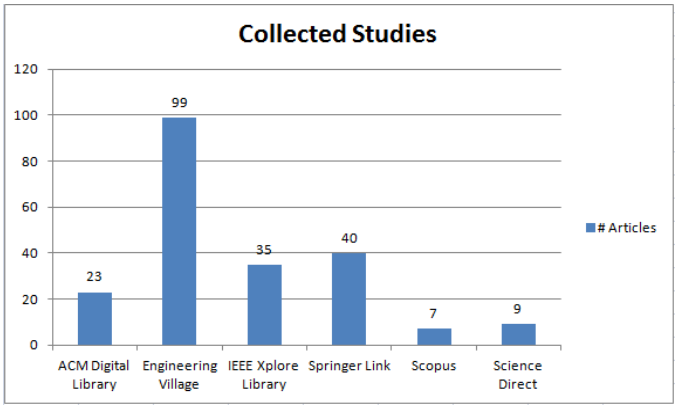


Figure 2 : Articles listed in the databases queries. Data generated in January of 2014.

Five rounds for analysis of the criteria for inclusion and exclusion of the collected studies were performed. In each round, the studies were analyzed in order to filter the most relevant articles within the research theme. After performing the first 4 rounds the studies listed came to the total of 25 possible studies for primary selection, as Table I. Thus, preserving privacy in data mining has become a very important issue in the last decade.

TABLE I. DESCRIPTION THE STEPS IN REVIEW

	STEPS	DESCRIPTION
Primary Selection	1st Round	Checking whether the study has full text available on the web
	2nd Round	Check if the written language is English
	3rd Round	Check for duplicates
	4th Round	Analysis Title, Abstract and Keywords.
Sscundary Selection	5th Round	Analysis of introduction, method and conclusion, verifying that addresses new Fully homomorphic encryption schemes applied to Encrypted Data Mining with method and evaluation.

With the results of the primary selection complete, then the secondary selection began with the implementation of the fifth round, which aims to identify studies that show a method or protocol used in mining that aims to preserve the privacy and confidentiality of data in addition to its fully homomorphic encryption base. However, the evaluation of the proposed method was considered in addition to their results. Thus, Table II presents the data of secondary selection with the final results of six cataloged studies after the implementation of the five rounds.

TABLE II. RESULT OF SELECTION AND STUDIES INCLUDED IN REVIEW

Base	Primary Selection	Secundary Selection: Proposed Method of FHE applied in Data mining	Studies Included
IEEE Xplore	9	3	3
ACM Digital	6	0	0
Springer Link	4	1	1
Scopus	0	0	0
Elsevier ScienceDirect	5	2	2
Enginering	1	0	0
Total	25	6	**6**

Aiming to collect a maximum amount of studies, the period of publication was limited. But analyzing the studies listed, it was realized that studies relevant to the issue only occurred after the year 2011 and that there are few studies cataloged when specifying the application of data mining as a research theme. This demonstrates and confirms that few authors highlight studies of fully homomorphic encryption applied in data mining, since there is only recent evidence that fully homomorphic encryption is possible.

With the aim of expanding the coverage of the survey and to reassure the researcher, manual searches were performed. In the manual search, two articles pertinent to the purpose of this research study were identified, totaling eight relevant studies.

4 Analysis of Studies

The studies listed address schemes using fully homomorphic encryption that can be directly applied in data mining. Each study provides specific features, where the authors propose a fully homomorphic encryption – based on a solution that is feasible, efficient and ensures the privacy, confidentiality and integrity of mined data.

The studies cover different steps of data mining, such as process: the exploitation of data to statistical analysis using linear regression; protocol PIR (Private Information Retrieval) extended to a protocol PBR (Private Block Retrieval); association data rules; search on encrypted data; operations with distributed data set; recovery protocol SPIR (Symmetrically Private Information Retrieval); SQL queries on encrypted data and homomorphic signatures.

Conducting a survey of relevant studies, using the mechanisms of systematic mapping, this mechanism establishes a formal investigation in the literature, in order to give credibility to the ongoing research in the area.

4.1 Statistical analysis using the Linear Regression

Linear regression is a statistical analysis tool widely used in data mining because it allows you to verify the linear relationship between the dependent and independent variables in a dataset. But, in the current approaches, it is necessary to have data in plain text so that we can perform statistical operations. The FHE is a solution to perform statistical analyzes of the encoded data while preserving confidentiality and privacy.

Cataloged in this study "Privacy Preserving linear regression modeling of distributed databases", the authors Weiwei Fang et al. [10], address the tradeoff between privacy and statistical analysis with a focus on linear regression applied in the field of data mining. The question of this study is defined based on the following survey: how to identify a feasible process to maintain the linear relationship between the dependent and independent variables, without disclosing your personal data in the data mining process.

Thus, the PPRCP (Privacy Preserving Regression Coefficient Protocol) protocol, which allows one to perform a linear regression using the FHE was presented. In evaluating implementation of this protocol, no further information was leaked, so the authors argue that the protocol is computationally indistinguishable. The protocol securely computes the regression coefficient without the leakage of sensitive data.

The security of the proposed protocol is based on the cryptographic scheme, if the fully homomorphic encryption is secure, then the protocol is also secure.

4.2 Recovery blocks of data bits with the protocol PIR

The PIR (Private Information Retrieval) is a family of two-party protocols in which one party has a database, and the other part wants to consult it with some restrictions and guarantees of privacy. This approach was introduced by Chor, Goldreich, Kushilevitz and Sudan [7], and since then has attracted attention of researchers.

The PIR protocol allows a client to retrieve a certain element of your choice in a database without the owner of the database being able to determine which element was selected. To ensure the confidentiality admittedly is a trivial solution, sending the entire database to the client allowing it to see with perfect privacy, but there is a problem regarding the computational cost, because it is detrimental to communication in the occurrence of large databases. Another problem in this proposal is that the user can obtain additional information.

Collected in the study "Single-Database Private Information Retrieval from Fully homomorphic encryption", was the presentation of the protocol for private information retrieval that allows a user to retrieve the *i* bit of a database of n bits that are shown, without disclosing to the administrator of the database the value of index *i*, at a lower cost of communication. Furthermore, propose an extension of the PIR protocol for a recovery protocol of private blocks PBR (Private Block Retrieval) being more efficient, both protocols (PIR and PBR) are based on FHE.

Comparing the proposal with existing protocols PIR and PBR, it is concluded that the presented method is conceptually simpler. Overall, the PBR protocol is practical and more efficient than the existing PBR protocols in terms of total execution time when a high-speed network is available.

In the safety analysis presented, it was demonstrated that the protocol is semantically secure if the fully homomorphic encryption scheme is semantically secure.

4.3 Association Rules of Data with Privacy Preservation

According to Brusso [6], "association rules are descriptive patterns that represent the probability that a set of items will appear in a transaction where another set is present". An algorithm is widely applied in mining due to the possibility of finding patterns.

Typically, association rules represent patterns in existing stored transactions. For example, from a database that enters their items purchased by customers, a strategy for mining using association rules could generate the following rule: {belt, purse} → {shoes}, which indicates that a customer purchasing a belt and a bag, with a certain degree of certainty, will also buy shoes.

The authors, Mohamend Kaosar, Russell Paulet and Yi You [15] propose a technique ARM (Association Rule Mining) of secure comparision (making checks of interesting correlations in a set of database) based on homomorphic encryption scheme completely promoting greater efficiency due to the reuse of resources.

In order to evaluate the presented method, a prototype software was implemented to test the feasibility of this approach by using a fully homomorphic encryption scheme open source library called Smart-Vercauteren for cryptographic operations. With this library, the authors could measure the time required for the method of comparison of integers.

The authors state that the main contribution of the present study is the use of fully homomorphic encryption to solve the problem of association rules while preserving privacy. The protocol was safe, based on hardness assumption of the cryptographic system. The technique proposed privacy preservation can be used in the collaborative filtering of data between two databases.

4.4 System Search on Encrypted Data Base

The search data is an important tool for mining, where the storing of date in encrypted form is required it will necessary to find a solution that allows for the processing of such data. Thus, the authors present in the study "The Implemention and Application Fully Homomorphic Encryption Scheme" [14], a solution that combines the Attributes Based Encryption (ABE - Attribute Based Encryption) and Fully Homomorphic Encryption that can perform computations with encrypted data. In the proposed solution, the encoded data is computed by servers in the cloud in order to preserve the privacy of the system´s input and output of the circuit, but the FHE schemes based on lattices and ideals based on LWE (Learning With Errors) do not offer solutions that are truly practical for the method presented.

ABE is a collection of cryptographic tools based on attributes and policies assigned to users by an authority. In particular, it allows attaching attributes and policies to the message to be encrypted so that only a receiver that is assigned policies/attributes can decrypt it. The attributes are Boolean variables with arbitrary labels and political calculations are represented as Boolean circuits with the attribute variables (which evaluate to true or false) [3].

The method presented allows one to make searches on encrypted data stored in the cloud, ensuring privacy and confidentiality of information in storage, in the survey conducted, algorithms that run in this process and in transmission between the server and the requesting user. In the proposed system, all the data computed by the cloud servers are encrypted in order to preserve the privacy and confidentiality of data.

But according to reviews, FHE is not yet a practical and efficient scheme that allows its use in making systems as presented. Efforts must be made to improve the efficiency of cryptographic schemes in order to make practical applications that require security, confidentiality and privacy that can provide FHE.

4.5 Set Operations in Distributed Data

The mining algorithms need to unite the various databases so that they can perform operations on these sets. Since the amount of data submitted to the applications of data mining has grown considerably as an indirect result of reductions in the cost of collection, transmission and storage of data there is an increasing amount of sensitive data stored on untrusted remote servers which raises concern for confidentiality.

During the mining process steps are required to perform operations on data sets, such as union and intersection, to identify relationships. These operations are widely used in the mining process and requires confidentiality and privacy, as it will handle third-party data.

Thus, the reported study in [8] has proposed a protocol that preserves the privacy of the data using the disjunctive normal form, called PPDNF (Privacy Preserving Disjunctive Normal Form) in operations with distributed set data, without revealing any data beyond the information inferred from the input operation. The fully homomorphic encryption scheme is used to ensure the safety of the protocol and perform operations on encrypted data. The structure of PPDNF protocol does not depend on the structure of the cryptographic system, however, the efficiency of the proposed protocol depends on the efficiency of the fully homomorphic encryption scheme.

4.6 Preserving Privacy with Protocol SPIR

Data privacy is a natural and fundamental requirement in many contexts, an example occurs in a commercial database that sells information to users, such as stock information, and is collecting the amount of data that the user retrieves. In this example, both the privacy and confidentiality of the user are essential [13]. The Symmetric PIR (SPIR) protocol should prevent the user to learn more than one record from the database during a session. The main method to measure the cost of such systems, like the SPIR protocol, is by its communication complexity.

The SPIR model (Symmetrically Private Information Retrieval) aims to ensure data privacy and user privacy, ie. each invocation of a SPIR scheme, while maintaining the user's privacy, the scheme should also prevent the user (even as a

rogue) from getting any information other than a single physical bit of data.

The evaluation of the SPIR protocol, proposed in [25], is based on fully homomorphic encryption presents an improvement in the communication complexity making it ideal. Furthermore, the step of bootstrapping FHE makes the protocol computationally less efficient.

4.7 Execution of SQL Queries on Encrypted Data Base

In the study "A Secure Database System using homomorphic Encryption Schemes", the authors Youssef Gahi, Mouhcine Guennoun and Khalil El - Khatib [11] present a technique for executing SQL statements about the encrypted data. They developed a system of secure database processing such queries. The parameters of SQL queries are encrypted by the client and sent to the server for proper processing. The server performs the requested operation in an encrypted database and returns an encrypted result to the client. The advantage of this system is that the database server does not get the content nor the position of the records generated by the query. Tests with the fully homomorphic encryption scheme proposed by Craig Gentry were conducted to verify the efficiency and safety of the method.

The proposed system supports a set of SQL operations such as SELECT, UPDATE, DEL and statistical operations such as COUNT and AVG. During the simulation, the researchers claim that the fully homomorphic encryption scheme is not efficient, citing performance issues with the proposed solution.

4.8 Homomorphic Signatures for Polynomial Functions

The authors Dan Boneh and David Mandell Freeman [24], presented the study "Homomorphic Signatures for Polynomial Functions", which recounts the first homomorphic signature scheme capable of evaluating multivariate polynomials on signed data. Thus, the authors' focus is on the functions that perform arithmetic operations on a set of data such as mean, standard deviation, and other data mining algorithms.

With this approach it is possible to store data and their signatures in an untrusted storage server. If a third party needs to make some calculation, as the standard deviation, it can order directly to the untrusted server that will perform the requested calculation and its homomorphic signature from the signatures of each stored data. With homomorphic signature it is possible to validate the integrity of the result.

However, if you publish a pair (f, σ), where f is the result of the operation of the standard deviation and σ is the homomorphic signature bypass operation (derived from calculations homomorphic with the signing of other data), any party who owns the Alice's public key can check the signature and verify the integrity of the results generated by the untrusted server over the computed data.

This method can be applied to a tool known as a decision tree that is widely used in data mining processes. The decision tree is widely used in ranking algorithms, it is a simple representation of knowledge and an efficient way to build classifiers that predict or indicate classes or information based on the values of attributes of a dataset. The decision tree is very useful in the process of extracting previously unknown information from large databases.

This study is presented as the first step towards fully homomorphic signature, since the proposed system uses ideal lattices that is the basis of the proposal of CCH presented by Craig Gentry.

In this context, if the fully homomorphic signature is implemented (based on arbitrary calculations on signed data), an untrusted server can execute more complex data mining algorithms about a particular data set and verify the authenticity of the transactions. For example, given a set of signed data, the server could publish a signed decision tree. As the signatures are private, if publish a signed decision tree, the original dataset will not be exposed and can be permitted to validate the published results.

A fully homomorphic signature scheme would be a parallel system and useful for existing fully homomorphic encryption schemes. Even if a fully homomorphic signature scheme was not developed, it would be very useful to amplify the set of admissible functions proposed in the present study [24].

5 Conclusions

Preservation of privacy and confidentiality for data mining is a problem for data owners and many researchers devote efforts to solve this problem. A fully homomorphic encryption is presented as an amazing possibility for solving this challenge.

Homomorphic encryption provides privacy and security for data processing in untrusted environments, and can be used in protocols that perform the steps of data mining. Thus, it was shown in this study the significance and importance of using a fully homomorphic cryptographic scheme in the application of data mining.

Analyzing the different cataloged studies, it was observed that each author seeks to solve the problem of protecting confidentiality and privacy in data mining, addressing it in isolated parts of the mining process.

All analyzed studies make it clear that the safety of the proposed solution depends on the security of the cryptographic system completely, i.e. if the cryptosystem is secure then the proposed scheme is also secure.

It is quite clear that homomorphic encryption still requires deeper studies in order to become a practical and efficient implementation that will succeed in the preservation of security, integrity, privacy and confidentiality of data in several areas of data mining.

Another noticeable point is that the efficiency of the schemes that have been proposed in the studies catalogued, also depends on the efficiency of the fully homomorphic cryptosystem. At this point the cryptographic system still does not meet the desired requirements, but scholars are seeking to solve this problem with new solutions.

6 Future Works

This section is an important point in the research, it allows for opportunities for future work aimed to continue this research. Aiming to highlight these opportunities there are some suggestions presented for new referrals that were identified during the study in question:

- Conduct a search through the references presented in the studies cataloged in order to identify new studies;
- Evaluate the proposed methods to verify the possibility of integrating them with the goal of keeping all steps of data mining safe and preserving data privacy;
- Assess new practical and efficient fully homomorphic encryption schemes that can be applied in data mining;
- Resolve or improve the computational cost of the cryptographic process used in the fully homomorphic encryption.

As shown above, by using homomorphic encryption in data mining is not feasible due to the cost of the cryptographic system processing. If the fully homomorphic cryptosystems are improved, then the trend is that more protocols to the steps of data mining will be developed, requiring less processing on cryptographic regard and devoting greater computational power to the mining process.

7 References

[1] AGRAWAL, R., e SRIKANT, R. "Privacy preserving data mining". Proceedings of ACM SIGMOD Conference on Management of Data (SIGMOD'2000), Dallas, TX.

[2] BONEH, Dan; FREEMAN, David Mandell. "Homomorphic Signatures for Polynomial Functions". v. 6632, p. 149–168, 2011.

[3] BONEH, Dan; SAHAI, Amit; WATERS, Brent. "Functional Encryption: A new vision fo Public-Key Cryptography". Communications of the ACM, Vol. 55 No. 11, Pages 56-64.

[4] BRAKERSKI, Z.; VAIKUNTANATHAN, V. "Efficient Fully Homomorphic Encryption from (Standard) LWE". 2011 IEEE 52nd Annual Symposium on Foundations of Computer Science, p. 97–106, 2011. IEEE.

[5] BRICKELL, Justin Lee. "Privacy-preserving Computation Data Mining". 2009.

[6] 17 BRUSSO, M. J. "Access Miner: Uma proposta para a Extração de Regras de Associação Aplicada à Mineração do Uso da Web". Master's thesis, PPGC da UFRGS, Porto Alegre - RS, 2000.

[7] CHOR, B.; KUSHILEVITZ, E.; GOLDREICH, O.; SUDAN, M. Private information retrieval. Journal of the ACM, v. 45, n. 6, p. 965–981, 1998.

[8] CHUN, JI Young; HONG, Dowon; JEONG, Ik Era; LEE, Dong Hoon. "Privacy-preserving disjunctive normal form operations on distributed sets". Information Sciences, v. 231, p. 113–122, 2013.

[9] EVFIMIEVSKI, Alexandre; GRANDISON, Tyrone Grandison. "Privacy-Preserving Data Mining". IBM 2009.

[10] FANG, Weiwei. ZHOU, Changsheng. YANG, Bingru. "Privacy Preserving linear regression modeling of distributed databases". Optimization Letters, v. 7, n. 4, p. 807–818, 2012.

[11] GAHI, Youssef; GUENNOUN, Mouhcine; EL-KHATIB, Khalil. "A Secure Database System using Homomorphic Encryption Schemes. Security", n. c, p. 54–58, 2011.

[12] GENTRY, Craig. "Fully homomorphic encryption using ideal lattices". In: Proceedings of the 41st annual ACM symposium on Theory of computing. New York, NY, USA: ACM, 2009, p. 169–178.

[13] GERTNER, Y.; ISHAI, Y.; KUSHILEVITZ, E.; MALKIN, T. "Protecting Data Privacy in Private Information Retrieval Schemes". STOC '98 Proceedings of the thirtieth annual ACM symposium on Theory of computing. Pages 151-160 1998.

[14] HAN, Jing-Li; YANG, MIng; WANG, Cai-Ling e XU, Shan-Shan. "The Implemention and Application of Fully Homomorphic Encryption Scheme". Second International Conference on Instrumentation, Measurement, Computer, Communication and Control, p. 714–717, IEEE, 2012.

[15] KAOSAR, M. G.; PAULET, R.; YI, X. "Fully homomorphic encryption based two-party association rule mining". Data & Knowledge Engineering, v. 76-78, p. 1–15, 2012. Elsevier B.V.

[16] LASKARI, E. C.; MELETIOU, G. C.; TASOULIS, D. K.; VRAHATIS, M. N.; "Data Mining and Cryptology". Proceedings of the International Conference of Computational Methods in Sciences and Engineering (ICCMSE 2003), T.E. Simos (ed.), P. 346-349, World Scientific Publishing, 2003.

[17] LINDELL, Y., e PINKAS, B. "Privacy preserving data mining". In Lecture notes in computer science. Vol. 1880. Proceedings of Advances in Cryptology: Crypto' 2000 (pp. 20-24). Springer-Verlag.

[18] NIKOLAENKO, Valeria; BONEH, Dan. "Data-Mining on GBytes of Encrypted Data". Stanford 2013 Security Workshop.

[19] PATEL, Smita D.;TIWARI, Sanjay."Privacy Preserving Data Mining". International Journal of Computer Science and Information Technologies. 2013.

[20] RAMAIAH, Y. Govinda; KUMARI, G. Vijaya. "Efficient Public Key Homomorphic Encryption Over Integer Plaintexts. Information Security and Intelligence Control (ISIC)", 2012 International Conference on IEEE.

[21] RIVEST, R L; ADLEMAN, L; DERTOUZOS, M L. "On data banks and privacy homomorphisms", in R. A. Demillo et al. In Eds.), Foundations of Secure Computation, pages 169–179. Academic Press, 1978.

[22] RIVEST, R. L.; SHAMIR, a.; ADLEMAN, L. "A method for obtaining digital signatures and public-key cryptosystems". Communications of the ACM, v. 21, n. 2; p. 120-126, 1978

[23] SMART, Nigel P.; VERCAUTEREN, Frederik. "Fully Homomorphic Encryption with Relatively Small Key and Ciphertext Sizes". PKC 2010, LNCS 6056, pp. 420–443, 2010.

[24] YAO, Andrew C. "Protocols for Secure Computations". IEEE Foundations of Computer Science, 1982, Pag. 160-164. SFCS '08. 23rd Annual Symposium. DOI: 10.1109/SFCS.1982.38.

[25] ZHONG, Hong; YI, Lei; ZHAO, Yu; YUAN, Xiaping e SHA, Xianju. "Fully-homomorphic Encryption Based SPIR". n. 60773114, p. 4–6, 2011.

Secure Outsourcing Multiparty Computation on Lattice-based Encrypted Cloud Data under Multiple Keys

Yi Sun, Qiaoyan Wen, Hua Zhang, Zhengping Jin
State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing, China

Abstract - *In secure multiparty computation, the computation and communication complexities of each user always depend polynomially on the complexity of the function to be computed. Therefore, users look for a better method that can liberate them from troublesome computations especially when the computing function is too complex. The emergence of the cloud, which is powerful in storing and computing, makes this possible. In this paper, we focus on how to securely outsource the computation task to the cloud and propose a secure outsourcing multiparty computation protocol on lattice-based encrypted cloud data under multiple keys in two cloud servers scenario. The host-cloud server firstly transforms the ciphertexts encrypted by different users' public keys to the ones blinded by the assist-cloud and than operates on these transformed ciphertexts. Related users who are authorized to get the result only need to decrypt the returned custom-made ciphertexts for them respectively to recover the desired result.*

Keywords: Outsourcing Computation, Secure Multiparty Computation, Lattice-based Encryption, Cloud Computing

1 Introduction

Secure Multiparty Computation (SMC) is dedicated to deal with the problem of secure computation among distrustful participants. It was first introduced by Yao in 1982 [1], and then was extended by Goldreich, Micali, Wigderson [2] and many other researchers [1-7]. Generally speaking, SMC is a method to implement cooperative computation with participants' private data, ensuring the correctness of the computation as well as not disclosing additional information except the necessary results. It has become a research focus in the international cryptographic community due to its wide applications in various areas and a mass of research results have been published one after another. Informally speaking, assuming that there are m participants, $P_1, P_2, \ldots, P_m$, each of them has a private number, respectively $x_1, x_2, \ldots, x_m$. They want to cooperate to compute the function $y = f(x_1, x_2, \ldots, x_m)$ without revealing x_i of P_i to other parties P_j, $j \neq i, i, j \in \{1, \ldots, m\}$, as well as guaranteeing that any unauthorized ones cannot get the result y. In the past, researchers mainly focus on designing the style of secure multiparty computation protocols that users themselves cooperatively complete the function evaluation through their internal interactions [1, 2, 8-11]. The computation and communication complexities always depend polynomially on the complexity of the function to be computed. Therefore, users suffer from the heavy overload of these protocols.

The emergence of the cloud [12, 13] makes it possible for users to apply the powerful computing ability of the cloud to conduct complex computations especially in outsourcing computation to the cloud where users expect that the cloud can independently complete any function computation on their outsourced data although the data has been encrypted by their own keys for security. Moreover, the final result should be kept private to the cloud even though it is the cloud that completes all of the computations about the computing function. In this way, users only need to encrypt their data and decrypt the returned message to get the desired result. All computations about the computing function are in the charge of the cloud. There is non-interaction of users whatsoever and the computation and communication complexities of each user are independent of the function. However, this expectation is proven to be impossible in the single cloud server setting due to the impossibility of program obfuscation [14]. In this paper, we try to solve this problem by introducing one more cloud server to the original model described above. More precisely, we consider the following scenario:

There are m+2 distrusted parties including m users and two cloud servers in our system. We assume that all of them act semi-honestly. The m users $P_1, P_2, \ldots, P_m$, each having a private input, respectively x_i, as well as a pair of public-private keys denoted as $(pk_i, sk_i), i = 1, \ldots, m$, encrypt their respective private inputs by their own public keys and then upload the ciphertexts of the inputs to a cloud server. They want to obtain the value $y = f(x_1, x_2, \ldots, x_m)$ even if they may not be aware of what the computing function $f(\cdot)$ is, by applying two cloud servers to operate on the outsourced encrypted data without revealing $x_1, x_2, \ldots, x_m$ as well as the result y.

In this paper, we study the outsourcing computation problem in multiple users-two cloud servers scenario and propose a secure outsourcing multiparty computation protocol to compute any function on lattice-based encrypted data

under multiple keys of the users. Herein, we apply one cloud server called as the host-cloud (HC) to store the outsourced data encrypted by users and another cloud server called as the assisted-cloud (AC) to help HC to compute $f(\cdot)$ on the encrypted data. Our proposed protocol is completely non-interactive between any users and both of the computation and communication complexities of each user are independent of the computing function.

2 Related works

As mentioned above, previous research mainly focuses on designing the kind of secure multiparty computation protocols that the users themselves cooperatively complete function evaluation through their internal interactions [1, 2, 8-11]. Users suffer a lot from the heavy overload of the computation and communication complexities, which always depend polynomially on the complexity of the function to be computed. Therefore, a secure protocol with the least interactions and computations is the most preferred solution to all users. In fact, when computing a function $f(\cdot)$, the inputs of the computing function are from the users and the result (or some intermediate result that can help users to get the final result) has to be returned to a certain set of users who are authorized to know the final result. Hence, the two rounds of communication, that is, sending the inputs from users and receiving the returned result from others, cannot be avoided no matter how the protocols are constructed. Thus, we wish to obtain a secure protocol that makes users cost the least computations with the least two rounds of communication.

In this aspect, secure outsourcing computation [15-19] to the powerful cloud, which can liberate users from heavy computations and communications by having the cloud to complete the computation task on behalf of the users, is a promising solution. In cloud scenario, users store their data on the cloud in the encrypted forms by their own public keys respectively and then the cloud conducts corresponding computations on the encrypted data instead of the original private data to obtain an intermediate result encrypted by user's public key so that the user can finally obtain the desired result by decrypting the returned massage using its own private key. Compared with general secure multiparty computation, the cloud server in secure outsourcing computation can be seen as a party that is more powerful in computing than other usual parties and thus we expect the cloud to do computations related to the computing function as more as possible.

Obviously, by completely outsourcing computation to the cloud, users no longer need to do any computations about the computing function and even can be unaware of what the computing function is. The computation complexity for a user is only related to the encryption scheme it has used when encrypting the private data but no longer depends on the computing function. It can be regarded as the most preferred computing model to users. What's more, we can find that it is also possible to completely avoid the interaction of users if the encryption schemes are chosen appropriately. In short, there are two aims we want to achieve when designing secure outsourcing protocols. That is,

> **a**. The cloud is to do all of the computations related to the computing function while users would do nothing except encrypting their private inputs and decrypting the returned result.
>
> **b**. There is no interaction between any users.

However, it is not so easy to achieve the two aims as we have thought. There are many problems to be considered.

2.1 To the users

2.1.1 Privacy of the inputs

In secure outsourcing computation, users have to contribute their private data as the inputs of the function while not participating in the computation process. Moreover, all parties of the protocol including all users and cloud servers are mutually distrusted. Therefore, users would not like to submit their private data to the cloud. Allowing for security, a usual solution is to encrypt the private data before outsourcing them to the cloud. And there are some basic encryption models according to the encryption keys users used.

(1) The public key of the cloud server

The naive method is to assume that there is a trusted cloud. Users encrypt their private inputs under the public key of the cloud server and then submit the ciphertexts to it. The cloud can complete any computation on these private inputs by first decrypting the ciphertexts and then directly computing on the decrypted inputs. However, in reality, once the cloud is corrupted, the private inputs as well as the result will be revealed immediately. Users would not like to bear this risk.

(2) The joint public key of the users

In 2009, Gentry [20] presents a new model where all users use a joint public key to encrypt their own private inputs while sharing the private key. Therein, the cloud cannot obtain the inputs nor the result because they are protected by the encryption scheme while the cloud does not have the private key. However, users have to participate in another interactive protocol to firstly recover the private key and then achieve the desired result. The processes, producing a joint public key, sharing the private key, and jointly recovering the result by their shared private key, bring large rounds of additional interaction among the users, which is contrary to our expectation that we want to design a secure protocol with the least communications, just sending out the inputs and receiving the result. Herein, in cloud outsourcing scenario, it means that there are no interactions among the users

whatsoever except the least two rounds of interactions between the user and the cloud server.

(3) Respective public keys of the users

A recent work by Asharov et al. [21] proposes a scheme where users utilize their own public keys to encrypt their inputs respectively and guarantees that the cloud can succeed in computing the function on their private inputs by computing on the ciphertexts of the inputs encrypted under different keys. Although users still have to interact to obtain the result in the last step, encrypting respective input by the public key of each user is the best encryption model so far.

2.1.2 Privacy of the result

In 2011, Halevi et al. [22] propose a non-interactive protocol to securely realize outsourcing computation. Therein, the server is entitled to learn the result. However, the computing result maybe some vital information to the users in some scenario and so it cannot be revealed to others. Hence, besides the security of the inputs discussed above, users must consider the security of the result when constructing protocols. It should guarantee that any unauthorized users are not able to get the result although they may contribute their inputs and the cloud is not able to get the result although the result is computed by the cloud. To this aspect, reference [20] have already protected the result by a joint public key of the users. However, each authorized user is also not able get it individually. It does not achieve the first aim because of the additional interactions when recovering the result. How to guarantee the security of the result without effecting the two aims is what we should consider in this work.

2.2 To the cloud

2.2.1 Feasibility of operating on encrypted inputs

As discussed above, users would like to upload the encrypted inputs under their respective public keys to the cloud server rather than the original inputs. Therefore, the cloud, whose aim is to compute a function on users' private inputs, would only obtain the ciphertexts of the inputs. That means, the cloud has to compute the function on users' private inputs through performing corresponding computations on the ciphertexts of the inputs encrypted by different public keys of users. As we know, fully homomorphic encryption (FHE) [20, 23] can operate on the ciphertexts of the inputs to compute the desired result produced by the inputs. But the usual FHE schemes are single-key in the sense that they only can perform computations on ciphertexts encrypted under the same key. It is not feasible to conduct computations on the ciphertexts encrypted under different keys.

In order to solve this problem, Adriana López-Alt et al. [24] propose a new FHE called as multikey fully homomorphic encryption (MFHE) which has applied the techniques of bootstrapping, modulus reduction and relinearization to operate on the ciphertexts of the inputs encrypted by multiple, unrelated keys. When outsourcing private data to the cloud, user can firstly encrypt it by its own key by applying MFHE. It is indeed the optimal solution from the point of view of the feasibility of operating on ciphertexts and the privacy of inputs. However, as we mentioned before, it is still not satisfactory because users need to evaluate the decryption key and then use it to recover the result interactively by participating in another SMC protocol.

It seems that it is too difficult to achieve a secure outsourcing multiparty computation protocol in multiple users-one cloud server scenario with no interaction of users. In fact, according to [14], it is proved that it is indeed impossible to construct a completely non-interactive protocol in single server setting due to the impossibility of program obfuscation. Hence, if we want to obtain a secure protocol with completely no interaction of users in outsourcing computation, we need at least one more cloud server.

In short, combining the two aims and all of the above factors users and cloud servers should consider, we can conclude that if we want to construct a completely non-interactive secure outsourcing multiparty computation protocol where the computation and communication complexities of each user are independent of the computing function, it is reasonable to consider it in two cloud servers scenario. Moreover, we should allow users to encrypt their inputs by their own public keys and obtain the result by decrypting the ciphertexts of the result by themselves as well as guaranteeing that the result will not be revealed to the cloud servers. Thus, in this paper, we follow the idea of [19, 25] to propose a secure outsourcing multiparty computation protocol to compute desired functions by users' ciphertexts. Compared with our precious work [25] in this area, we herein deal with the ciphertexts under a different encryption algorithm and similarly prove the security of our scheme.

3 Preliminaries

3.1 Lattice-based encryption

Since the privacy of the inputs and the computation complexity of each user depend on the encryption algorithm he used, an encryption scheme both outstanding in security and efficiency is the right one users want to adopt. Hence, lattice-based encryption, which is against quantum attacks and much more efficient than RSA and even the elliptic curve cryptosystem, becomes the first choice of rational users. Herein, we will show how the two cloud servers deal with the outsourced data encrypted by the lattice-based public key encryption scheme proposed in [24] (denoted as LE scheme in this paper). Specifically, we remind it as follows.

Notations: Let k be the security parameter. Then the LE scheme is parametrized by a prime $q = q(k)$ and B-bounded error distribution χ over the ring $R_q = Z_q[x]/ < x^n + 1 >$, i.e., the ring of degree n polynomials modulo $x^n + 1$ with

coefficients in Z_q. All operations in the LE scheme take place in the ring $R_q = R_q / qR$ (i.e. modulo q and $x^n + 1$).

The LE encryption scheme consists of the following three algorithms $(KeyGen(\cdot), Enc(\cdot), Dec(\cdot))$.

$KeyGen(1^k)$: Sample polynomials f' , g from the distribution χ , denoted as $f', g \leftarrow \chi$. Then the private key is $sk := f = 2f' + 1$, the public key is $pk := 2gf^{-1}$. If f is not invertible in R_q, resample f'.

$Enc(pk, m)$: Sample polynomials $s, e \leftarrow \chi$. Output the ciphertext $c := hs + 2e + m$, where all operations are done modulo q and $x^n + 1$.

$Dec(sk, c)$: Compute $\mu := fc \in R_q$ and output $m' := \mu (\mathrm{mod}\, 2)$.

In fact, this scheme is a variant of the one of the earliest lattice-based public key encryption schemes [26]. In reference [24], they make some changes to the original scheme to get the LE scheme and then apply the techniques of bootstrapping, modulus reduction and relinearization to operate on the ciphertexts of the inputs encrypted by multiple, unrelated keys. Therein, they have obtained a secure outsourcing multiparty computation protocol on lattice-based encrypted data under multiple keys of users in one server scenario. However, it is not satisfactory because the interaction in the decryption stage is still inevitable. In this paper, based on this encryption scheme, we consider the outsourcing problem in two cloud servers scenario and succeed to construct a secure non-interactive outsourcing protocol that achieves the two aims mentioned at the beginning of this paper.

3.2 Security model

In this paper, we will discuss our protocol in the semi-honest model and analyze its security using the real-ideal paradigm [5].

Firstly, in the ideal world, the computation of the functionality $\mathcal{F}$ on users' private inputs is conducted by an additional trusted party, that receives x_i from user $P_i, i = 1, 2, \ldots, m$, and returns the result $f(x_1, x_2, \ldots, x_m)$ to the authorized users P_t while other unauthorized parties do not get any output. Hence, in the ideal world, all users' private inputs are well protected and only authorized users are able to learn the result. However, there is no trusted party in the real world and so all parties have to run a protocol Π to get the desired result. During executing protocol Π, all parties act semi-honestly following the protocol and make effort to gain more information about other parties' inputs, intermediate results, or overall outputs by the transcripts of the protocol. An adversary can corrupt a party to receive all messages directed to it and control the messages to be sended out from it.

Herein, we denote the joint output of the ideal world adversary $\mathcal{S}$ and the outputs of the rest parties in an ideal execution for computing the functionality $\mathcal{F}$ with inputs $\vec{x} = (x_1, x_2, \ldots, x_m)$ as $IDEAL_{\mathcal{F},\mathcal{S}}(\vec{x})$, the joint output of the real world adversary $\mathcal{A}$ and the outputs of the rest parties in an execution of protocol Π with inputs $\vec{x} = (x_1, x_2, \ldots, x_m)$ as $REAL_{\Pi,\mathcal{A}}(\vec{x})$. Then, we say that protocol Π securely realizes functionality $\mathcal{F}$ if for every real adversary $\mathcal{A}$ corrupting any parties and possibly the cloud servers, there exists an ideal world adversary $\mathcal{S}$ with black-box access to $\mathcal{A}$ such that for all input vectors $\vec{x}$, $IDEAL_{\mathcal{F},\mathcal{S}}(\vec{x})$ is computationally indistinguishable to $REAL_{\Pi,\mathcal{A}}(\vec{x})$, that is,

$$IDEAL_{\mathcal{F},\mathcal{S}}(\vec{x}) \overset{c}{\approx} REAL_{\Pi,\mathcal{A}}(\vec{x}) .$$

4 Our result

We consider the secure outsourcing computation problem in the multiple users-two cloud servers scenario described as follows.

There are m+2 parties including m users and 2 non-colluding cloud servers, one is called as the host-cloud HC and the other is called as the assisted-cloud AC. Each user P_i has a private input x_i and a pair of public-private keys (pk_i, sk_i) while sharing a private random r_i with HC. AC has a pair of public-private keys (k_{AC}, k_{AC}^{-1}) . We want to outsource the task of computing function $f(\cdot)$ on users' private inputs to HC, who stores the ciphertexts of these data encrypted by a lattice-based encryption scheme under users' public keys, while keeping the security of the inputs and ensuring only the authorized users to get the result. What's more, we wish that the cloud servers take charge of all of the computations related to the function $f(\cdot)$ and there is no interaction of users whatsoever so that the computation and communication complexities of each user are independent of the function to be computed.

Herein, we deem that the two rounds of inevitable communications and a request from a user to the cloud servers for computing function $f(\cdot)$ are the three basic rounds of communication in this paper. Then for each user, they expect that there is no other interactions at all between any user-to-user or user-to-server except the three basic rounds of communication. Furthermore, the computation complexity of each user depends on the encryption scheme it has used.

The framework of our construction can be illustrated in Fig. 1.

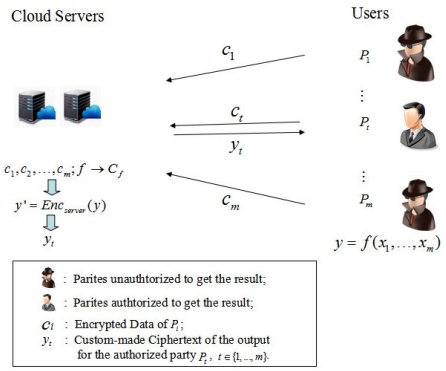


Fig .1. Framework of our construction

In this following section, we formally propose our solution denoted as protocol Π for convenience in detail and then analyze its security using the real-ideal paradigm in the semi-honest model.

Without loss of generality, we represent the function $f(\cdot)$ to be computed by means of arithmetic circuit $\mathcal{C}_f$ consisting of any number of addition gates and l multiplication gates where each gate has two input wires and one output wire. Then any functionality can be reduced to the two basic operations, addition and multiplication over two inputs. Our construction can be summarized as follows:

Protocol Π

Setup

For $i = 1, 2, \ldots, m$, user P_i obtains the public-private keys (pk_i, sk_i) while sharing a private random r_i with HC. AC obtains a pair of public-private key (k_{AC}, k_{AC}^{-1}) . As a preparation, user P_i firstly sends $\mathrm{r}_i \cdot f_i$ to AC; AC computes $k_{AC} \cdot \mathrm{r}_i \cdot f_i$ and then sends it to HC. Then, HC obtains $\varphi_i = k_{AC} \cdot f_i$.

Upload

For $i = 1, 2, \ldots, m$, each user P_i encrypts its own private input x_i by the LE scheme as $c_i = h_i s_i + 2e_i + x_i$, where $s_i, e_i \leftarrow \chi$, and then uploads it to HC.

Computation

After receiving all ciphertexts of the private inputs from users, HC computes the function $f(\cdot)$ in three steps following the circuit $\mathcal{C}_f$.

1. *Transforming*.

Firstly, HC transforms the ciphertexts encrypted by users' own keys to the ciphertexts blinded by AC as $c_i^{AC} = \varphi_i \cdot c_i = k_{AC} \cdot f_i \cdot (h_i s_i + 2e_i + x_i) \bmod 2 = k_{AC} \cdot x_i$.

2. *Computing*.

And then, HC computes the ciphertext of the result by the transformed ciphertexts of inputs.

Add. For any addition gate,

$$c_i^{AC} \oplus c_j^{AC} = k_{AC} \cdot (x_i + x_j) ;$$

Mul. For any multiplication gate,

$$c_i^{AC} \otimes c_j^{AC} = k_{AC}^{\ 2} \cdot (x_i \times x_j)$$

3. *Invert-transforming*.

After computing gate by gate following the circuit $\mathcal{C}_f$ of $f(\cdot)$, HC obtains the intermediate result encrypted by the private key of AC, that is, $y' = k_{AC}^{\ l+1} \cdot y$, where $y = f(x_1, x_2, \ldots, x_m)$ and l is the number of the multiplication gates of $\mathcal{C}_f$. By invert-transforming, HC computes the custom-made ciphertext y_t for the authorized party $P_t, t \in \{1, 2, \ldots, m\}$ by $y_t = (\varphi_t^{-1})^{l+1} = (f_t^{-1})^{l+1} \cdot (k_{AC}^{-1})^{l+1} \cdot y' = (f_t^{-1})^{l+1} \cdot y$.

Output

For any authorized party $P_t, t \in \{1, 2, \ldots, m\}$, it obtains the result y by $y = f_t^{l+1} \cdot y_t = f_t^{l+1} \cdot (f_t^{-1})^{l+1} \cdot y$.

In setup, each user P_i invokes $KeyGen(1^k)$ to compute its public-private keys (pk_i, sk_i) . At the same time, each P_i selects a random r_i and sends it to HC via a secure channel while AC chooses a pair of public-private keys (k_{AC}, k_{AC}^{-1}) . Assuming that all users' private data $x_i, i = 1, 2, \ldots, m$, are the real inputs of function $f(\cdot)$, then each P_i sends $\mathrm{r}_i \cdot f_i$ to AC. AC further computes $k_{AC} \cdot \mathrm{r}_i \cdot f_i$ and sends it to HC. After that, P_i submits the ciphertext c_i of the private input x_i encrypted by its own public key f_i to HC in the upload process.

In computation process, all related computations about $f(\cdot)$ are conducted by HC in three steps.

Firstly, HC transforms the outsourced data which are encrypted by different keys of users to the ciphertexts which are blinded by the same key of AC so that HC can do further computations.

Then, for each gate, HC can easily get the intermediate result by adding/multiplying the two transformed ciphertexts of the inputs. Computing gate by gate following the circuit $\mathcal{C}_f$, HC can obtain an intermediate result y' .

In the last process, HC produces the custom-made ciphertext y_t by invert-transforming y' to y_t for each authorized user $P_t, t \in \{1,2,\ldots,m\}$ who is designated to get the result so that they can obtain it by decrypting y_t using its own private key.

5 Analysis

From the protocol described above, the correctness is obvious due to the homomorphic properties of the transformed ciphertexts. We will have a detailed discussion on its security. Note that before the actual computations which are performed by HC, there are setup and upload processes. We will individually illustrate the security of these processes at first. Afterwards, we will prove the security of the core of our protocol, i.e. the actual computation process, in the real-ideal framework. Finally, from the composition theorem [5], we can conclude that our protocol is secure.

Theorem 1. Protocol Π is secure as long as the LE scheme is secure and HC and AC are non-colluding.

Proof. Firstly, we look at the setup and upload processes individually. In setup, each user respectively encrypts its private input by its own public key which is produced by invoking a semantically secure LE scheme. AC produces a pair of public-private keys (k_{AC}, k_{AC}^{-1}) by itself. The security in these two processes are obvious. Afterwards, P_i sends $\mathrm{r}_i \cdot f_i$ to AC and AC sends $k_{AC} \cdot \mathrm{r}_i \cdot f_i$ to HC. Herein, P_i's private key f_i is protected by the blinding factors r_i, which is private to P_i and HC, and k_{AC} which is private to AC. Therefore, the private keys of users will not be revealed in this process. In upload, users outsource the encrypted data to HC. Since the LE scheme is semantically secure, given two ciphertexts $c_i(m_1), c_i(m_2)$ uploaded by P_i, it is computationally infeasible for HC to distinguish the two ciphertexts. Hence, users can store their encrypted data in HC securely.

In the actual computation process, we will discuss the security in the real-ideal framework. From the security definition, we say that protocol Π is secure if all adversarial behavior in the real world can be simulated in the ideal model where exists an additional trusted party to perform all computations related to the function $f(\cdot)$ to be computed. We assume that there is a simulator $\mathcal{S}$ in the ideal world and then prove that it can simulate the semi-honest adversary $\mathcal{A}$ that exists in the real execution. Since HC is able to independently complete addition and multiplication operations, we only need to prove that **Add** and **Mul** are secure against the semi-honest adversary $\mathcal{A}$ corrupting HC. We prove this as follows,

Simulator $\mathcal{S}$: Run $\mathcal{A}$ on input $\{c_{\mathcal{S}}(m_1), c_{\mathcal{S}}(m_2)\}$.

Firstly, $\mathcal{S}$ computes

$$c_{\mathcal{S}}(m_1) = Enc(pk_{\mathcal{S}}, 1);$$
$$c_{\mathcal{S}}(m_2) = Enc(pk_{\mathcal{S}}, 1).$$

and sends $c_{\mathcal{S}}(m_1), c_{\mathcal{S}}(m_2)$ to $\mathcal{A}$.

Secondly, $\mathcal{A}$ sends two ciphertexts $c_{\mathcal{S}}(m_1^*), c_{\mathcal{S}}(m_2^*)$ to $\mathcal{S}$. Then, $\mathcal{S}$ computes

$$c_{\mathcal{S}}(m_1^* + m_2^*) = c_{\mathcal{S}}(m_1^*) \oplus c_{\mathcal{S}}(m_2^*);$$
$$c_{\mathcal{S}}(m_1^* \times m_2^*) = c_{\mathcal{S}}(m_1^*) \otimes c_{\mathcal{S}}(m_2^*).$$

and returns $c_{\mathcal{S}}(m_1^* + m_2^*)$, $c_{\mathcal{S}}(m_1^* \times m_2^*)$ to $\mathcal{A}$.

Finally, $\mathcal{S}$ outputs what $\mathcal{A}$ outputs.

Now, we can prove the security of **Add** and **Mul** algorithms by contradiction. Firstly, we assume that the view of the adversary $\mathcal{A}$ in the real world is distinguishable from the view simulated by the simulator $\mathcal{S}$. Then, we could find an algorithm to distinguish the ciphertexts encrypted by the LE encryption scheme, which is contrary to our assumption that the LE is semantically secure. Hence, the view of the adversary $\mathcal{A}$ in the real world is indistinguishable to the view simulated by the simulator $\mathcal{S}$. That is,

$$IDEAL_{\mathcal{F},\mathcal{S}}(c_{\mathcal{S}}(m_i)) \overset{c}{\approx} REAL_{\Pi,\mathcal{A}}(c_{\mathcal{S}}(m_i)), i = 1,2.$$

Therefore, the two algorithms **Add** and **Mul** are secure. Furthermore, from the composition theorem [5], we can conclude that our protocol is secure as long as the LE scheme is secure and HC and AC are non-colluding in semi-honest scenario.

6 Conclusions

In this paper, we construct a secure outsourcing multiparty computation protocol on lattice-based encrypted cloud data under multiple keys in two non-colluding cloud servers scenario, which greatly facilitates privacy-preserving computations among distrusted parties. We also give the security proof in semi-honest model. How to make it secure in the malicious model is our next work.

7 Acknowledgment

This work is supported by NSFC (Grant Nos. 61300181, 61272057, 61202434, 61170270, 61100203, 61121061), the Fundamental Research Funds for the Central Universities (Grant No. 2012RC0612, 2011YB01).

8 References

[1] A. C. Yao, "Protocols for secure computations"; Proceedings of the 23rd Annual IEEE Symposium on Foundations of Computer Science, Chicago, pp. 160-164, 1982.

[2] O. S. Goldreich. S. Mical, A. Wigderson, “How to Play any Mental Game”; Proceedings of the nineteenth annual ACM symposium on Theory of computing STOC’87, New York: ACM, pp. 218-229, 1987.

[3] Y. Lindell, B. Pinkas. “A proof of Yao’s protocol for secure two-party computation”; Journal of Cryptology, vol. 22, pp. 161-188, 2009.

[4] O. S. Goldreich, “Secure multiparty computation”; Manuscript, Preliminary version, 1998.

[5] O. S. Goldreich, “Foundations of Cryptography: Volume 2, Basic Applications”. Cambridge University press, 2004.

[6] M.M. Prabhakaran, A. Sahai, eds., “Secure multiparty computation”. IOS press, 2013.

[7] R. Fagin, M. Naor, P. Winkler, “Comparing information without leaking it”; Communications of the ACM, vol. 39, pp. 77-85, 1996.

[8] D. Chaum, C. Crépeau, I. Damgård, “Multiparty unconditionally secure protocols (extended abstract)”; STOC, ACM, pp. 11-19, 1988.

[9] I. Damgård, V. Pastro, N.P. Smart, S. Zakarias, “Multiparty computation from somewhat homomorphic encryption”; CRYPTO, LNCS, Springer, vol. 7417, pp. 643-662, 2012.

[10] Y. Lindell, B. Pinkas, “An efficient protocol for secure two-party computation in the presence of malicious adversaries”; EUROCRYPT, LNCS, Springer, vol. 4515, pp. 52-78, 2007.

[11] B. Pinkas, T. Schneider, N.P. Smart, S.C. Williams, “Secure two-party computation is practical”; ASIACRYPT, LNCS, Springer, vol. 5912, pp. 250-267, 2009.

[12] M. Armbrust, A. Fox, R. Griffith, et al., “A view of cloud computing”; Communications of the ACM, vol. 53(4), pp. 50-58, 2010.

[13] T. Velte, A. Velte, R. Elsenpeter, “Cloud computing, a practical approach”; McGraw-Hill, Inc., 2009.

[14] M. Van Dijk, A. Juels, “On the impossibility of cryptography alone for privacy-preserving cloud computing”; In: HotSec. USENIX, pp. 1-8, 2010.

[15] J. Loftus, N.P. Smart, “Secure outsourced computation”; Progress in Cryptology-AFRICACRYPT 2011, Springer Berlin Heidelberg, pp. 1-20, 2011.

[16] M.J. Atallah, K.B. Frikken, “Securely outsourcing linear algebra computations”; Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security, ACM, pp. 48-59, 2010.

[17] S. Kamara, P. Mohassel, M. Raykova, “Outsourcing MultiParty Computation”; IACR Cryptology ePrint Archive, 2011.

[18] A. Peter, E. Tews, S. Katzenbeisser, “Efficiently Outsourcing Multiparty Computation under Multiple Keys”; IACR Cryptology ePrint Archive, 2013.

[19] B. Wang, M. Li, M. Chow, et al., “Computing encrypted cloud data efficiently under multiple keys”; Communications and Network Security (CNS), 2013 IEEE Conference on. IEEE, pp. 504-513, 2013.

[20] C. Gentry, “A fully homomorphic encryption scheme”. Doctoral dissertation, Stanford University.

[21] G. Asharov, A. Jain, A. López-Alt, E. Tromer, V. Vaikuntanathan, D. Wichs, “Multiparty computation with low communication, computation and interaction via threshold fhe”; In Advances in Cryptology-EUROCRYPT 2012, Springer Berlin Heidelberg, pp. 483-501, 2012.

[22] S. Halevi, Y. Lindell, B. Pinkas, “Secure computation on the web: computing without simultaneous interaction”; Advances in Cryptology-CRYPTO 2011, Springer, pp. 132-150, 2011.

[23] Z. Brakerski, V. Vaikuntanathan, “Efficient fully homomorphic encryption from (standard) LWE”; Foundations of Computer Science (FOCS), 2011 IEEE 52nd Annual Symposium on. IEEE, pp. 97-106, 2011.

[24] A. López-Alt, E. Tromer, V. Vaikuntanathan, “On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption”; Proceedings of the 44th symposium on Theory of Computing, ACM, pp. 1219-1234, 2012.

[25] Y. Sun, Q.Y. Wen, et al., “Two Cloud Servers-Assisted Secure Outsourcing Multiparty Computation”; Accepted by The Scientific World Journal, 2014.

[26] J. Hoffstein, J. Pipher, J.H. Silverman, “NTRU: A ring-based public key cryptosystem”; Algorithmic number theory. Springer Berlin Heidelberg, pp. 267-288, 1998.

SESSION

LATE BREAKING PAPERS AND POSITION PAPERS: BIG DATA ANALYTICS AND RELATED ISSUES

Chair(s)

Prof. Mary Yang
Prof. Hamid R. Arabnia

Querying Clusters of Biological Sequences

Vicente Molieri, Lina J. Karam and Zoé Lacroix
Electrical Engineering Department, Arizona State University, Tempe, Arizona 85287-5706

***Abstract*—Querying and managing scientific database are poorly supported by traditional approaches and technology. The use of an alignment tool is often the only method made available for querying biological sequences and all the sequences of a database typically need to be reprocessed entirely when the database is updated. In contrast the method presented in this paper proposes the use of two methods successfully used in image processing to query and update a database of clusters of biological sequences. The proposed method consists of efficient hash-based searching and clustering schemes. A novel metric is introduced and used to quantify the degree of matching and to determine which cluster(s) the query sequence belongs to. Evaluation results are presented to illustrate the performance of the proposed methodology.**

I. Introduction

In recent years, biomedical databases have been developed for a large variety of applications. Among them biological sequences play a fondamental role to support translational genomics. Meaningful and efficient access to biological sequence data often relies on a variety of clustering methods to address problems such as sequence classification among multiple genomes and homologue identification [1], [2], classification of sequences into related groups and automatic annotation [3], family classification [4], and family analysis [5]. When such clusters are used to organize the database, updates are often limited because they may alter the internal database organization. When new sequences are added to the database, the clusters are typically re-computed periodically. In addition, queries are also limited to the existing sequences and clusters. This paper presents a method for updating a database in real-time using clustering information. In addition to typical query and search mechanisms which can be used to explore information in an existing database, the proposed database is designed to be updated in real-time, using user-uploaded sequences to continually refine and re-define clusters. We demonstrate the approach on BIPASS,footnoteThe BIPASS database is available at http://bip.umiacs.umd.edu:8080/. a database of transcripts developed to analyze alternative splicing events [6]. In the database, each cluster corresponds to a gene and contains all transcripts at different levels of splicing. The database was developed through a timely process involving two alignment steps against a reference genome. Although the database offered users the ability to submit their transcripts, because of the limitations of the approach, BIPASS was not able to position the input transcripts in the existing database and identify the cluster(s) they would belong to. In contrast, the proposed method increases significantly the query functionality of the database by identifying among the clusters already computed the one an input sequence belongs to. In addition, the method will be used to maintain the BIPASS database.

The paper is organized as follows. An overview of related work is given in Section II. Section III presents a background of the methods used in this work. The proposed methodology is presented in Section IV. The evaluation of the proposed approach is presented in Section IV-A and conclusions are given in Section V.

II. Related Work

A clustering method captures some sort of family or grouping of data. By nature of the pair-wise comparison of biological sequences, homologues are determined by most clustering algorithms, implicitly if not explicitly. Homology in biological sequences is interesting because similar sequences could perform similar functions. PCDB is a database that is used by sequence comparison and classification tools such as PLATCOM [2] and CLASSEQ [1] and which consists of clusters of data extracted from Gen-Bank, Swiss-Prot, COG, and KEGG. The PCDB data clusters are computed using BAG [7]. This separate PCDB database is critical to the functionality of sequence comparison and classification tools because computing multiple genome comparisons is a time-intensive process and is difficult to perform in practice without a pre-computed established framework of clusters. In [2], an overview of functions available through an online interface is presented. Users can upload or paste sequence(s) and choose from a set of reference genomes to which the users' sequences will be compared to. Using the suite of tools presented in [2], users can look for probable gene fusion events, generate visualizations of genomic families, and perform clustering on a user-input sequence (relative to the reference genomes present in PCDB). In [1], a more narrowly-scoped tool is presented. CLASSEQ is an online tool proposed in [1] to compare a user sequence to a set of user-chosen reference genomes from PCDB, which can be used to determines genome relationships to the specified sequence. In [3], Z-scores from pair-wise match scoring are used to compute clusters from the UniProt database of proteins. The online interface allows users to search the database for proteins using accessions or names found in UniProt, and returns either a set of clusters to which the protein belongs or returns a list of sequences which share some sort of sequence similarity with that protein. In the SYSTERS database[4], a compilation of sequences from

the Swiss-Prot, TrEMBL, and PIR databases were clustered into protein families and super-families. The purpose of the SYSTERS database was to partition the full set of protein into groups of high similarity, and from that establishment, to allow user queries of the derived clusters. In [5], sequence similarity was used to cluster sequences from the arabidopsis and rice genomes and the resulting genome cluster database allows users to search for family and singlet proteins available in each of these two species.

III. Background

A database containing clusters of nucleotide sequences extracted from BIPASS has been developed. During the initial database generation, a file is generated consisting of a list of all sequences which are in the database and containing, for each sequence, a unique ID number as well as the sequence annotation and the sequence itself (in FASTA format). The clusters are generated through a three step process: (1) hashing, (2) matching, and (3) clustering. Hashing is, in short, a process used to convert a multi-point comparison into a single datum comparison. The hashing vector of random numbers has been stored in the database in a binary file (due to float precision, binary files are needed to store float values without loss of precision). This is the vector which is used to generate all hash IDs that are stored in the database, and which is also used to hash all future entries in the database (since generating a new random number vector would generate incomparable hash IDs). A hash ID is generated by first using a rectangular window, equal in length to the hashing vector, in order to denote a set of data points within a sequence to be hashed into a hash ID. Then, the hashing vector and the data points within the rectangular window are multiplied point-wise and the results are summed to generate a single hash ID as follows:

$$ID[j] = \sum_{i=0}^{\ell} X[i+j] * h[i] \quad (1)$$

where j is the position within a sequence, ℓ is the length of the window and the hashing vector, X is the considered sequence and h is the hashing vector. Once the hash ID is computed, the window is slid forward by one nucleobase and the process is repeated until all possible hash IDs have been computed. This operation is akin to an operation referred to as convolution in signal processing. A simple example illustrating the creation of hash IDs is given in Figure 1and the pseudocode for the algorithmic creation of hash IDs is shown in Algorithm 1. In the database, the hashes are stored in a file which contains the hash IDs (sorted by value) along with their associated sequence IDs (denoting which sequence in the database the hash ID belongs to), as well as the position within the corresponding sequence (j in (1)). Sorting hash IDs by value enables faster database searching because sorted lists are easily exploited by methodologies such as binary searching.

Matching is a process in the database generation which is used to determine relationships between sequences which are to be stored in the database. This process works by comparing

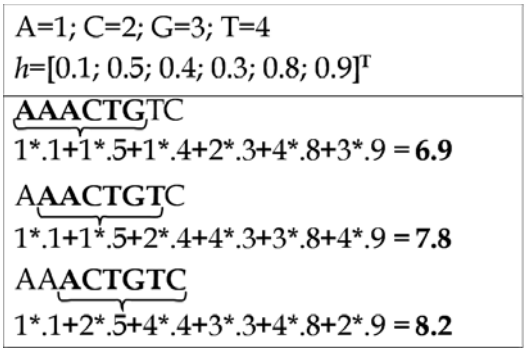


Fig. 1. Hashing

generated hash IDs to find ranges of matching characters in any two sequences. A simple example illustrating the process of matching is given in 2. This process is exhaustive and, so, when completed, all sequences within the database have links established between one another (many links are null-valued, however). In the database, this matching information is stored in a file which contains the sequence annotation followed by N integers, with N being the number of sequences in the database. Each of the integers following the sequence annotation denote the number of matching nucleobases between the sequence and the other sequences in the database. Finally, clustering takes place by examining all of the inter-sequence links found in the matching stage, and using them to determine clusters. Clusters are developed such that all sequences belong to only one cluster. These clusters are stored in another file in the database which contains a unique cluster ID for each cluster and a list of the annotations of the sequences which are in the cluster. Another file is generated containing time stamps which track the run time of the database generation and the time to complete each step.

```
Input: hasharray : vector of random numbers ∈ [0, 1]
Q : query sequence
Output: X : vector of hashid class which has elements
          hash, sequenceID, startPosition
count=0
for i=0 to |Q| do
    for j=0 to |Q[i].sequence|-|hasharray| do
        sum=0
        for k=0 to |hasharray| do
            sum+=Q[i].sequence[j+k]*hasharray[k]
        end
        X[count].hash=sum
        X[count].sequenceID=i
        X[count].startingPosition=j
        count++
    end
end
```

Algorithm 1: Hashing

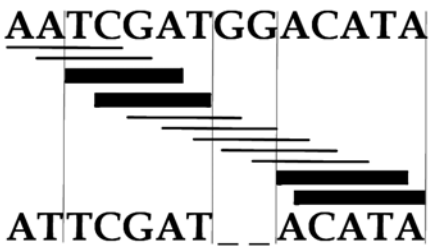


Fig. 2. Matching

IV. Methodology

The methodology in this paper aims at determining quickly the relationship between a queried set of sequences (1 or more) and the stored database of sequences which have been hashed and clustered. To achieve this goal, each queried sequence is processed in a manner similar to that which generated the database. However, since clusters have already been generated, there is no need to compute matches and clusters, but only to examine or update this existing infrastructure. This process is completed in several main steps, the number of steps required depends on the desired output of the database, but all processes can be completed using a combination of: (1) hashing, (2) searching, and (3) updating.

For all database inquiries, a FASTA format text file is the input, and the sequences in this file are processed according to the nature of the query. For a database query, a sequence must be hashed and then the darabase is searched using a matching threshold of 100% and the matching sequence, which is already in the database, is returned along with the cluster to which it belongs. For a database search, the goal is to determine the cluster(s) to which a new sequence belongs. This sort of inquiry can use the same threshold of matching as was used to generate the database, or a different one. For a database update, however, in order to maintain uniformity, the same matching threshold must be used as was used in the initial database generation. The output of this algorithm is an updated database with the new sequences either added to an existing cluster, creating a new cluster, or merging two previously unrelated clusters. The workflow described above is illustrated in Figure 3.

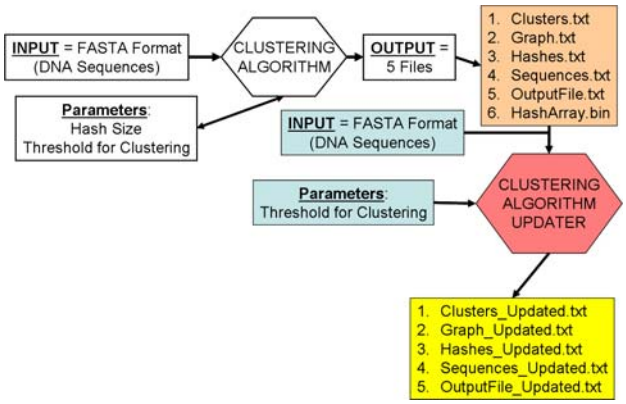


Fig. 3. Update Workflow

A. Hashing

Hashing is performed in the exact same way as was done when the database was generated. The formulation in (1) is used on all new sequence queries prior to specific processing based on the desired output. Details about the searching and updating steps are provided below.

B. Searching

Searching is the core of the methodology presented in this paper. Note that all sequences that are in the database have been hashed previously; so, their hashes can be simply extracted from the database. Each queried sequence will generate $n = l - \ell$ hash IDs using (1), where l is the length of the queried sequence and ℓ is the size of the hash window which is extracted from the database. Each of these hash IDs must be compared to the already computed hash IDs of the sequences that are in the database in order to determine if any matches exist. Using a binary search-like algorithm, each search can be computed in $O(log_2 N)$, where N is the number of hashes stored in the database. The search will return a range of indices in the database with the same hash ID, as well as information about which sequence the hash ID is related to (including the sequence ID and position within the sequence). As matches are found, tracking devices are employed to monitor how many matching hash IDs are found between each queried sequence and each sequence in the database. This portion of the searching algorithm is given in Algorithm 2.

Once all the possible matches between the new sequences and all sequences in the database are computed, the number of matching hash IDs is used to compute the number of matching nucleobases between all sequences. For a database query, if any sequence exists wherein there is 100% matching between both the queried sequence and a sequence already in the database (i.e., the query is not merely contained within a sequence already in the database), then the algorithm will return the sequence which matches, otherwise it will inform the user that no match was found. If the inquiry is to search or update the database, however, the number of matching nucleobases is divided by the minimum of the number of total nucleobases in the two sequences to compute a relation metric (RM) as given below:

$$RM = \frac{\#\, matching\, nucleobases}{min(\ell_{S1}, \ell_{S2})} \tag{2}$$

where ℓ_{S_i} is the length of a sequence S_i, i=1,2. If the resulting RM is below a given threshold, it is set to zero (denoting an insignificant link). For all links with a significant RM, the cluster to which a sequence belongs is extracted from the database and stored. For each queried sequence, a list of the clusters to which the queried sequence is related, is created. This second search algorithm is presented in Algorithm 3. If the purpose of the database inquiry is not to update and only to search, at this point the algorithm will return the cluster(s)

```
Input: X : vector of hashid class which has elements
       hash, sequenceID, startPosition – from queried
       sequences
Y : vector of hashid class which has elements hash,
sequenceID, startPosition – from database
Output: Z : vector of match class which has elements
        newsequenceID, sequenceID, totalmatches
sort X by hash value
Y is already sorted in database
for i=0 to |X| do
    if X[i].hash exists in Y.hash then
        for j=first instance of match in Y to last instance
        of match in Y do
            if |Z|==0 then
                push X[i].sequenceID, Y[j].sequenceID
                onto Z
            else
                loc=binarysearch for Y[j].sequenceID in
                Z.sequenceID
                if loc!=-1 and
                Z[loc].newsequenceID==Y[i].sequenceID
                then
                    Z[loc].totalmatches++
                else
                    push X[i].sequenceID,
                    Y[j].sequenceID onto Z
                    sort Z by sequenceID
        end
end
```

Algorithm 2: Searching

to which the queried sequence belongs to.

C. Updating

The list of related clusters for each queried sequence is examined for the update of the database. There are three cases which the queried sequence can generate. The sequence can either be found in (1) no clusters in the database, (2) one cluster in the database, or (3) multiple clusters in the database. If the sequence is not found in any clusters in the database, this means that there were not enough matching nucleobases between the queried sequence and all sequences in the current database, and so the queried sequence will be denoted by its own cluster. If the sequence is only found in one cluster (which is likely the case in a thorough database), it will be added to that cluster, and all sequences which had matching hash IDs with the queried sequence will be updated in the database to reflect that match. Meanwhile, the new sequence will have its own entry with all relevant information added. Finally, if a queried sequence is found to have significant matching in multiple clusters, this will cause all of those clusters to be merged into one. In this latter case, all sequences which had matching hash IDs with the queried sequence will be updated

```
Input: Z : vector of match class which has elements
       newsequenceID, sequenceID, totalmatches
Q : query sequence
S : vector of sequences in database
threshold : value which separates biological meaning
from lack thereof
Output: C : 2-D vector of numbers
for i = 0 to |Z| do
    if (Z[i].totalmatches/min(|S[Z[i].sequenceID]|,
    |S[Z[i].newsequenceID]|))¿threshold then
        temp=cluster to which Z[i].sequenceID belongs
        if temp not found in C[Z[i].newsequenceID] then
            push temp onto C[Z[i].sequenceID] sort
            C[Z[i].newsequenceID]
end
```

Algorithm 3: Searching Part II

in the database to reflect that match and the new sequence will have its own entry with all relevant information added. The algorithm to process this multiple-case scenario is outlined in Algorithm 4.

V. Evaluation and Discussion

Alternative splicing invalidates the theory that one gene codes for only one protein. Unlike what was formerly assumed, the rearrangement of exons from a given gene at the splicing stage may produce several mature mRNA corresponding to the production of different proteins. The first studies assumed that alternative splicing was a rare process but the systematic evaluation of splicing events on existing transcripts (thanks to access to digital libraries) shows that alternative splicing takes place more often that previously expected [8], [9]. The availability of large digital resources providing information on sequences, genes, and proteins makes it possible to provide an infrastructure to evaluate whether a gene is known to go over alternative splicing. The development of such resources requires the integration of data from a variety of resources providing sequence, gene, and protein information. Existing resources include ASPic [10], Hollywood [11], ASD [12], ProSplicer [13], ASAPII [14], ASG [15], H-DBAS [16], and BIPASS [6]. In addition to the access to transcription data contained in the database, BIPASS is the only resource that offered the ability for users to submit their sequences for clustering [17]. The limitation of the approach reluies in the inability to fully integrate the functionality as a querying mechanism in the database. Indeed, the system will produce the clusters of the input sequences but not identify the clusters of the database the input sequences belong to. The integration of the sequence querying in the database requires the ability to re-compute if necessary the clusters when a new sequence is submitted, task similar to a database update. In general, the maintenance of these resources requires the systematic mapping of transcrips to the corresponding gene and organism. While most approaches rely on the use of

```
Input: C : 2-D vector of numbers
for i = 0 to |C| do
    if |C[i]|==1 then
        Add Q[i] to cluster C[i][0] in database
        Add all hash IDs for Q[i] to hash IDs in database
        (sorted)
        Add Q[i] graph links to database
        for j=0 to |Q[i].graph| do
            if Q[i].graph[j]!=0 then
                Add symmetric link to sequence j in
                database
            else
                Add 0 link to sequence j in database
        end
    else if |C[i]|==0 then
        Add Q[i] to new cluster in database
        Add all hash IDs for Q[i] to hash IDs in database
        (sorted)
    else
        for j=1 to |C[i]| do
            Add Q[i] to cluster C[i][0] in database
            Add all hash IDs for Q[i] to hash IDs in
            database (sorted)
            Add Q[i] graph links to database
            for j=0 to |Q[i].graph| do
                if Q[i].graph[j]!=0 then
                    Add symmetric link to sequence j in
                    database
                else
                    Add 0 link to sequence j in database
                Add sequences in C[i] to C[0]
                Delete C[i]
            end
        end
end
```

Algorithm 4: Updating

a published genome to perform the mapping and clustering of transcripts, the approach proposed in this paper clusters transcripts regardless of any additional genomic information. The advantage of the method is to provide a mechanism that can be used for querying and updating the database.

We evaluate the querying approach on transcripts extracted from BIPASS [6] as follows. 17 clusters were selected randomly from the clusters retrieved with queries *kinase*, *collegenase*, and *transcription factor* from BIPASS. Clusters are usually small so in order to obtain test datasets that would correspond to a single cluster yet of a larger size (100 sequences), we created a tool that generates sequences randomly possible transcripts at various splicing stages. The tool has two options: Exons+Introns that produces intermediate mRNA and Exons only. The sequences from datasets 1-7 were were generated with this method. We consider seven datasets of various sizes and characteristics (see Table I). DB_1 and DB_2 are generated from BIPASS cluster Hs.chr17.p.7106 that corresponds to the gene Homo sapiens SRY (sex determining region Y)-box 15 (SOX15). DB_3 and DB_4 are generated from BIPASS cluster Hs.chr1.p.1323 that corresponds to the gene CD53. Each of the datasets DB_{1-4} are composed of a unique cluster. DB_5 and DB_6 are each composed of two clusters generated from Hs.chr17.p.7106 and Hs.chr1.p.1323. DB_7 is composed of sequences generated from 17 genes. DB_8 is composed of all the transcripts of the two clusters Hs.chr17.p.7106 and Hs.chr1.p.1323 of BIPASS whereas DB_9 is composed of all transcripts extracted from 17 BIPASS clusters. For each datset the number of sequences, the length of the shorter sequence, the length of the longest sequence, the average length of sequences, and their composition is reported in Table I).

TABLE I
EXPERIMENTAL DATA

	Size	Min. (bp)	Max. (bp)	Average Length	Exons or Exons+Introns
DB_1	100	152	2434	887	Exons
DB_2	100	177	10,964	2,453	Exons+Introns
DB_3	100	136	1,166	563	Exons
DB_4	100	159	1,418	831	Exons+Introns
DB_5	200	136	2,434	725	Exons
DB_6	200	159	10,964	1,642	Exons+Introns
DB_7	1,243	63	10,686	616	Exons
DB_8	16	186	1,788	1,037	Exons+Introns
DB_9	563	66	9,272	724	Exons+Introns

We evaluated four sequence queries against each dataset: Q_1: a sequence selected randomly from the database, Q_2: a sequence not included in the database, Q_3: a sub-sequence randomly selected from the database, and Q_4: a sub-sequence randomly selected from the database concatenated with a sequence not in the database. All tests were performed with a DELL Inspiron Intel CoreDuo @ 2.13 Ghz with 2 GB of RAM. The operating system was Ubuntu Desktop version 8.04 (hardy) with the Linux Kernel 2.6.24-24. In all cases Q_1 returned the cluster the query sequence belonged to, Q_2 returned no cluster, and Q_3 returned the cluster the query sequence was generated from. Q_4 returned the cluster the query sequence was generated from in most cases. The result depended of the length of the random query not in the database and the length of the sub-sequence selected from the database. We report the execution time for the query evaluation in Table II. This first implementation in C++ was developed to test the methodlogy and will be improved to exploit the hashing information more efficiently as a database index.

TABLE II
QUERY EXECUTATION TIME (IN SECONDS)

	DB_1	DB_2	DB_3	DB_4	DB_5	DB_6	DB_7	DB_8	DB_9
Q_1	41	640	8	63	49	306	1132	12	268
Q_2	23	71	14	22	39	96	226	4	119
Q_3	13	173	8	24	48	153	420	5	109
Q_4	37	245	23	47	89	253	669	9	224

VI. Conclusions and Future Work

We propose a new approach to support biological sequence clusters querying and updating. It exploits an efficient hash-based searching and clustering schemes. A novel metric is introduced and used to quantify the degree of matching and to determine which cluster(s) the query sequence belongs to. Evaluation results are presented to illustrate the performance of the proposed methodology. This first implementation will be improved to scale up to large databases such as BIPASS where it will be provided as a new functionality.

Acknowledgment

We thank Naji Mounsef for sharing his hash-based approach for sequence assembly and prototype implemented in MatLab [18]. We thank Christophe Legendre for contributing greatly to the project and Matthew Land and Ben J. Piorkowski for putting the tool to the test. We acknowledge Louiqa Raschid and Ben Snyder for the design and development of BIPASS. This research was partially supported by the National Science Foundation[1] (grants IIS 0431174, IIS 0551444, IIS 0612273, IIS 0738906, IIS 0832551, and CNS 0849980).

References

[1] K. Choi, Y. Yang, and S. Kim, "Classeq: Classification of sequences via comparative analysis of multiple genomes," in *Sixth International Conference on Machine Learning and Applications, 2007 (ICMLA 2007)*, December 2007, pp. 554–559.

[2] K. Choi, Y. Ma, J.-H. Choi, and S. Kim, "Platcom: a platform for computational comparative genomics," *Bioinformatics*, vol. 21, no. 10, pp. 2514–2516, 2005.

[3] W. Fleischmann, E. M. Zdobnov, E. V. Kriventseva, and R. Apweiler, "Clustr: a database of clusters of swiss-prot+trembl proteins," *Nucleic Acids Research*, vol. 29, no. 1, pp. 33–36, 2001.

[4] A. Krause, J. Stoye, and M. Vingron, "The systers protein sequence cluster set," *Nucleic Acids Research*, vol. 28, no. 1, pp. 270–272, 2000.

[5] K. Horan, J. Lauricha, J. Bailey-Serres, N. Raikhel, and T. Girke, "Genome cluster database: a sequence family analysis platform for arabidopsis and rice," *Plant Physiology*, vol. 138, no. 1, pp. 47–54, 2005.

[6] Z. Lacroix, C. Legendre, L. Raschid, and B. Snyder, "BIPASS: BioInformatics Pipelines Alternative Splicing Services," *Nucleic Acids Research*, July 2007, vol. Web Services Issue.

[7] S. Kim and J. Lee, "Bag: a fast program for clustering and sequencing large sets of protein or nucleotide sequences," *International Journal of Data Mining and Bioinformatics*, vol. 1, no. 2, pp. 178–200, 2006.

[8] D. Brett, H. Pospisil, J. Valcarcel, J. Reich, and P. Bork, "Alternative splicing and genome complexity," *Nat. Genet.*, vol. 30, no. 1, pp. 29–30, 2002.

[9] J. C. Venter and et al, "The sequence of the human genome," *Science*, vol. 291, no. 5507, pp. 1304–51, 2001.

[10] P. Bonizzoni, R. Rizzi, and G. Pesole, "ASPIC: a novel method to predict the exon-intron structure of a gene that is optimally compatible to a set of transcript sequences," *BMC Bioinformatics*, vol. 6, pp. 244–259, 2005.

[11] D. Holste, G. Huo, V. Tung, and C. B. Burge, "HOLLYWOOD: a comparative relational database of alternative splicing," *Nucleic Acids Res.*, vol. 34, no. Database issue, pp. D56–62, 2006.

[12] S. Stamm, J. J. Riethoven, V. Le Texier, C. Gopalakrishnan, V. Kumanduri, Y. Tang, N. L. Barbosa-Morais, and T. A. Thanaraj, "ASD: a bioinformatics resource on alternative splicing," *Nucleic Acids Res.*, vol. 34(Database issue), pp. D46–55, 2006.

[13] H. D. Huang, J. T. Horng, C. C. Lee, and B. J. Liu, "ProSplicer: a database of putative alternative splicing information derived from protein, mRNA and expressed sequence tag sequence data," *Genome Biology*, vol. 4, no. 4, p. R29, 2003.

[14] N. Kim, A. Alekseyenko, M. Roy, and C. Lee, "The ASAP II database: analysis and comparative genomics of alternative splicing in 15 animal species," *Nucleic Acids Res.*, vol. 35, no. suppl 1, pp. D93–98, 2007.

[15] J. Leipzig, P. Pevzner, and S. Heber, "The Alternative Splicing Gallery (ASG): bridging the gap between genome and transcriptome," *Nucleic Acids Res.*, vol. 32, no. 13, pp. 3977–3983, 2004.

[16] J. Takeda, Y. Suzuki, M. Nakao, T. Kuroda, S. Sugano, T. Gojobori, and T. Imanishi, "H-DBAS: alternative splicing database of completely sequenced and manually annotated full-length cDNAs based on H-Invitational," *Nucleic Acids Res.*, vol. 35, no. Database issue, pp. D104–9, 2007.

[17] Z. Lacroix and C. Legendre, "BIPASS: Design of Alternative Splicing Services," *Int. J. Computational Biology and Drug Design*, vol. 1, no. 2, pp. 200–217, 2008.

[18] L. J. Karam, Z. Lacroix, N. Mounsef, and C. Legendre, "A low-complexity probabilistic genome assembly based on hashing," in *IEEE International Workshop on Genomic Signal Processing and Statistics (GENSiPS)*, June 2008, pp. 1–4.

[1] Any opinion, finding, and conclusion or recommendation expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

MapReduce-Accelerated Framework for Identifying Minimum-Sized Influential Vertices on Large-Scale Weighted Graphs

Ying Xie[1], **Jing (Selena) He**[1], **and Vijay V. Raghavan**[2]
[1]Department of Computer Science, Kennesaw State University, Kennesaw, GA, USA
[2]The Center for Advanced Computer Studies, University of Louisiana at Lafayette, Lafayette, LA, USA

Abstract— *Weighted graphs can be used to model any data sets composed of entities and relationships. Social networks, concept networks and document networks are among the types of data that can be abstracted as weighted graphs. Identifying Minimum-sized Influential Vertices (MIV) in a weighted graph is an important task in graph mining that gains valuable commercial applications. Although different algorithms for this task have been proposed, it remains challenging for processing web-scale weighted graph. In this paper, we propose a highly scalable algorithm for identifying MIV on large-scale weighted graph using the MapReduce framework. The proposed algorithm starts with identifying an* individual zone *for every vertex in the graph using an α-cut fuzzy set. This approximation allows us to divide the whole graph into multiple subgraphs that can be processed independently. Then, for each subgraph, a MapReduce based greedy algorithm is designed to identify the minimum-sized influential vertices for the whole graph.*

Keywords: MapReduce Framework, Minimum-sized Influential Vertices, Large-Scale Weighed Graph, Big Data Analysis.

1. Introduction

Weighted graphs can be used to model any data sets composed of entities and relationships. Social networks, concept networks, and document networks are among the types of data that can be abstracted as weighted graphs. Identifying Minimum-sized Influential Vertices (MIV) in a weighted graph is an important task in graph mining that gains valuable commercial applications. Consider the following hypothetical scenario as a motivating example. A small company develops a new online application and would like to market it through an online social networks (Word-of-mouth or viral marketing differentiates itself from other marketing strategies because it is based on trust among individuals' close social circle of families, friends, and co-workers. Research shows that people trust the information obtained from their close social circle far more than the information obtained from general advertisement channels such as TV, newspaper, and online advertisements [1]). The company has a limited budget such that it can only select a small number of initial users to use it (by giving them gifts or payments). The company wishes that these initial users would like the application and start influencing their friends on the social networks to use it. And their friends would influence their friends' friends and so on, and thus through the word-of-mouth effect a large population in the social network would adopt this application. In sum, the MIV problem is whom to select as the initial users (keep the size as small as possible or under some budget) so that they eventually influence the largest number of people in the network.

The problem is first introduced for social networks by Domingos and Richardson in [2] and [3]. Subsequently, Kempe et al. [4] proved this problem to be NP-hard and propose a basic greedy algorithm that provides good approximation to the optimal solution. However, the greedy algorithm is seriously limited in efficiency because it needs to run Monte-Carlo simulation for considerably long time period to guarantee an accurate estimate. Although a number of successive efforts have been made to improve the efficiency, the state-of-the-art approaches still suffer from excessively long execution time due to the high-computational complexity for large-scale weighted graph. Furthermore, the graph structure of real-world social networks are highly irregular, making MapReduce acceleration a non-trivial task. For example, Barack Obama, the U.S. president, has more than 11 million followers in Twitter, while more than 90% of Twitter users, the follower number is under 100 [8]. Such irregularities may lead to severe performance degradation.

On the other hand, MapReduce framework has recently been widely used as a popular general-purpose computing framework and has also been shown promising potential in accelerating computation of graph problems such as breadth first searching and minimum spanning tree [5], [6], [7], due to its parallel processing capacity and ample memory bandwidth. Therefore, in this paper, we explore the use of MapReduce framework to accelerate the computation of MIV in large-scale weighted graphs.

The proposed framework starts with identifying an individual zone for every vertex in the graph. The individual zone of a given vertex is the set of vertices that the given vertex can influence. To design a scalable algorithm to address this, we approximate individual zone by using the concept of α-cut fuzzy set. This approximation allows us to reduce the complexity of multi-hop influence propagation to the level of single-hop propagation. Subsequently, we aim to find a minimum-sized set of vertices whose *influence* (the

formal definition will be presented in Section 3) reaches a pre-defined threshold. To reach this goal, a MapReduce-based greedy algorithm is designed by processing *individual zones* (the formal definition will be presented in Section 3) for all vertices.

As a summary, the contribution of this paper can be summarized as follows:

- A fuzzy propagation model was proposed to describe multi-hop influence propagation along social links in weighted social networks;
- An α-cut fuzzy set called *individual zone* was defined to approximate multi-hop influence propagation from each vertex.
- MapReduce algorithms were designed to locate each *individual zone* and then identify Minimum-sized Influential Vertices (MIV) using a greedy strategy.

This remainder of this paper is organized as follows: Section 2 reviews related literatures of the MIV problem. Network model and the formal definition of the MIV problem are given in Section 3. The MapReduced-accelerated framework are presented in Section 4 and Section 5. Finally, the work is concluded in Section 6.

2. Related Work

Finding the influential vertices and then eventually influencing most of the population in the network is first proposed by Domingos et al. in [2], [3]. They model the interaction of users as a Markov random field and provide heuristics to choose users who have large influence in network. Kempe et al. [4] formulate the problem as a discrete optimization problem and propose a greedy algorithm. However, the greedy algorithm is time-consuming. Hence, recently huge amount of researchers try to improve the greedy algorithm in two ways. One is reduce the number of individual searched in the graph. The other is improving the efficiency of calculating the influence of each individual. Leskovec et al. [9] propose an improved approach which is called CELF to reduced the number of individual searched in the graph. Later, Goyal et al. [10] propose an extension to CELF called CELF++, which can further reduce the number. Kimura et al. [11] utilize the Strong Connected Component (SCC) to improve the efficiency of the greedy algorithm.

Although many algorithms are proposed to improve the greedy algorithm, they are not efficient enough for the large scale of current social networks. Hence, some works are proposed to fit for large-scale networks. Chen et al. proposed a method called MixGreedy [12] that reduces the computational complexity by computing the marginal influence spread for each node and then selects the nodes that offers the maximum influence spread. Subsequently, Chen et al. [13] use local arborescence of the most probable influence path between two individual to further improve the efficiency of the algorithm. However, both of the algorithms provide no accuracy guarantee. In [14], Liu et al. propose ESMCE, a power-law exponent supervised Monte-Carlo method that efficiently estimates the influence spread by randomly sampling only a portion of the nodes. There have been also many other algorithm and heuristics proposed for improving the efficiency issues for large-scale social networks, such as [15], [16]. However, all of the aforementioned improvements are not effective enough to reduce execution time to an acceptable range especially for large-scale networks.

Completely different from the previous mentioned work, Liu et al. [17] present a GPU-framework to accelerate influence maximization in large-scale social networks called IMGPU, which leveraging the parallel processing capability of Graphics Processing Unit (GPU). The authors frist design a bottom-up traversal algorithm with GPU implementation to improve the existing greedy algorithm. To best fit the bottom-up algorithm with the GPU architecture, the authors further develop an adaptive K-level combination method to maximize the parallelism and reorganize the influence graph to minimize the potential divergence. Comprehensive experiments with both real-world and synthetic social network traces demonstrate that the propose IMGPU framework outperform the state-of-the-art influence maximization algorithm up to a factor of 60.

In this paper, we focus on addressing the MIV problem on large-scale weighted graphs using the MapReduce framework. The proposed method first improve the algorithm efficiency by divide the whole graph into some subgraphs using fuzzy propagation model. Subsequently, a MapReduce greedy algorithm is presented to search the best candidates in each subgraphs to achieve highly parallelism. The proposed framework show potential to scale up to extraordinarily large-scale graphs.

3. Graph Model and Problem Definition

3.1 Graph Model

We model a weighted graph by an undirected graph $G(V, E, W(E))$, where V is the set of N vertices, denoted by v_i, and $0 \leq i < N$. i is called the vertex ID of v_i. An undirected edge $e_{ij} = (v_i, v_j) \in E$ represents weights between the pair of vertices. $W(E) = \{p_{ij} \mid \text{if } (v_i, v_j) \in E, 0 < p_{ij} \leq 1, \text{ else } p_{ij} = 0\}$, where p_{ij} indicates the weights between vertices v_i and v_j. For simplicity, we assume the links are undirected (bidirectional), which means two linked vertices have the same weight (*i.e.*, p_{ij} value) on each other. Figure 1 shows an example of a weighted graph.

3.2 Problem Definition

The objective of the MIV problem is to identify a subset of influential vertices in the weighted graph. Such that, eventually large number of vertices in the graph can be influenced by these initially selected vertices. As we mentioned in Section 1, we first partition the whole graph

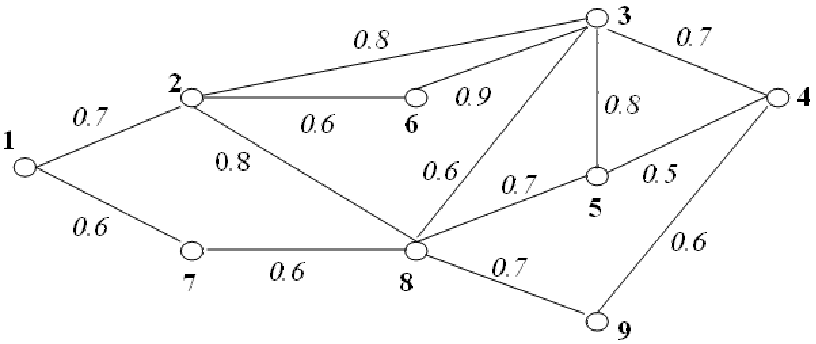


Fig. 1: A sample of a weighted graph.

Algorithm 1: Mapper Part

Method Map (vertexID id, vertexRecord: r)
Instantiate Vertex v from id and r;
$i = 0, j = 0, k = 0$;
for $i < v.ColorToTargets.size()$ **do**
  if $v.ColorToTargets[i] = G$ **then**
    for $j < v.NeighborsID.size()$ **do**
      $membershipToTarget = v.MembershipToTarges[i] \times v.MembershipOfNeighbors[j]$;
      if $membershipToTarget > \alpha$ **then**
        Instantiate Vertex vv for $Neighbors[j]$;
        $vv.ID = NeighborsID[j]$;
        $vv.NeighborsID = Null$;
        $vv.MembershipOfNeighbors = Null$;
        for $k < v.ColorToTargets.size()$ **do**
          if $i == k$ **then**
            $vv.MembershipToTargets[k] = membershipToTarget$;
            $vv.ColorToTargets[k] = G$;
            $vv.parentToTargets[k] = v.VertexID$;
          else
            $vv.MembershipToTargets[k] = 0$;
            $vv.ColorToTargets[k] = W$;
            $vv.ColorToTargets[k] = -1$;
        Create record rr from vv in the following format:
        list of adjacent vertices and weights | membership values target vertices |
        color decorations towards target vertices | immediate parent towards target vertices;
        $EMIT(vv.VertextID, rr)$;
    $v.ColorToTargets[i] = B$;
Create record r from v in the following format:
list of adjacent vertices and weights | membership values target vertices | color decorations towards target vertices |
immediate parent towards target vertices;
$EMIT(id, r)$;

Algorithm 1: Reducer Part

Method Reduce(vertexID $id, [r_1, r_2, r_3, \cdots, r_l]$)
Instantiate Vertex v from id;
$v.VertexID = id$;
$v.NeighborsID = Null$;
$v.MembersOfNeighbors = [0, 0, 0, \cdots]$;
$v.ColorToTargets = [W, W, W, \cdots]$;
$v.ParentToTargets = [-1, -1, -1, \cdots]$;
$i = 0, j = 0$;
for each r_i in $[r_1, r_2, r_3, \cdots, r_l]$ **do**
  Instantiate Vertex vv from id, and r_i;
  if $vv.NeighborsID! = Null$ **then**
    $v.NeighborsID = vv.NeighborsID$;
    $v.MembershipOfNeighbors = vv.MembershipOfNeighbors$;
  for $j < vv.ColorToTargets.size()$ **do**
    if $(vv.ColorToTargets[j] > v.ColorToTargets[j]$ **then**
      $v.ColorToTargets[j] = vv.ColorToTargets[j]$;
  for $j < vv.MembershipToTargets.size()$ **do**
    if $vv.MembershipToTargets[j] > v.MembershipToTargets[j]$ **then**
      $v.MembershipToTargets[j] = vv.MembershipToTargets[j]$;
for $i < v.ColorToTargets.size()$ **do**
  if $v.ColorToTargets[i] = G$ **then**
    Increment a predefined job counter called $numberOfIterations$;
Create record r from v in the following format:
list of adjacent vertices and weights | membership values target vertices | color decorations towards target vertices | immediate parent towards target vertices;
$EMIT(id, r)$;

into subgraphs by applying a fuzzy propagation modes. The subgraph is called *individual zone* for each vertex. Subsequent, we formally define *individual zone* as follows:

Definition 3.1: individual zone ($Zone_v$). For weighted graph $G(V, E, W(E))$, the individual zone is a fuzzy set (U_v, M_v), where, U_v is the set of vertices, and M_v is a function: $U_v -> [0, 1]$, such that for any vertex $v \in U_v$, we have

$$M_v(x) = \begin{cases} 1, & \text{if } x = v; \\ \prod_{e_{ij} \in P_{xv}} W(e_{ij}), & \text{otherwise.} \end{cases}$$

where P_{xv} is the path from vertex x to vertex v, e_{ij} is an edge in the path. We further define $W(e_{ii}) = 1$.

Another important terminology *influence* must be formally defined before the problem definition.

Definition 3.2: influence (ζ_v). For weighted graph $G(V, E, W(E))$, the influence of vertex v is denoted by ζ_v, which is the sum of the membership value of all vertices in $Zone_v$.

Now, we are ready to define the Minimum-sized Influential Vertices (MIV) problem as follows:

Definition 3.3: Minimum-sized Influential Vertices (MIV). For weighted graph $G(V, E, W(E))$, the MIV problem is to find a minimum-sized influential vertices $\chi \subseteq V$, such that $\forall v \in \chi$, $\zeta_v \geq N \times x\%$, where $x\%$ is a pre-defined threshold.

4. MapReduce Algorithm for Identifying Individual Zones

To scale the MIV problem to a large-scale weighted graph, we approximate individual zones by using α-cut fuzzy sets. That is, given a vertex v, the α-cut individual zone of v contains and only contains all vertices whose membership value towards v is greater than or equal to the give parameter α. For simplicity, in the description of the MapReduce algorithms shown in Section 4 and 5, *individual zone* actually means *α-cut individual zone*.

A given weighted graph will be represented by using adjacency lists, a similar representation used in MapReduce based algorithms for breath first searching and minimum spanning tree [5], [6], [7]. For instance, the weight graph shown in Figure 1 is described as follows:

1) 2(0.7), 7(0.6)
2) 1(0.7), 3(0.8), 6(0.6), 8(0.8)
3) 2(0.8), 4(0.7), 5(0.8), 8(0.6), 6(0.9)
4) 3(0.7), 5(0.5), 9(0.6)
5) 3(0.8), 4(0.7), 8(0.7)
6) 2(0.6), 3(0.9)
7) 1(0.6), 8(0.6)
8) 2(0.8), 3(0.6), 5(0.7), 7(0.6), 9(0.7)
9) 8(0.7), 4(0.6)

For a large-scale weighted graph, we divide its adjacency lists into k equal-sized files, where $k = total - size/block -$

$size$ ($total - size$ is the total size of the adjacency lists of the graph data, and $block - size$ is the block size of the HDFS of the Hadoop cluster). Assume there are n vertices for each of the k files. Then we select m vertices from each file respectively to form a set of target vertices for which we identify *individual zones*. In other words, the execution of the following MapReduce algorithm is able to identify individual zones for $k \times m$ target vertices. Therefore, we just need to run n/m times of this algorithm in order to identify individual zones for all vertices. Fortunately, the n/m times of executing this algorithm is totally independent to each other, thus can run in a completely parallel manner.

As an illustration, we assume $k = 2$ and $n = 4$ for the weighted graph given in Figure 1. Then, let $m = 2$, *i.e.*, for each of the 2 files, we select 2 vertices as the target vertices. Assume for the first run of the algorithm, we select vertices 1 and 2 from File 1, and vertices 5 and 6 from File 2. Then we will have the following two input files to identify individual zones for vertices $1, 2, 5,$ and 6.

File 1:

1) 2(0.7), 7(0.6) | 1, 0, 0, 0 | G, W, W, W | 0, -1, -1, -1
2) 1(0.7), 3(0.8), 6(0.6), 8(0.8) | 0, 1, 0, 0 | W, G, W, W | -1, 0, -1, -1
3) 2(0.8), 4(0.7), 5(0.8), 8(0.6), 6(0.9) | 0, 0, 0, 0 | W, W, W, W | -1, -1, -1, -1
4) 3(0.7), 5(0.5), 9(0.6) | 0, 0, 0, 0 | W, W, W, W | -1, -1, -1, -1

File 2:

5) 3(0.8), 4(0.7), 8(0.7) | 0,0,1,0 | W,W,G,W | -1, -1, 0, -1
6) 2(0.6), 3(0.9) | 0,0,0,1 | W,W,W,G | -1, -1, -1, 0
7) 1(0.6), 8(0.6) | 0,0,0,0 | W,W,W,W | -1, -1, -1, -1
8) 7(0.6), 2(0.8), 3(0.6), 5(0.7), 9(0.7) | 0,0,0,0 | W,W,W,W | -1, -1, -1, -1
9) 8(0.7), 4(0.6) | 0,0,0,0 | W,W,W,W | -1, -1, -1, -1

In the two input files shown above, each vertex is represented in the following format: *VertexID list of adjacent vertices and weights |membership values target vertices |color decorations towards target vertices |immediate parent towards target vertices*.

Taking vertex 1 as an example, its membership values to the four target vertices (vertex 1, 2, 5, and 6) are initialized to be 1, 0, 0, 0, respectively. Its color decoration is set to be Gray (G), White (W), White (W), White (W) respectively, where the first G means that more vertices belonging to the individual zone of the first target vertex need to be located starting from this vertex; the rest W means that, for other target vertices, no immediate action is needed from this vertex; the other possible color value is Black (B), which means no further development from the current vertex is needed for the corresponding target vertex. We further assume the ordinal among the three color values are $B > G > W$. Its immediate parent vertices towards each of the target vertices are initialized to be 0, -1, -1, and -1, respectively, where the first 0 means that this vertex itself is the first target vertex; and the rest -1 mean that its parent vertex to the rest of the target vertices remains unknown for right now.

We further use the following data structure Vertex to hold the information on each individual vertex.

- ID: int
- NeighborsID: $List < Integer >$
- MembershipOfNeighbors: $List < Double >$
- MembershipToTargets: $Array < Double >$
- ColorToTargets: $Array < Char >$
- ParentToTargets: $Array < Integer >$

Then, the MapReduce Algorithm can be described in Algorithm 1.

When this MapReduce job is executed on input File 1 and File 2, it will generate the following output , if $\alpha = 0.5$:

1) 2(0.7), 7(0.6) | 1, 0.7, 0, 0 | B, G, W, W | 0, 2, -1, -1
2) 1(0.7), 3(0.8), 6(0.6), 8(0.8) | 0.7, 1, 0, 0.6 | G, B, W, G | 1, 0, -1, 6
3) 2(0.8), 4(0.7), 5(0.8), 8(0.6), 6(0.9) | 0, 0.8, 0.8, 0.9 | W, G, G, G | -1, 2, 5, 6
4) 3(0.7), 5(0.5), 9(0.6) | 0, 0, 0.7, 0 | W, W, G, W | -1, -1, 5, -1
5) 3(0.8), 4(0.7), 8(0.7) | 0,0,1,0 | W,W,B,W | -1, -1, 0, -1
6) 2(0.6), 3(0.9) | 0,0.6,0,1 | W,G,W,B | -1, 2, -1, 0
7) 1(0.6), 8(0.6)| 0.6,0,0,0 | G,W,W,W | 1, -1, -1, -1
8) 7(0.6), 2(0.8), 3(0.6), 5(0.7), 9(0.7) | 0,0.8,0.7,0 | W,G,G,W | -1, 2, 5, -1
9) 8(0.7), 4(0.6) | 0,0,0,0 | W,W,W,W | -1, -1, -1, -1

Since the output records contains G color, the job counter numberOfIterations will be greater than 0. So we run above MapReuce job for another iteration by using the output of the first run as input. This process will continue until no record in output contains any G color. Then the output contains information on individual zones for the target vertices 1, 2, 5, and 6. In the same way, we are able to obtain individual zones for vertices 3, 4, 7, 8, and 9.

5. MapReduce Algorithm for solving MIV

Using the graph shown in Figure 1 as an illustration again, the output of Algorithm on this graph can be easily converted to the following format by a MapReduce process. Each record represents the α-cut individual zone for a vertex. Given a vertex, its influence is the sum of all the membership values in its α-cut individual zone. For example, the influence of vertex 1 is $0.7 + 0.6 + 0.56 + 0.56 = 2.42$. Now, the task is to find a minimum-sized influential vertices whose influence reaches $N \times x\%$.

1) 2(0.7), 7(0.6), 8(0.56), 3(0.56)
2) 1(0.7), 8(0.8), 6(0.6), 3(0.8), 4(0.56), 5(0.64), 9(0.56)
3) 2(0.8), 6(0.9), 8(0.64), 5(0.8), 4(0.7), 1(0.56)
4) 3(0.7), 5(0.5), 9(0.6), 2(0.56), 6(0.63)
5) 8(0.7), 3(0.8), 4(0.56), 2(0.64), 6(0.72)
6) 2(0.6), 3(0.9),8(0.54),5(0.72), 4(0.63)
7) 1(0.6), 8(0.6)
8) 7(0.6), 2(0.8), 3(0.6), 5(0.7), 9(0.7), 1(0.56), 6(0.54)
9) 8(0.7), 4(0.6)

We design a MapReduce based greedy algorithm for computing this task. Let the minimum-sized set of influential vertices be denoted as S. We also introduce another set denoted as I, which includes all the vertices that are influenced by vertices in S as well as their maximum membership values towards all influential vertices in S. For instance, for the graph shown in Figure 1, if $S = \{2, 8\}$, then $I = \{2(1), 8(1), 1(0.7), 6(0.6), 3(0.8), 4(0.56), 5(0.7), 9(0.7), 7(0.6)\}$, and the influence of S is $1 + 1 + 0.7 + 0.6 + 0.8 + 0.56 + 0.7 + 0.7 + 0.6 = 6.66$.

A MapReduce based greedy algorithm for identifying minimum-sized influential vertices can be described in Algorithm 2:

Algorithm 2: Mapper Part

Method Map (vertexID id, vertexRecord: r)
$I_{temp} = I$;
if id is in I_{temp} **then**
  reset its membership value in I_{temp} to be 1;
else
  add $id(1)$ to I_{temp};
for each vertex v_i in r **do**
  if v_i is in set I_{temp} **then**
    let its membership value recorded in I_{temp} be mm;
    if $m > mm$ **then**
      replace mm with m in I_{temp} for v_i;
  else
    add $v_i(m)$ to I_{temp};
$sumInfluence =$ sum of all membership values in I_{temp};
$Emit(0, id \mid Itemp \mid sumInfluence)$;

Algorithm 2: Reducer Part

Method Reduce(Key $k, [I_{temp\_0} \mid sumInfluence_{0,\cdots,id_i} \mid I_{temp\_i} \mid sumInfluence_{i,\cdots}])$
$max = 0, id_{max} = null, I_{max} = null$;
for each $id_i \mid I_{temp_i} \mid sumInfluence_i$ **do**
  if $sumInfluencei > max$ **then**
    $max = sumInfluence_i$;
    $id_{max} = id_i$;
    $I_{max} = I_{temp_i}$;
Add id_{max} to the set S;
$I = I_{max}$;
if $max < N \times x\%$ **then**
  Increment a predefined job counter called $numberOfIterations$;

If the job counter numberOfIterations is greater than 0, then the above MapReduce algorithm will run again to add the next most influential vertex to S.

6. Conclusion

In this paper, we propose a fuzzy propagation model to simplify the description of how a vertex influences others in a large-scale weighted social network. A MapReduce algorithm was then designed to locate individual zone for each vertex of the network. The concept of individual zone approximates the influence propagated from a vertex by using an α-cut fuzzy set. Then, a MapReduce-based greedy algorithm was designed to identify MIV from all individual zones.

Acknowledgement

This research is partly supported by the Kennesaw State University College of Science and Mathematics Interdisciplinary Research Opportunities (IDROP) program.

References

[1] Nail, J. 'The Consumer Advertising', *Forrester Research and Intelliseek Market Research Report*, 2004.
[2] Domingos, P., and Richardson, M. 'Mining the Network Value of Customers', *SIGKDD*. pp. 57-66, 2001.
[3] Richardson, M, and Domingos, P. 'Mining Knowlege-Sharing Sites for Viral Marketing', *SIGKDD*. pp. 61-70, 2002.
[4] Kempe, D., Kleinberg, J., and Tardos, E. 'Maximizing the Spread of Influence through a Social Network', *SIGKDD*. pp. 137-146, 2003.
[5] Goodrich, M.T. 'Simulating parallel algorithms in the MapReduce framework with applications to parallel computational geometry', *CORR*, 2010.
[6] Lattanzi, S., Moseley, B., Suri, S., and Vassilvitskii, S. 'Filtering: a method for solving graph problems in MapReduce', *SPAA*. pp. 85-94, 2011.
[7] White, T. 'Hadoop: The Definitive Guide (3rd ed.)', *O'Reilly*, 2012.
[8] Kwak, H., Lee, C., Park, H., and Moon, S. 'What is Twitter, a Social Network or a News Media?', *WWW*. pp. 591-600, 2010.
[9] Leskovec, J., Krause, A., Guestrin, C., Faloutsos, C., VanBriesen, J., and Glance, N. 'Cost-effective Outbreak Detection in Networks', *SIGKDD*. pp. 420-429, 2007.
[10] Goyal, A., Lu, W., and Lakshmanan, L.V. 'Celf++: Optimizing the Greedy Algorithm for Influence Maximization in Social Networks', *WWW*. pp. 47-58, 2011.
[11] Kimura, M., Saito, K., and Nakano, R. 'Extracting Influential Nodes for Information Diffusion on a Social Network', *NCAI*. 22: 1371-1380, 2007.
[12] Chen, W., Wang, Y. and Yang, S. 'Efficient Influence Maximization in Social Networks', *SIGKDD*, 2009.
[13] Chen, W, Wang, C., and Wang, Y. 'Scalable Influence Maximization for Prevalent Viral Marketing in Large0Scale Social Networks', *SIGKDD*, 2010.
[14] Liu, X., Li, S., Liao, X., Wang, L., and Wu, Q. 'In-Time Estimation for Influence Maximization in Large-Scale Social Networks', *ACM Proceedings ib EuroSys Workshop Social Network Systems*, 2012. pp. 1-6.
[15] Hu, J., Meng, K., Chen, X., Lin, C., and Huang, J. 'Analysis of Influence Maximization in Large-Scale Social Networks', *ACM Sigmetrics Big Data Analytics Workshop*, 2013.
[16] Kim, J., Kim, S.K., and Yu, H. 'Scalable and Parallelizable Processing of Influence Maximization for Large-Scale Social Networks', *ICDS*, 2013.
[17] Liu, X, Li, M., Li, S., Penng, S., Liao, X., and Lu, X. 'IMGPU: GPU-Accelerated Influence Maximization in Large-Scale Social Networks', *Transcations on Parallel and Distributed Systems*, 25(1):136-145, 2014.

Lessons Learned: Building a Big Data Research and Education Infrastructure

G. Hsieh, R. Sye, S. Vincent and W. Hendricks

Department of Computer Science, Norfolk State University, Norfolk, Virginia, USA
[ghsieh, wthendricks]@nsu.edu, [r.sye, s.m.vincent]@spartans.nsu.edu

Abstract – *Big data is an emerging technology which has been growing and evolving rapidly in related research, development, and applications. It is used by major corporations and government agencies to store, process, and analyze huge volumes and variety of data. The open source Apache Hadoop platform and related tools are becoming the de facto industry standard for big data solutions. It is a very powerful, flexible, efficient, and feature-rich framework for reliable, scalable, distributed computation using clusters of commodity hardware. On the other hand, Hadoop and its ecosystem are very complex and they change rapidly with new releases, features, packages, and tools. They are also relatively new, and thus lack adequate documentation and broad user experience base that come with mature products. Hence, it is very challenging to learn, install, configure, operate, and manage Hadoop systems, especially for smaller organizations and educational institutions without plenty of resources. In this paper, we describe our experiences and lessons learned in our efforts to build up a big data infrastructure and knowledge base in our university during the past nine months, using a variety of environments and resources, along with an incremental and iterative learning and implementation approach. In addition, we discuss the plan being implemented to enhance the infrastructure to provide enterprise-class capabilities by the end of 2014.*

Keywords: *big data, Hadoop, lab, learning.*

1 Introduction

In a recent report by the National Institute of Standards and Technology Big Data Public Working Group [1], "Big Data refers to the inability of traditional data architectures to efficiently handle new data sets."

"***Big Data*** *consists of extensive datasets, primarily in the characteristics of volume, velocity, and/or variety that require a scalable architecture for efficient storage, manipulation, and analysis.*"

Big data is an emerging technology which has been growing and evolving rapidly in related research, development, and applications. It is used by major corporations (e.g., Google, Facebook, and Amazon) and government agencies (e.g., Department of Defense) to store, process, and analyze huge volumes and variety of data to help make better decisions, improve situational awareness, grow customer base, and gain strategic advantage. In a recent forecast published in Dec. 2013 [2], International Data Corporation (IDC) "expects the Big Data technology and services market to grow at a 27% compound annual growth rate through 2017 to reach $32.4 billion."

The open source Apache Hadoop platform and related tools [3] are becoming the de facto industry standard for big data solutions. It is a very powerful, flexible, efficient, and feature-rich framework for reliable, scalable, distributed computation using clusters of commodity hardware. On the other hand, Hadoop and its ecosystem are very complex and they change rapidly with new releases, features, packages, tools, and even modified API's distributed in a very fast pace. They are also relatively new, with a major portion in beta or production releases within the past year or two. Thus, they lack adequate documentation and broad user experience base that come with mature products. Overall it is very challenging to learn, install, configure, and operate Hadoop systems, especially for smaller organizations and educational institutions without plenty of resources.

Recognizing the importance of big data, a new research effort was launched at Norfolk State University (NSU) in August 2013, with Raymond Sye and Shontae Vincent, both M.S. students in the Department of Computer Science, conducting research in this subject area for their M.S. Thesis/Project under the supervision of Dr. George Hsieh, a professor in the department. The main objectives of this coordinated research effort were:

(a) Learn the fundamentals of Hadoop architecture, processing model, and key technological components.
(b) Install and configure small-scale Hadoop clusters in the lab to gain hands-on knowledge, skills and experiences.
(c) Apply the acquired knowledge and skills, and use the established lab infrastructure to perform graph-based computations.

1.1 Phased Approach

To accomplish these objectives, an incremental and iterative approach was used to tackle the complexity and challenges discussed earlier.

The main activities for this research effort can be grouped into six major steps in a roughly sequential order:

(1) Get started with Hadoop using Hortonworks Sandbox [4] and its interactive tutorials in a single-node virtual machine configuration.
(2) Install and configure a multi-node Hadoop cluster in virtual machines, using Hortonworks Data Platform (HDP) [5] and Ambari cluster management tool [6] .
(3) Install and configure a five-node Hadoop cluster on commodity hardware, using HDP and Ambari.
(4) Get started with Hadoop application development, using Cloudera QuickStart VM [7], in a single-node virtual machine configuration.
(5) Install and configure a seven-node Hadoop cluster on commodity hardware, using Cloudera's CDH [8] and Cloudera Manager [9].
(6) Develop a Hadoop graph-based application, using the Cloudera based, multi-node Hadoop cluster.

Note that we chose Hortonworks and Cloudera, which are among the top commercial vendors that provide customized Hadoop software distributions based on the common Hadoop code managed by Apache Software Foundation. These vendors also develop and supply additional tools and capabilities beyond the common Hadoop code base, such as Cloudera Manager, to simplify and automate the installation, configuration, and administration of Hadoop systems.

We also chose the open-source CentOS Linux [10] as the base operating system for all of our Hadoop systems, virtual and physical, primarily because of its ease of use, enterprise-class features, and security enhancements.

1.2 Infrastructure Enhancement

In February 2014, Norfolk State University was awarded an equipment grant entitled "Building a Cloud Computing and Big Data Infrastructure for Cybersecurity Research and Education" by the U.S. Army Research Office. The funds from this grant will allow NSU to significantly enhance its big data research and education infrastructure by bringing in enterprise-class capabilities in computing, storage, and networking.

The knowledge, skills, and experiences accumulated through the past year are extremely useful for planning and designing this new phase of infrastructure expansion. To date, all necessary hardware, software, and facilities for the planned expansion have been selected, designed, and ordered. We plan to stand up the expanded infrastructure around the fourth quarter of 2014.

1.3 Outline

The remainder of the paper is organized as follows. In Section 2 we provide an overview of the major steps used in our phased approach. In Section 3 we describe our planned expansion of the infrastructure in more detail. In Section 4 we conclude the paper with a summary and some how-to recommendations for building up a big data research and education infrastructure in a timely and cost-effective manner without requiring significant upfront investments in people and resources.

2 Initial Infrastructure

In this section, we discuss the six steps used to build up our initial big data research and education infrastructure from both human expertise and system resources perspectives, during the past nine months from September 2013 to May 2014.

2.1 Getting Started with Hadoop

Given the complexity and rapid changing pace of Hadoop and its ecosystem, it was truly challenging to figure out an effective way of getting started with Hadoop without relying on professional services or staff.

After a relatively short period of investigation and experimentation, we chose Hortonworks Sandbox as the preferred platform for the "getting-started" training on Hadoop for ourselves and five additional members who joined the research team later.

According to Hortonworks, "Sandbox is a personal, portable Hadoop environment that comes with a dozen interactive Hadoop tutorials. Sandbox includes many of the most exciting developments from the latest HDP distribution, packaged up in a virtual environment that you can get up and running in 15 minutes!" [4]

We started with Version 1.3 of Sandbox and then migrated to Version 2.0 when the newer version became available. Sandbox is provided as a self-contained virtual machine for three different virtualization environments: (a) Oracle VirtualBox; (b) VMware Fusion or Player; and (c) Microsoft Hyber-V. We tried to use Sandbox with both VirtualBox and VMware environments, and found that Sandbox worked better with VirtualBox which is also the recommended virtualization environment for Sandbox.

After downloading the Sandbox VM image, the next step was to import the appliance into VirtualBox. This step was very straightforward for people with basic knowledge and experience in VirtualBox.

Once the Sandbox VM is started, a user can initiate a Sandbox session by opening a web browser and entering a pre-configured IP address (e.g., http://127.0.0.0:8888/). Once connected to the web server running locally, a user can learn Hadoop on Sandbox by following a dozen or more hands-on tutorials.

We found Hortonworks Sandbox to be a very effective learning environment for "getting-started" with Hadoop. Sandbox's integrated, interactive, and easy-to-use tutorial environment enables a user to focus on the key concepts for the tasks on hand, without having to learn the detailed mechanics behind the scene immediately. It also provides a rich set of video, graphical, and text based instructions along with informative feedback during exercises and suggestions for corrective actions when errors occurred.

Running Sandbox for learning Hadoop does not require a great deal of hardware resources. It runs well on commodity 64-bit systems with virtualization technology hardware support and a minimum of 4 GB RAM. Note that some Intel 64-bit processors do not provide virtualization

technology hardware support, and Sandbox will fail to run on these systems.

2.2 Multi-Node HDP Cluster in VMs

After completing the tutorials provided by Sandbox and having gained the basic knowledge about Hadoop, we proceeded to learn and experiment with installing and configuring a multi-node Hadoop cluster using Hortonworks Data Platform running in multiple virtual machines on the same physical host system. This step was designed to leverage our familiarity with Hortonworks Sandbox gained earlier while tackling the more challenging task of setting up a multi-node Hadoop cluster.

We chose to install and configure the HDP 2.0 based Hadoop cluster using the Apache Ambari Install Wizard [11]. Ambari provides an easy-to-use graphical user interface for users to deploy and operate a complete Hadoop cluster, manage configuration changes, monitor services, and create alerts for all the nodes in the cluster from a central point, the Ambari server.

The first step in this implementation was to layout a design for the Hadoop cluster including:

(a) The number of hosts: 6.
(b) The types of hosts - Ambari Server: 1; Masters: 2; Slaves: 2; and Client: 1.
(c) FQDN and IP address for each host (without using DNS).

The second step was to create six VMs each loaded with CentOS 6.4 (or newer). Then configure each VM to set up the appropriate hostname, IP address, host file, password-less SSH, and other prerequisite software (e.g., JDK).

The third step was to install Apache Ambari on the host designated as Ambari Server. Once the Ambari service was started, the next step was to access the Ambari Install Wizard through a web browser to complete the installation, configuration and deployment of this Hadoop cluster.

We found the Apache Ambari to be a very easy-to-use tool for installing, configuring, and deploying a Hadoop cluster, as it automates many of the underlying tasks for the user who only needs to supply high-level information such as the hosts, their roles (manager, master, slave, or client), and the services to be assigned to the hosts.

Ambari allocates these services to the appropriate hosts for load balancing and reliability concerns, and then proceeds to install, configure, start and test the appropriate software on these hosts automatically.

For this exercise, we used commodity Windows based PC's with moderate processing power (e.g., 64-bit CPU with 2 to 4 cores, and 8 to 16 GB RAM). Installation using Ambari did run into capacity related problems from time to time, especially when the Ambari server was downloading and installing software to all targeted hosts simultaneously. Some of the underlying tasks could fail and thus cause the installation to fail. One side effect of this failure was that the rpm database often got corrupted. Rebuilding the rpm database often resolved this kind of problem and allowed the installation to proceed (at least incrementally until the next failure occurred).

2.3 Multi-Node HDP Cluster

The experiences gained in designing, installing, configuring, and operating the six-node HDP based Hadoop cluster in a virtual environment were very useful for our next step of setting up a multi-node HDP cluster using multiple physical hosts.

As shown in Table 1, four commodity systems were used for this Hadoop cluster based on HDP and managed by Apache Ambari.

Table 1. Multi-node HDP cluster

Hosts	Hostname/ local IP	CPU	RAM	Disk
Ambari Server	HDPcs2AMBARI 199.111.112.169	Intel Xeon (4C) 3GHz	8 GB	500 GB
Master Node	HDPcs2MASTER 199.111.112.171	Intel Core 2 Duo 3GHz	4 GB	250 GB
Data Node 1	HDPcs2DN1 199.111.112.180	Intel Core 2 Duo 3GHz	4 GB	250 GB
Data Node 2	HDPcs2DN2 199.111.112.189	Intel Xeon (4C) 3.2GHz	12 GB	900 GB

Monitoring and managing a large scale distributed Hadoop cluster is a non-trivial task. To help the users deal with the complexity, Ambari collects a wide range of information from the nodes and services and presents them in an easy-to-use dashboard, as shown in Figure 1. Ambari also allows users to perform basic management tasks such as starting and stopping services, adding hosts to a cluster, and updating service configuration.

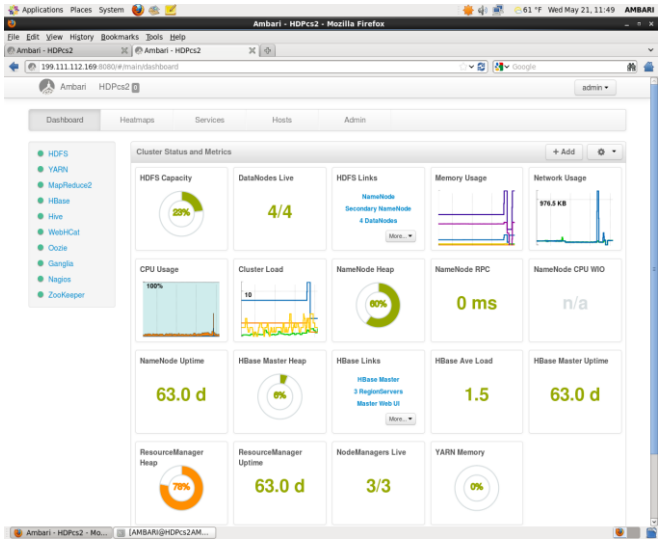


Figure 1. Ambari Dashboard display

2.4 Cloudera QuickStart VM

Cloudera has been considered the market leader among pure play Hadoop vendors that provide Hadoop related software and services. It also builds proprietary products on top of open source Hadoop code with an "open source plus proprietary model".

On such product is Cloudera Manager [9] which is included in Cloudera Express and Cloudera Enterprise. With Cloudera Express, which is a free download, users can easily deploy, manage, monitor and perform diagnostics on Hadoop clusters. Cloudera Enterprise, which requires an annual subscription fee, includes all of these capabilities plus advanced management features and support that are critical for operating Hadoop and other processing engines and governance tools in enterprise environments.

Given Cloudera's market leadership position and the potential benefits of its proprietary products, we simply did not want to ignore it.

The easiest way to get started with Cloudera's products was to use its QuickStart VM [12] which contains a single-node Apache Hadoop cluster including Cloudera Manager, example data, queries, and scripts. The VM is available in VMware, VirtualBox and KVM flavors, and all require a 64 bit host OS. This VM runs CentOS 6.2. We used primarily the CDH 4.4 and CDH 4.6 versions of the QuickStart VM.

Cloudera QuickStart VM did not provide an integrated tutorial environment or a collection of tutorials that were as easy to use as those provided by Hortonworks Sandbox. On the other hand, it provided all the commonly used Hadoop platform and tools. Thus, users did not need to download, install, and configure these packages individually.

In addition, Cloudera QuickStart VM included many of the commonly used software development tools (e.g., Eclipse and JDK) which made it a more suitable platform for developing Hadoop applications than Hortonworks Sandbox.

Getting started with developing Hadoop applications, beyond the simple "Hello World" type of tutorial app, can be quite challenging. Many tasks require executing Linux shell scripts with long lists of command line arguments. In addition, the binaries, shell scripts, and configuration files can be in different locations, depending on how the Hadoop system is installed and configured and which Hadoop distribution is used. Furthermore, the user accounts can be set up differently. All these factors make it challenging to get started with Hadoop application development, as the user needs to first gain a good understanding of the lay of the land so (s)he can navigate around these issues. The user also needs to have a sufficient level of proficiency in working with Linux OS and prior software development experiences in general.

To gain the basic knowledge and skills in Hadoop application development, we used primarily two books as resources. The first book entitled "Hadoop Beginner's Guide" [13], by Gary Turkington and published in February 2013, provided a very useful introduction to Hadoop application development with clear description and good example code. It was also not too difficult to get started with running the example code, as we used the Cloudera QuickStart VM as the platform which already contained the vast majority of the Hadoop software and prerequisite software development tools. Furthermore, the example code, although written with the older versions of Hadoop software and tools at the time of publication, worked well with the newer versions bundled with Cloudera QuickStart VM.

Another book we used for learning Hadoop application development was entitled "Hadoop in Practice" [14] by Alex Holmes and published in October 2012. The Appendix A contained background information on all the related Hadoop technologies in the book. Also included were instructions on how to build, install, and configure related projects.

To set up an environment as specified in the appendix, we started with creating a virtual machine loaded with CentOS 6.4 (Software Development Workstation option). We next installed the Hadoop base using CDH 3.0 distribution and configure our Hadoop system for the pseudo-distributed mode. We then installed and configured the remaining nineteen packages manually and individually. These packages included MySQL, Hive, Pig, Flume, HBase, Sqoop, Oozie, R, etc.

It was very challenging to go through all the steps to install and configure this target Hadoop system using primarily manual procedures and separate packages one at a time. Many of these challenges could be alleviated by using cluster provisioning and management tools such as Apache Ambari and Cloudera Manager.

Nonetheless, going through this process helped us to gain much deeper understanding and appreciation of the interdependencies and intricacies involved in getting all these packages installed and configured correctly so they can function together. This kind of knowledge and skills are important for troubleshooting problems and customizing installations, configurations, and operations, even with the cluster management tools available. Some of the important lessons learned include:

(a) Installing a Linux OS option pre-packaged with software development tools can save a lot of time and effort, as numerous extra packages are generally required to be downloaded, installed, configured, and even built on demand.

(b) The installed directories for the same software could be different, depending on the installation procedures and instructions. For example, installing from tarballs versus installing via rpm/yum could install the same software in different directories. So it is important to recognize this potential difference and make plans or adjustments accordingly.

(c) Make sure all the required environmental variables (e.g., PATH), and profiles are set up correctly. It is useful to have them set up consistently across user accounts and across hosts. Some Hadoop packages require specific global environmental variables to be defined in their specific configuration files.

(d) There could be many hard and symbolic (soft) links in the file system allowing multiple filenames (or directories) to be associated with a single file (or directory). It is important to understand these links to make sure that the correct files (or directories) are updated and links are not broken accidentally.

(e) Similarly, it is important to understand the Linux alternatives system which uses symbolic links extensively to manage the alternatives for a given generic name. For example, several different Java packages and JDK's may be installed on the same system. Activating the specific packages may require rearranging the alternatives (in their preferences).

2.5 Multi-Node CDH Cluster

The next step was to set up a multi-node Hadoop cluster using Cloudera distribution while taking advantage of the capabilities provided by Cloudera Manager.

For this exercise, we installed and configured a seven-node cluster, one as the Manager, two as masters, and four as slaves. The Manager node has two Ethernet connections, one to Internet and the other to an internal network for the Hadoop cluster. All remaining nodes are connected only to the internal network physically. The Manager node also performs IP forwarding for the remaining nodes so they can access the Internet indirectly through the Manager node. Figure 2 shows the connectivity among the nodes.

Again, the nodes were implemented using commodity machines all running CentOS 6.4 (Software Development Workstation option). The Cloudera software deployed was based on Cloudera Express 5.0.0-beta-2 release which contained Hadoop Version 2.2.0. Also installed was Hue Version 3.5.0 which is an open-source Web interface that supports Hadoop and its ecosystem. Hue provides a Web application interface for Apache Hadoop. It supports a file browser, JobTracker interface, Hive, Pig, Oozie, HBase, and more. Table 2 shows the hardware configurations for the Cloudera based Hadoop cluster.

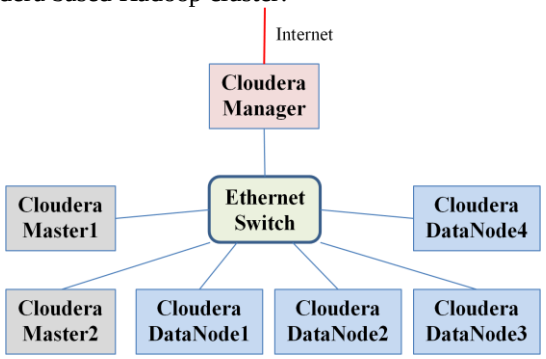


Figure 2. Multi-node Cloudera cluster

Table 2. Hardware configuration for CDH cluster

Hosts	Hostname/ local IP	CPU	RAM	Disk
Manager	CDHcs1mgr 192.168.48.1	Intel Xeon (4C) 3GHz	8 GB	500 GB
Master1	CDHcs1MN1 192.168.48.10	Intel Xeon (4C) 2.5GHz	8 GB	150 GB
Master2	CDHcs1MN1 192.168.48.2	Intel Xeon (4C) 2.5GHz	8 GB	150 GB
DataNode1	CDHcs1DN1 192.168.48.11	Intel Xeon (4C) 2.5GHz	8 GB	150 GB
DataNode2	CDHcs1DN2 192.168.48.12	Intel Xeon (4C) 3.2GHz	8 GB	150 GB
DataNode3	CDHcs1DN3 192.168.48.13	Intel Core 2 Duo 3GHz	4 GB	250 GB
DataNode4	CDHcs1DN4 192.168.48.14	Intel Core 2 Duo 3GHz	4 GB	250 GB

We chose to use the Cloudera Express 5.0.0-beta-2 release, because a decision had been made around that time to deploy Cloudera distribution for the new equipment being acquired for our infrastructure enhancement effort. Thus, we wanted to become familiar with the Cloudera 5.0 release, even when it was still in beta stage, so we would be prepared to work with it when the new equipment is deployed. As a result, we had to work with the beta version of the Cloudera Manager Installation Guide which did not contain as much information as the most recent Version (5.0.1) of the guide [15] published on May 28, 2014.

Although Cloudera Manager provided an automated installation option, "This path is recommended for demonstration and proof of concept deployments, but is not recommended for production deployments because it's not intended to scale and may require database migration as your cluster grows." [15].

Based on this recommendation, we chose to follow the Installation Path B – Manual Installation Using Cloudera Manager Packages. This path required a user to first manually install and configure a production database for the Cloudera Manager Server and Hive Metastore. Next, the user needed to manually install the Oracle JDK and Cloudera Manager Server packages on the Cloudera Manager Server host. To install Oracle JDK, Cloudera Manager Agent, CDH, and managed service software on cluster hosts, we used Cloudera Manager to automate installation.

Table 3 shows the roles assigned to the CDH cluster hosts to implement the selected features while balancing the computing, storage, and networking resources needs. Figure 3 shows the status display of the deployed cluster by Cloudera Manager.

Table 3. Roles assigned to CDH cluster hosts

Hostname	Roles
CDHcs1mgr	Cloudera Activity Monitor; Cloudera Alert Publisher; Cloudera Event Server; Cloudera Host Monitor; Cloudera Reports Manager (enterprise version); Cloudera Service Monitor. Hive Gateway; Hive Metastore. Hue Server.
CDHcs1MN1	HBase Master. HDFS Httpfs; HDFS Namenode-Active. Hive Gateway; Hive HiveServer2. Spark Master. Zookeeper Server – follower.
CDHcs1MN2	HBase Region Server. HDFS Datanode; HDFS NFSGateway; HDFS NameNode – Secondary. Solr Server. Spark Worker.
CDHcs1DN1	HBase Region Server. HDFS Datanode. HIVE Gateway. Oozie Server. YARN Job History; YARN Node Manager; YARN Resource Manager.
CDHcs1DN2	HBase Region Server. HDFS Datanode. Hive Metastore; Hive HiveServer2; Hive WebCat. YARN Node Manager.
CDHcs1DN3	Flume Agent. HBase REST server. HDFS Datanode. YARN Node Manager. Zookeeper

	Server – leader.
CDHcs1DN4	HBase Thrift Server. HDFS Datanode. YARN Node Manager. Zookeeper Server – follower.

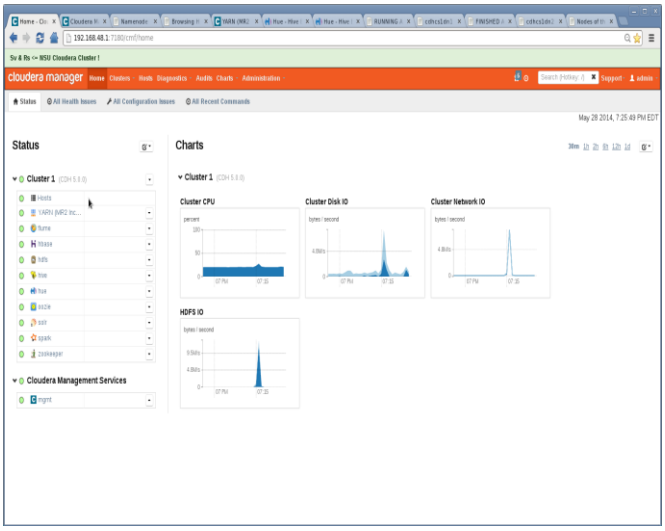


Figure 3. Cloudera Manager status display

Installing and configuring this Hadoop cluster using the semi-automated approach was quite challenging. Some of the important lessons learned include:

(a) Permissions. Many of the Hadoop packages require access to shared resources. The beta version of the installation guide, as far as we knew, did not include detailed instructions on setting up the appropriate permissions for various Hadoop components so they could work together. Hence, we needed to figure out how to grant permissions, primarily through group memberships and group permissions, to various Hadoop software and resources (e.g., mapred must be a member of the hadoop group as well). This issue has been addressed by the latest version of the installation guide.

(b) User accounts and groups. Similarly, the latest guide also provided detailed instructions on the user accounts and groups that Cloudera Manager and CDH used to complete their tasks. This standardized setup should be followed to make sure the user accounts, groups, and permissions are consistent across all hosts. This also makes it easier to ensure that the environmental variables and profiles are set up consistently across hosts.

(c) Interdependencies. Although it might not be stated in the documentation, the order in which the various packages are installed may make a difference in the ease of configuring these packages that have interdependencies. For example, our experience indicated that it was better to install and configure the ZooKeeper before installing Hive. Attempting to install ZooKeeper after Hive was installed could cause issues with the HiveServer service.

(d) Performance. Although Hadoop has a very flexible distributed architecture, sometimes it is better to run closely related services/tasks on the same physical host to reduce the latency and overhead. This was especially important during the installation phase and using hardware with limited resources.

(e) It is critical to keep a close watch on disk storage and memory use. The available disk space could be depleted when a large volume of log files were generated. The available memory could also be depleted after a period of operation. Running low on disk space and memory usually caused systems to reboot or become nonresponsive.

(f) Files in some directories could be deleted by Cloudera Manager after making configuration changes through Manager. Make sure important files are not kept in these directories or they are backed up somewhere else.

2.6 Hadoop App for Graph Processing

Graph-based processing was one of the first categories of Hadoop applications in which we were interested. So we worked with Apache Giraph (v1.0.0) which "is an iterative graph processing system built for high scalability. For example, it is currently used at Facebook to analyze the social graph formed by users and their connections." [16]

Again, we used a phased approach in working with Giraph. First, we followed the Giraph Quick Start guide [17] to install and run Giraph in a single-node, pseudo-distributed Hadoop cluster on a VM loaded with Ubuntu Server 12.04.2 (64-bit) OS. We verified that the installation was operational by running the "SimpleShortestPathsComputation" example job and obtaining the desired output successfully.

Next we proceeded to install Giraph on the multi-node Cloudera based cluster described in the previous section. For this exercise, we used the information contained in another resource [18] to help install Giraph on CentOS which is the base OS for our Cloudera based cluster. Again, we ran the "SimpleShortestPathsComputation" example job to verify that the Giraph installation on this cluster was operational.

Our experience indicated that the node on which Giraph is executed should also have YARN (MR2) Node Manager service, HDFS DataNode service and ZooKeeper service running on the same node for better performance and increased level of robustness.

Without accessibility to Zookeeper, we experienced problems with running example Giraph jobs, as multiple failures could occur without clear error messaging. Also, other execution errors occurred with Giraph when the job was not run on a node with YARN Node Manager or the YARN node is not specified. Giraph and YARN work closely together. With large Giraph calculations, the connectivity to a remote Mapreduce service could become disconnected and cause the Giraph job to fail.

Running Giraph job was a bit of a challenge. As stated before, denoting the nodes that run the ZooKeeper service can help prevent failures. Giraph does come with example code that provides a wide range of functionality. For example, "SimpleShortestPath" works well with a properly formed file with adjacency lists. However, a user needs to make sure that no extraneous white spaces or blank lines are

included in the input text file. Otherwise, this example job could fail. However, the "PageRankBenchmark" example job did not actually produce any output, although it could be completed successfully.

3 Infrastructure Enhancement Planned

As mentioned earlier, an enhancement to the current infrastructure is planned for completion by 4Q2014. A new "production" system with five master nodes and twelve data nodes will be installed in a server room, while another new "integration and testing" system with five master and data nodes will be deployed in a research lab. The current systems will remain in the research lab and used primarily for learning, development, and development testing purposes.

The new equipment will add approximately six hundred Intel Xeon 64-bit CPU cores, 350 terabytes of disk storage, 3 terabytes of RAM, and three high-performance L2/L3 Ethernet switches supporting 40GbE connectivity.

4 Summary

This paper presented our lessons learned in building a big data research and education infrastructure. As big data continues to gain rapid growth in research, development, and deployment, it is important and beneficial for organizations in both public and private sectors to leverage big data to gain insights and improve their operation. It is also important and beneficial for educational institutions to engage in big data related research, education, and workforce development to help advance the state of the art of this critically important technology, and address the talent shortage problem forecast for many years to come.

However, due to the complexity, immaturity, and fast pace in evolving of big data platforms and tools, it is very challenging to build up a big data research and education infrastructure in both human and system resources, especially for small to medium businesses, organizations, and educational institutions without plenty of resources.

We took an incremental and iterative approach to build a small size infrastructure at a university with about 6,000 students in enrollment, without requiring investments in staff and hardware/software resources. The knowledge and skills were acquired through student research projects required for their degrees. This approach provided additional benefits to the students' professional development. The hardware used for this effort was all commodity hardware already available in the institution. The software used was all open source or free.

Even so, it was very challenging to get it done. Good planning, perseverance, and dedicated personnel can prevail.

5 Acknowledgement

This research was supported in part by U.S. Army Research Office, under grant numbers W911NF-12-1-0081 and W911NF-14-1-0045, and U.S. Department of Energy, under grant number DE-FG52-09NA29516/A000.

6 References

[1] NIST Big Data Public Working Group, "DRAFT NIST Big Data Interoperability Framework: Volume 1, Definitions (Draft Version 1)," April 23, 2014.

[2] International Data Corporation, "Worldwide Big Data Technology and Services 2013–2017 Forecast," Dec 2013.

[3] "Apache Hadoop," [Online]. Available: http://hadoop.apache.org/. [Accessed 31 May 2014].

[4] "Hortonworks Sandbox," [Online]. Available: http://hortonworks.com/products/hortonworks-sandbox/. [Accessed 31 May 2014].

[5] "Hortonworks Data Platform," [Online]. Available: http://hortonworks.com/hdp/. [Accessed 31 May 2014].

[6] "Apache Ambari," Hortonworks, [Online]. Available: http://hortonworks.com/hadoop/ambari/. [Accessed 31 May 2014].

[7] "Cloudera QuickStart VM," [Online]. Available: http://www.cloudera.com/content/cloudera-content/cloudera-docs/DemoVMs/Cloudera-QuickStart-VM/cloudera_quickstart_vm.html. [Accessed 31 May 2014].

[8] "Cloudera CDH," [Online]. Available: http://www.cloudera.com/content/cloudera/en/products-and-services/cdh.html. [Accessed 31 May 2014].

[9] "Cloudera Manager," [Online]. Available: http://www.cloudera.com/content/cloudera/en/products-and-services/cloudera-enterprise/cloudera-manager.html. [Accessed 31 May 2014].

[10] "CentOS," [Online]. Available: http://www.centos.org/. [Accessed 31 May 2014].

[11] "Hortonworks Data Platform: Installing Hadoop Using Apache Ambari," Hortonworks, 2013.

[12] "Cloudera QuickStart VM," [Online]. Available: http://www.cloudera.com/content/support/en/downloads/download-components/download-products.html?productID=F6mO278Rvo. [Accessed 31 May 2014].

[13] G. Turkington, Hadoop Beginner's Guide, Birmingham: Packt Publishing, 22 Feb 2013, p. 398.

[14] A. Holmes, Hadoop in Practice, Shelter Island, NJ: Manning Publications Co., October 2012, p. 536.

[15] Cloudera, "Cloudera Manager Installation Guide (Version 5.0.1)," Cloudera, 2014.

[16] "Apache Giraph," Apache Software Foundation, [Online]. Available: https://giraph.apache.org/. [Accessed 4 June 2014].

[17] "Apache Giraph Quick Start," Apache Software Foundation, [Online]. Available: http://giraph.apache.org/quick_start.html. [Accessed 4 June 2014].

[18] "Install giraph in hadoop node," [Online]. Available: http://www.sbarjatiya.com/notes_wiki/index.php/Install_giraph_in_hadoop_node. [Accessed 4 June 2014].

Big Data System Design for Digital Library Middleware

Behrooz Seyed-Abbassi and Chris Jerome Carey
School of Computing, University of North Florida, Jacksonville, Florida, USA

Abstract - *Libraries around the world contain vast amounts of assets that come in many different forms and formats, such as books, journals and magazines, audio and video CDs/DVDs, and microfilm. As technology is incorporated into libraries, many of these assets have been converted into digital formats that are stored on a computer located within the library. In order to handle the large amounts of digital data, libraries are turning to storehouses appropriately named digital libraries and to date, a multitude of digital libraries have been created. With the introduction of so many digital libraries, the generation of duplicate digital assets has become an unavoidable circumstance. To seek a remedy for this dilemma, consortiums have been established with the sole intent to develop standards for tools that can be built to offer central cataloging repositories. The main function of such repositories is to facilitate the sharing of the holdings of those libraries that have implemented a digital library system within their institutions. In this research work, the current central cataloging repositories are overviewed and areas in the tools that could be improved are identified. Then, a methodology work is described that may ease the discovery of available assets by allowing libraries to share their current holdings using data warehouse and storing accurate information for various search need and data mining. It would also enable those working in the libraries to find information about holdings that their institutions may not possess but are considering to acquire.*

Keywords: Digital Libraries, Database Design, Data Warehouse, Middleware, Retrieval

1 Background

The Integrated Library System (ILS) was introduced out of the necessity to share information about a library's bibliographic holdings in a digital manner [1], [2] and [3]. Prior to the introduction of the ILS, libraries around the globe held information about their holdings in a paper-cataloging format leading to multiple libraries holding the exact same of similar items. The result was a wide spread duplication of data.

To address the data duplication issue, a consortium was put together to develop a set of standards that would aid in the sharing of data between institutions. The consortium developed a set of standards by which information about the holdings of libraries could be stored digitally [4]. Currently, there are numerous standard formats that an institution can use to store their digital data.

Of the formats in use, there are a few major ones that are more widely utilized. One common format, Z39.50, was introduced by the America National Standards Institute (ANSI) as a network protocol that provides access to information systems and is currently being maintained by the Library of Congress (LOC) which provides a listing of institutions using the protocol [5], [6]. Another highly used standard, the Machine-Readable Cataloging (MARC) format and its successors MARC21 AND MARCXML, are data formats that simplify the transfer of data between institutions [7]. A third example, the Dublin Core (DC) format, was introduced by the Dublin Core Metadata Initiative (DCMI) as a set of simple standards that facilitate the finding, sharing, and management of information [8].

The ability to store data digitally and to share information about the stored data has resulted in the creation of tools that make sharing between institutions much easier. On the commercial side, there are tools such as Millenium (created by Innovative Interfaces Inc.), Aleph (created by Ex Libris), and Symphony (created by SirsiDynix) [9] that function as repositories for digitally stored data. On the open source side, there are tools such as Koha (funded by a group of New Zealand libraries) and Evergreen (begun by a Georgia consortium of public libraries) [10] that are also repositories for digital data. Other open source offerings include Connexion (introduced by the Online Computer Library Center (OCLC)) [11] and OAIster (originally funded by a grant from the Andrew W. Mellon foundation and created by the University of Michigan) that were both created as a means of providing access to millions of records representing open access resources [12]. Another open source offering is a tool by the name of MARCEdit, which is intended to provide individual users with an interface that allows them to import data in one format and transform it into a format more suitable to their needs [13].

In Section 2, existing tools such as metadata harvesting and cross-walking methodology are discussed. The research method and how to improve searching, sorting, sharing data by utilizing a data warehouse for users of the digital libraries are introduced in Section 3. Software and hardware requirements are considered in Section 4. Conclusions are presented in Section 5.

2 What's Out There and What's Missing

Many of the tools in existence were created with the intention of facilitating the processes of collecting and sharing data. Connexion and OAIster focus on the collection and storage of data in order to provide multiple users with a robust searching interface and advanced searching capabilities that make finding specific information less of a challenge [11], [12]. Tools like MARCEdit and MetaLib provide users with desktop applications, which allow for local aggregation of data and for transformation of the data format as needed [13].

In order to accomplish the intended goals, these tools use multiple methodologies provided by the convening standards organizations. For the collection of information, a methodology named Metadata Harvesting created by the Open Archive Initiative (OAI) is often employed [14]. Harvesting is used, primarily, to aggregate or collect metadata descriptions of records available in an archive and then store those findings for future use [15]. In many cases, the metadata information must undergo a transformation process before storage can take place. The Cross-Walking methodology is frequently employed for the transformation process to show equivalent elements between differing standard formats [16] The benefit is that the data can be converted from one format to another with speed, ease, and minimal data loss [16].

While the functionality of the existing tools is valuable in aiding users in locating available data, there are areas in which they fall short. For instance, the tools that take advantage of a backend database, store the data in a predetermined format that lacks a means of converting the holdings into other formats if needed. In contrast, standalone desktop applications that provide a means of conversion, lack the information stored in a backend database, and thus must rely on external repositories for their information.

Another drawback of the tools that are currently available is the limitation of the number of records that are returned with each search. Connexion, for example, limits this number to 100, leading to the need to send multiple requests out to obtain all possible records for a search [11]. Such limitations lead to longer waiting times for the results to be returned to the requester, as well as higher processing needs being placed upon the server in which the application is running.

3 Research Method and Work

During the research process, it became evident that no matter whether the tools were developed to be a full-blown ILS or a stand-alone data aggregation application, they were all designed for sharing digital information by either a process of storing and displaying data or by finding data from an external repository and converting it to an alternative format. It also became clear that somewhere in between the two ends of the spectrum a methodology could be developed that would maximize their best features. For the ILS, these features include faster searching and an intelligent searching methodology. In terms of the stand-alone data aggregation system, the cross walking methodology employed provides the ability to convert objects from one format to another format.

This research work proposes a methodology that take advantage of the positive features and explore enhancements that may improve the searching, storing, and sharing of data for users of digital libraries. It would combine features of both tools as an integrated system with a unified data collection using standards for field structures and a display format similar to the Dublin Core (DC), which provides a comprehensive format of twenty-one fields for identifying and displaying data [8]. The designed and implemented methodology as shown in Figure 1 integrates various environments as Data Warehouse. The listed labels for the overview design are shown below.

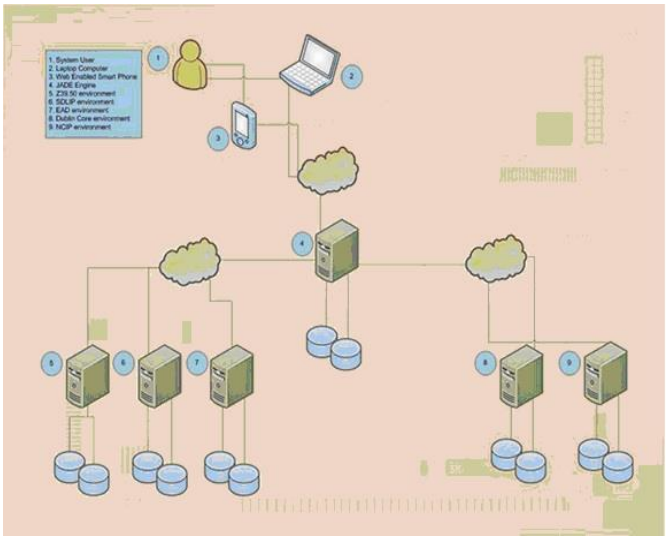


Figure 1. Overview of Designed Methodology

As shown in Figure 1, the methodology includes the data harvesting that utilizes different systems of 5, 6, 7, 8 and 9 to load the data warehouse (4). A system user (1) can utilize a laptop or smart phone to access the data warehouse for needed information.

1. System User
2. Laptop Computer
3. Web Enabled Smart Phone
4. System Data Warehouse and Search (JADE) engine
5. Z39.50 Environment
6. Simple Object Access Protocol(SOAP) Environment
7. EAD Environment
8. Dublin Core Environment
9. NCIP Environment

Figure 2 shows the detailed components of the data warehouse, a search management process with request processor and harvesting manager, and external repositories (or libraries).

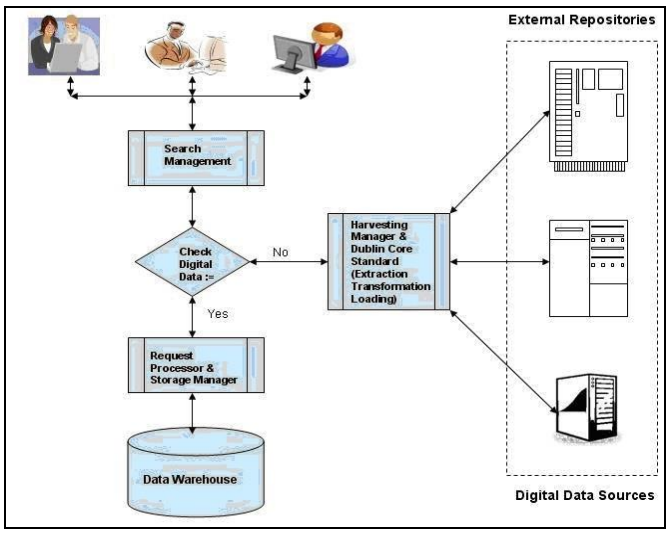


Figure 2. Detailed Diagram for Data Warehouse Methodology

The design provides comprehensive digital information retrieval that can take advantage of heterogeneous digital information from multiple data sources under the unified format [17]. The system stores the information in a data warehouse for ease of retrieval and include an option to locate digital data within a particular distance.

The data warehouse component supports a record structure similar to DC for bibliographical records with selected metadata fact information for title, subject, type, creator, publisher, and audience for categorization and optimization of search process as displayed in Figure 3. The dimension table for the record supports about 22 fields similar to the DC format and includes the metadata fields. The location dimension includes information regarding each individual library's place and locality information. The metadata dimension will hold the above six values for metadata categorization and has the capability to expand. The time dimension includes the date that the item was inserted into the warehouse.

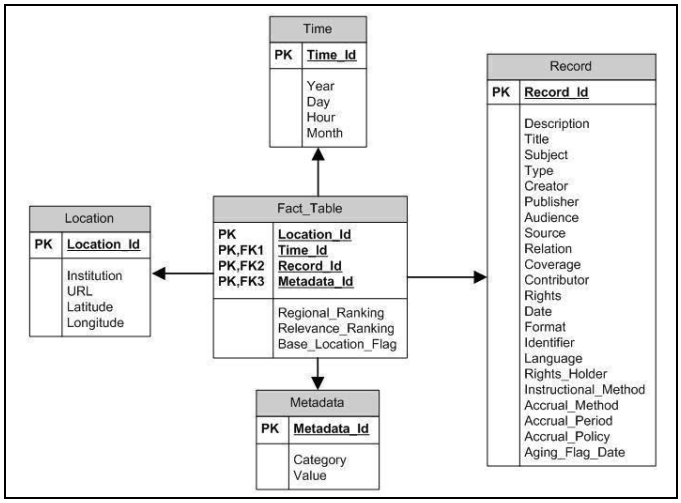


Figure 3. Data Warehouse for Unified Digital Information

The information that is stored in the data warehouse are gathered from a set of external repositories. The repositories may not communicate to each other but based on the design, they could disseminate information to the data warehouse through a harvesting manager. The harvesting manager utilizes an extraction, transformation, and loading (ETL) process to take advantage of the search process in the management component. When a user requests digital information, the search management component will ask the request processor and storage manager to check the data warehouse first for the digital information.

3.1 Transformation and Conversion Process

When a users submits a request, if the information is located in the data warehouse, the system returns the bibliographical records based on the metadata requested by the user. Otherwise, the search management component uses the harvesting manager for inquires from external repositories. Since each repository can have its own structure for digital information, the extracted search result may need to be converted or transformed to the bibliographical record structure based on the DC format using the algorithm shown in Figure 4.

1. Receive a set of bibliographic records
2. While there are more items in the set
   a. Determine the format of the current record
   b. If the record is already in the DC format
      i. Move to the next record
   c. Otherwise
      i. Determine the mapping needed to convert to DC
      ii. Apply mapping conversion
      iii. Replace current record in the set with new DC record
3. Return new set to calling process

Figure 4. Transformation and Conversion Process

3.2 Metadata Extraction

The transformed information extracted from external repositories using a search phrase will be returned to the search manager to be provided to the user and also loaded to the data warehouse. Storing information in the data warehouse allows the system to have ease of access and fast retrieval in the future, if users issue search request that are similar or that relate closely to information that has already been retrieved. Once each extracted record has been converted, it should be analyzed in a manner for mapping to the metadata. Figure 5 shows how records can be mapped to the associated metadata. Then, the data can be provided either to the data warehouse or to the user with the metadata categorization [18], [19].

1. Receive a set of DC records
2. While there are more records to analyze
   a. Find the six metadata items
   b. Store the six items in the metadata dimension
   c. Associate each item with the record
3. Return to calling process

Figure 5. Metadata Extraction

3.3 Relevance Ranking

Before the located metadata records are stored in the data warehouse or sent to the user, they will be associated with all their related metadata and information based on requested phrase or criteria from external repositories. Such an association means that when a search is executed for a phrase that matches against the metadata in the data warehouse then all associated records will be returned. This provides the requester with more information than they would have originally received with just searching against the records alone from external repositories. The warehouse also keeps track of items viewed by a user and associate the information with records opened during subsequent searches in a given period of time to provide suggestions to users looking at similar content.

The search information for multiple results can take advantage of a ranking schema. More data or multiple results can be organized by a relevance ranking based on a earch phase or criteria. If the result being returned is an exact match to the full phrase, the highest relevance ranking will be assigned to the record. If the results match each word of the phrase but are not an exact match, the next highest relevance ranking will be assigned to the record. This pattern of determination will continue in a manner that will start with records that match all but one individual word, and will continue until a point is reach where records that match only one word are found. The relevance ranking will be assigned in a decreasing fashion for the remaining determinations as well. The relevance ranking is shown in Figure 6.

1. Receive a set of records and a set of search criteria
2. While there are more records to review
   a. If the search criteria is a full phrase search
      i. Perform an exact phrase match
      ii. Assign highest ranking
   b. Otherwise
      i. Perform a match against the list of individual words
      ii. Assign appropriate ranking
   c. Attach criteria to found metadata entries
3. Store all ranking information in the warehouse

Figure 6. Relevance Ranking

3.4 Regional Ranking

Finally, before a record is stored in the data warehouse, it will be associated with the external repository region as shown in the dimension table of location from Figure 3. The region association will use the algorithm in Figure 7 for the assignment of location information to be used during the execution of the process that searches the data warehouse for future requests. When a search is initiated, a base location will be determined. This base location is used in the sorting of the records, where those records that exist geographically closest to the base location are pushed to the top of the listing, and those records that exist geographically farther away will be pushed to the bottom.

1. Receive a set of records, repository information and search base location
2. While there are more records
   a. Associate each record with a repository
   b. Based on the base location determine the distance from the repository
   c. Associate distance with the record
   d. Based on the distance assign a regional ranking to the record
   e. Associate the ranking with search criteria supplied by the user
3. Store all found information in the data warehouse
4. Return to calling process

Figure 7. Regional Ranking

3.5 Aging Process

It is essential that stored information in the data warehouse is current and up to date. For that reason, a process must be included to check the data warehouse for current information and to prune the old information. The purpose of the pruning process is to take each record in the data warehouse and issue a request against the repository from which the record was retrieved. The intention of this request is to ensure that the item still exists. If the item does still exist, no further processing is needed. However, if the item no longer exists, the item should be removed from the search listing by flagging it as old, so that it cannot be included in subsequent searches but should not be removed from the data warehouse. If the item still exists, but the information has changed, a new record should be stored in the data warehouse and the old record should be removed from the search listing so that it can not be included in subsequent requests.

The purpose of keeping the records that no longer exist as well as those items that have been changed is for historical purposes. A timestamp in reference to the data in which a change was made will be associated with these flagged records. This is a process that will be given the name "aging". The aging process will allow the pruning process to gain a better understanding of how often the information changes as well as helping with faster search processes. With this knowledge, the pruning process should be able to adjust the frequency with which it should run without the intervention of a system administrator.

3.6 Searching Process

The searching process will be handled in two different areas of the designed methodology. One is the harvesting manager and the other is the request processor, shown in Figure 2.

The search by the harvesting manager is performed on external repositories based on a phrase or criteria as an extraction using the ILS and crosswalk tools for data retrieval [20]. Although many of the ILS listed earlier are proprietary software, open-source software is available with similar capabilities that can be used for testing and evaluating results of the methodology comparisons. Also, the search

performance results may be improved by the use of multi-threading

A request processor in the design will implement loading and searching of information to and from the data warehouse. The process could benefit from utilizing the metadata, relevance ranking, and regional ranking by way of increased speed of return times as well as accurate information. Also, the search performance results may be improved by the use of indexing. Response times will be collected among different searches.

4 Software and Hardware

The designed system will have web interfaces to support access to data from different sources. The web services would need to adhere to currently provided standards such as Search/Retrieve via URL (SRU) and Search/Retrieve Web service (SRW) [21], [17]. The middleware is developed using Java programming language. To build the system, the latest version of the Java software development kit. The data warehouse is using the Microsoft SQL Server environment.

Another important factor of the system is the threading methodology used within the searching algorithm. For optimal performance of the parallel operations, a minimum requirement of two processors or a multi-core processor is used.

5 Conclusions

An anticipated benefit of the designed methodology is that the data in the data warehouse will be indexed in a manner that allows for multiple types of searching. Current tools allow for searching against the contents of each bibliographical record. The enhanced design would allow for searching against not only the content, but also the metadata and geographical location of digital information. It would check for association with other records by searching external repositories along side the data warehouse and utilize a threading method for faster access to data stored in the external repository. These additions may result in an increase in the actual number of relevant records returned and a reduced response time to the user requests. The relevance ranking sorts the records so that the ones that most closely match the request are listed first. The regional ranking provides the user with the nearest location of digital library information. Various testing comparisons between the ILS, cross walking, and the designed methodology will be performed to determine if the enhancements provide intelligent searching techniques that yield more relevant results in less time. It is believed the result should provide a potential improvements to sharing of digital information.

6 References

[1] Aruna, A. "Z39.50: An information retrieval protocol". DESIDOC Bulletin of Information Technology vol. 21 , no. 6 , (November 2001): 25-39, Last access: April 2014.

[2] Greenstein, D. & Thorin, S. "The Digital Library: A Biography" Council on Library and Information Resources <http://www.clir.org/pubs/reports/pub109/pub109.pdf>, Last access: April 2014.

[3] Medeiros, N. & Miller, L. "White Paper on Interoperability between Acquisitions Modules of Integrated Library Systems and Electronic Resource Management Systems" Digital Library Foundation Jan. 2008 <http:/www.diglib.org/standards/ERMI_Interop_Report_20080108.pdf>, Last access: April 2014.

[4] Nelson, Micheal L., 2001, "Better interoperability through the Open Archives Initiative"

[5] Aruna, A. "Z39.50: An information retrieval protocol". DESIDOC Bulletin of Information Technology vol. 21 , no. 6 , (November 2001): 25-39, Last access: April 2014.

[6] Library of Congress WWW/Z39.50 Gateway" <http://www.loc.gov/z3950>, Last access: April 2014.

[7] "MARC STANDARDS (Network Development and MARC Standards Office, Library of Congress)" http://www.loc.gov/marc, Last access: April 2014.

[8] Hillmann, Diane. "Using Dublin Core" Dublin Core Metadata Initiative 2005-11-07 <http://dublincore.org/documents/usageguide>, Last access: April 2014.

[9] SirsiDynix SAAS Overview n.d, n.p, <http://lcldinfo.wikispaces.com/file/view/SirsiDynix+SaaS+Solutions.pdf>, Last access: April 2014.

[10] Yang, Sharon Q., and Melissa A. Hofmann. 2010. "The next generation library catalog: A comparative study of the OPACs of Koha, Evergreen, and Voyager." Information Technology in Libraries 29:141-150

[11] "Connexion [OCLC - Cataloging options for your library]" <http://www.oclc.org/connexion>, Last access: April 2014.

[12] Garrison, Adriene, Winter 2010 "OAISTER: The Roots and the Resource, Adriene Garrison"

[13] Reese, Terry, September 2004, "MARCEdit Metadata Suite"

[14] Lynch, Clifford A., "Metadata Harvesting and the Open Archives Initiative," ARL: A Bimonthly Report, no. 217 (August 2001): 1–9.

[15] Lynch, Clifford A., "The Z39.50 Information Retrieval Standard Part I: A Strategic View of Its Past, Present and Future" D-Lib Magazine, April1997 <http://webdoc.sub.gwdg.de/edoc/aw/d-lib/dlib/april97/04lynch.html>, Last access: April 2014.

[16] Chiticariu, Laura, Tan, Wang-Chiew September 2006, "Debugging Schema Mappings with Routes"

[17] Morgan , Eric L. "An Introduction to the Search/Retrieve URL Service (SRU)", n.d <http://www.ariadne.ac.uk/issue40/morgan/>, Last access: April 2014.

[18] Duval, Erik, Hodgins, Wayne, Sutton, Stuart, Weibel Stuart L., Metadata Principles and Practicalities, D-Lib Magazine April 2002 Volume 8, Number 4 <http://www.dlib.org/dlib/april02/weibel/04weibel.html>, Last access: April 2014.

[19] Warner, S. "Exposing and harvesting metadata using the OAI metadata harvesting protocol: a tutorial". HEP Libraries Webzine, 4, 2001. <http://library.cern.ch/HEPLW/4/papers/3>, Last access: April 2014.

[20] Van de Sompel, Herbert, Jeff Young and Thom Hickey "Using the OAI-PMH... Differently," D-Lib Magazine, Volume 9, Number 7/82, July/August 2003. Open Archives Initiative. http://www.openarchives.org/OAI/2.0/guidelines.htm>. Last access: April 2014.

[21] Shen, R. Applying the 5S framework to integrating digital libraries.2006. Dissertation (Doctoral)- Virginia Tech, Blacksburg, VA, USA, 2006. Disponível em: <http://scholar.lib.vt.edu/theses/available/etd-04212006-135018>. Acesso em: 2006, Last access: April 2014.

Finding Critical Samples for Mining Big Data

Andrew H. Sung[1], **Bernardete M. Ribeiro**[2], **Qingzhong Liu**[3], **and Divya Suryakumar**[4]
[1]School of Computing, The University of Southern Mississippi, Hattiesburg, MS 39406, U.S.A.
[2]Department of Informatics Engineering, University of Coimbra, 3030-290 Coimbra, Portugal
[3]Department of Computer Science, Sam Houston State University, Huntsville, TX 77341, USA
[4]Apple, Inc., Sunnyvale, California, USA

Abstract – *To ensure success of big data analytics, effective data mining methods are essential; and in mining big data two of the most important problems are sampling and feature selection. Proper sampling combined with good feature selection can contribute to significant reductions of the datasets while obtaining satisfactory results in model building or knowledge discovery. The critical sampling size problem concerns whether, for a given dataset, there is a minimum number of data points that must be included in any sampling for a learning machine to achieve satisfactory performance. In this paper, the critical sampling problem is analyzed and shown to be intractable–in fact, its theoretical formulation and proof of intractability immediately follow that of the previously studied critical feature dimension problem. Next, heuristic methods for finding critical sampling of datasets are proposed, as it is expected that heuristic methods will be practically useful for sampling in big data analytic tasks*.

Keywords: Sampling, Mining Big Data, Machine Learning

1 Introduction

One of the many challenges of big data analytics is how to reduce the size of datasets without losing useful information contained therein. Many datasets that have been or are being constructed for intended data mining purposes, without sufficient prior knowledge about what is to be specifically explored or derived from the data and how to do it, likely have included measurable attributes that are actually insignificant or irrelevant, which results in large numbers of useless attributes (or features) that can be deleted to greatly reduce the size of datasets without any negative consequences in data analytics or data mining [1]. Likewise, many of these massive datasets conceivably already contain much more data points (or samples, vectors, patterns, observations, etc.) than necessary for knowledge discovery (model building, hypothesis validation, etc.), leading to the questions of what sampling size is sufficient (in, say, machine learning tasks) and how to generate the sample (or training dataset) to ensure successful data analytic results.

For dimension reduction, effective feature ranking and selection algorithms [2] can be utilized to reduce the size of the dataset by eliminating features that are insignificant, irrelevant, or useless. The authors have recently studied the feature dimension problem in general settings by consider the question: Given a dataset with p features, is there a *Critical Feature Dimension* (or the smallest number of features that are necessary) that is required for, say, a particular data mining or machine learning process, to satisfy a minimal performance threshold? That is, any machine learning, statistical analysis, or data mining, etc. tasks performed on the dataset must include at least a number of features no less than the critical feature dimension – or it would not be possible to obtain acceptable results. This is a useful question to investigate since feature selection methods generally provide no guidance on the number of features to include for a particular task; moreover, for many poorly understood complex problems to which big data brings hope of scientific breakthrough there is little prior knowledge which may be otherwise relied upon in determining this number (of critical feature dimension).

Similarly, the question about sampling size can be raised: Given a dataset with n points, is there a *Critical Sampling Size* (or the smallest number of data points) that is required for any particular data mining (or machine learning, etc.) process to satisfy a minimal performance requirement? This is also an important and practical question to consider since various sampling techniques provide no clue with regard to the critical sampling size for any specific dataset. When dealing with big data where the number of data points (the value of n) is huge, the question becomes more relevant.

In previous papers by these authors, the critical feature dimension was shown to be intractable; and yet a simple heuristic method based on feature algorithms was demonstrated to be able to find approximate critical dimensions for many datasets of various sizes, and therefore provides a practically useful solution to the problem.

This position paper shows that the critical sampling size problem, formulated in general, has the same complexity as the critical feature dimension problem. In fact, the same proof of the complexity of the critical feature dimension problem carries over to the critical sampling size problem.

In section 2, the critical sampling size problem is formulated in general terms and shown to be intractable. In

section 3, a simple ad-hoc method is proposed as a first attempt to approximately solve the problem, and some discussions conclude the paper.

2 Critical Sampling Size

Assume the dataset is represented as the typical n by p matrix $D_{n,p}$ with n objects (or data points, objects, patterns, etc.) and p features (or measurements, attributes, etc.) The intuitive concept of the critical sampling size of a dataset with n points is that there may exist, with respect to a specific "machine" M and a given performance threshold T, a unique number $\nu \leq n$ such that the performance of M exceeds T when some suitable sample of ν data points is used; further, the performance of M is always below T when any sample with less than ν data points is used. Thus, ν is the critical (or absolute minimal) number of data points in a sample that is required to ensure that the performance of M meets the given threshold T.

Formally, for dataset D_n with n points (the number of features in the dataset, p, is considered fixed here and therefore dropped as a subscript of the data matrix $D_{n,p}$), ν (an integer between 1 and n) is called the *T-Critical Sampling Size* of (D_n, M) if the following two conditions hold:

1. There exists D_ν, a ν-point sampling of D_n (i.e., D_ν contains ν of the n vectors in D_n) which lets M to achieve a performance of at least T, i.e.,
   $(\exists D_\nu \subset D_n)$ $[P_M(D_\nu) \geq T]$, where $P_M(D_\nu)$ denotes the performance of M on dataset D_ν.
2. For all $j < \nu$, a j-point sampling of D_n fails to let M achieve performance of at least T, i.e.,
   $(\forall D_j \subset D_n)$ $[j < \nu \Rightarrow P_M(D_j) < T]$

Note that in the above, the specific meaning of $P_M(D_\nu)$, the performance of machine (or algorithm) M on sample D_ν, is left to be defined by the user to reflect a consistent setup of the data analytic (e.g. data mining) task and an associated performance measure. For examples, the setup may be to train the machine M with D_ν and define $P_M(D_\nu)$ as the overall testing accuracy of M on a fixed test set distinct from D_ν, or the setup may be to use D_ν as training set and use $D_n - D_\nu$ as testing set. The value of threshold T, which is to be specified by the user as well, may represent a reasonable performance requirement or expectation.

To determine whether a critical sampling size exists for a D_n and M combination is a very difficult problem. Precisely, the problem of deciding, given D_n, T, k $(1 < k \leq n)$, and a fixed M, whether k is the T-critical sampling size of (D_n, M) belongs to the class $\mathbf{D^P} = \{ L_1 \cap L_2 \mid L_1 \in \text{NP}, L_2 \in \text{coNP}\}$ [3], where it is assumed that the given machine M runs in polynomial time (in n). In fact, it is shown in the following that the problem is $\mathbf{D^P}$-hard.

Since NP and coNP are subclasses of $\mathbf{D^P}$ (Note that $\mathbf{D^P}$ is not the same as NP $\cap$ coNP), the $\mathbf{D^P}$-hardness of the Critical Sampling Size Problem (CSSP) indicates that it is both NP-hard and coNP-hard, and thus most likely to be intractable [3,4].

2.1 Proof CSSP is Hard

CSSP: The problem of deciding if a given k is the T-critical sampling size of a given dataset D_n belongs to the class $\mathbf{D^P}$ under the assumption that, for any $D_i \subset D_n$, whether $P_M(D_i) \geq T$ can be decided in polynomial (in n) time, i.e., the machine M can "process" D_i and has its performance measured against T in polynomial time. Otherwise, the problem may belong to some larger complexity class, e.g., Δ^P_2. Note here that (NP $\cup$ coNP) $\subseteq$ D^P $\subseteq$ Δ^P_2 in the polynomial hierarchy of complexity classes [4].

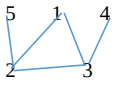
To prove that the CSSP is a D^P-hard problem, we take a known D^P-complete problem and transform it into the CSSP. We begin by considering the maximal independent set problem. In graph theory, a Maximal Independent Set (MIS) is an independent set that is not a subset of any other independent set; a graph may have multiple MIS's.

EXACT-MIS Problem (EMIS) – Given a graph with n nodes, and $k \leq n$, decide if there is a maximal independent set of size exactly k in the graph is a problem which is D^P-complete [3]. Now we describe how to transform the EMIS problem to the CSSP.

Given an instance of EMIS (a graph G with n nodes, and integer $k \leq n$), construct an instance of the CSSP such that the answer to the given instance of EMIS is Yes iff the answer to the constructed instance of CSSP is Yes, as follows: let dataset D_n represent the given graph G with n nodes (e.g., D_n is made to contain n data points, each with n features, representing the symmetric adjacency matrix of G); let T be the value "T" from the binary range {T, F}; let $\nu = k$ be the value in the given instance of EMIS; and let M be an algorithm that decides if the dataset represents a MIS of size exactly ν, if yes P_M = "T", otherwise P_M = "F"; then a given instance of the D^P-complete EMIS problem is transformed into an instance of the CSSP.

2.2 Explanation of Proof

Consider the 5-node graph given below, with its adjacency matrix:



0	1	1	0	0
1	0	1	0	1
1	1	0	1	0
0	0	1	0	0
0	1	0	0	0

This represents a graph with exactly one MIS of size 3, which is {1,4,5}, correspond to the shaded rows.

Example 1: k=3. Threshold T = "T" from the binary range {T, F} to mean true, ν = 3, and an exact MIS of size 3 exists in D_5 as highlighted in the adjacency matrix of G above. So,

algorithm M that decides if the dataset D_5 contains a MIS of size exactly 3 (or M "verifies" that some D_3 corresponds to a MIS of size 3) succeeds; i.e., $P_M(D_3)$ = "T" for some D_3. Since the solution to the instance of EMIS problem is yes, solution to the constructed instance of the CSSP is also yes, as required for a correct transformation.

Example 2: k=4. The constructed instance of CSSP has T = "T" and ν = 4. From D_5 it can be seen that there does not exist any independent sets of size 4, so no exact MIS of size 4 exists. Let M be an algorithm that decides if the dataset D_5 represents a graph containing a maximal independent set of size 4. In this instance M fails to find an exact MIS of size 4 and thus P_M = "F", i.e., $P_M(D_4)$ = "F" for all possible D_4. So the solution to the constructed instance of CSSP is no, as is the solution to the given instance of EMIS.

Example 3: k=2. The constructed instance of CSSP has T = "T" and ν = 2. Independent sets of size 2 exist but they are not MIS's, so algorithm M that decides that some $D_2 \subset D_5$ correspond to an MIS of size exactly 2 fails. The solution to the constructed instance of CSSP is no, as is the solution to the given instance of EMIS, as required.

The D^P-hardness of the Critical Sampling Size Problem indicates that it is both NP-hard and coNP-hard; therefore, it's most likely to be intractable (that is, unless P = NP).

In mining a big dataset $D_{n,p}$ the data analyst is naturally interested in obtaining $D_{\nu,\mu}$ (a ν-point sampling with μ selected features, and hopefully $\nu << n$ and $\mu << p$) to achieve high accuracy in model building or knowledge extraction. From the above analysis of the CFDP and CSSP, this is clearly a highly intractable problem and therefore calls for heuristic solutions.

3 Heuristic Methods for CSSP

The authors of this paper have previously studied heuristic methods for solving the critical feature dimension problem –due to its theoretical intractability, heuristic methods for approximate solutions are clearly called for [5]. Among the findings of the large number of experiments on datasets:

- Simple methods (such as eliminating one feature at a time) produced successful results in finding a critical number of features that is necessary to ensure performance of M exceeds a threshold. The heuristic method used in [5] works in conjunction with a feature ranking algorithm and purports to identify the critical features.
- The critical feature dimension, as determined experimentally by the heuristic method, is in fact different from–but hopefully close to–the formally defined critical feature dimension.
- For datasets with large numbers of features, their critical feature dimension may be much smaller than the total number of features, as shown in Figure 1.
- Many datasets, of various sizes, exhibit the phenomenon of having a critical feature dimension.
- If the critical feature dimension indeed exists for a dataset, then the performance of M is largely preserved when only the critical features are used, as shown in Figure 2.
- The feature ranking algorithm employed in the heuristic method has more significant influence (than the learning machine) on the value of the critical feature dimension.

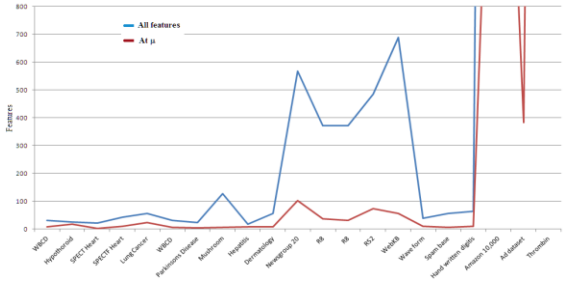


Figure 1. Reduction in feature size at the critical dimension

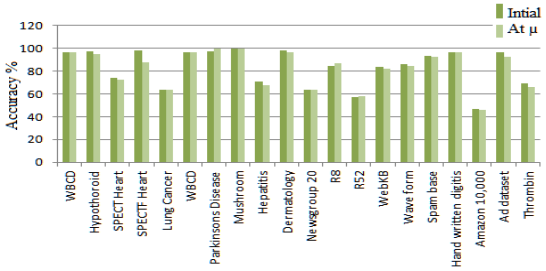


Figure 2. Accuracies with all features, and with critical features selected by the heuristic method

As the simple heuristic method is computationally feasible and appears to be quite sufficient (for many of the datasets studied in the experiments) in finding the critical feature dimension despite the problem's intractability, hope is raised that heuristic methods can be designed to approximately solve the critical sampling size problem satisfactorily as well. Proposed in the following as our position on the CSSP problem is such a heuristic method:

1. Apply a clustering algorithm such as k-means to partition D_n into k clusters.
2. Select, say randomly, m points from each cluster to form a sampling D with m·k points.
3. Apply M (learning machine, analytic algorithm, etc.) on the sample, then measure performance $P_M(D)$.
4. If $P_M(D) \geq T$, then D is a critical sampling, and its size ν is the critical sampling size for (D_n, M). Otherwise enlarge D by randomly select another m points from each cluster, and repeat until a critical sampling is found, or the whole D_n is exhausted and procedure fails to find ν.

The values of the parameters k and m are to be decided in consideration of the size and nature of the dataset, the specific data analytic problem or task being undertaken, and the amount of resource available. As usual in all data analytic problems, prior knowledge and domain expertise are always

helpful in designing the experimental setup. Likewise, whether the random sampling is done with or without replacement is a decision to be made according to the dataset and the problem. Also, experiments may need be performed repeatedly and adaptively (with regard to k and *m*) to obtain good results.

The authors are conducting experiments on many large datasets to observe if the "critical sampling size" indeed exists, and if so whether it is much smaller than the size of the whole dataset.

4 Conclusions

The issue of data mining and association rule extraction, etc. from small samples of large datasets have been studied by many authors before [6,7,8,9], and formal sampling techniques have been studied extensively in e.g. [10]. However, the problem of the critical sampling size of a dataset has not been studied previously. Not surprisingly, a complexity analysis of the problem, in its most general formulation, shows that it is highly intractable (in the sense of being both NP-hard and coNP-hard), thus defying any attempt for exact solutions and calling for heuristic methods for approximation.

Encouraged by the success of simple heuristic methods in finding critical feature dimensions of datasets with large numbers of features [11], a heuristic method is proposed in this paper for finding the critical sampling size of large datasets, and experiments are underway to validate the concept. Even though simple enough, the heuristic method–if it turns out to be successful like the simple heuristic method for finding critical feature dimension–can serve to provide a practical solution for sampling in data mining, which should be highly useful in coping with some of the challenges of big data [12].

We conclude with this statement of our position: Under formally defined conditions of optimality, both the feature selection problem and the sampling problem easily become intractable; however, simple and practically useful heuristic solutions can often be developed to deal with the feature selection and sampling size problems in data mining.

5 References

[1] H. Almuallim and T. G. Dietterich, "Learning with many irrelevant features", Ninth National Conference on Artificial Intelligence, MIT Press, pp.547-552, 1991.

[2] A. Blum and P. Langley, "Selection of relevant features and examples in machine learning", Artificial Intelligence, Vol. 97, 1997.

[3] C. H. Papadimitriou and M. Yannakakis, "The complexity of facets (and some facets of complexity)", Journal of Computer and System Sciences Vol. 28 No. 2, pp.244-259, 1984.

[4] M. R. Garey and D. S. Johnson, "Computers and Intractability: A Guide to the Theory of NP-Completeness", W. H. Freeman and Compnay, 1979.

[5] Q. Liu, B. M. Ribeiro, A. H. Sung and D. Suryakumar, "Mining the big data: the critical feature dimension problem", Proceedings of 2nd International Conference on Smart Computing and Artificial Intelligence (ICSCAI 2014), August 2014.

[6] J. Kivinen and H. Mannila, "The power of sampling in knowledge discovery", Proceedings of PODS '94, the 13th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, pp.77-85, 1994.

[7] H. Toivonen, "Sampling large databases for association rules", Proceedings of VLDB'96, 22th International Conference on Very Large Data Bases, pp.134-145, 1996.

[8] C. Goh, M. Tsukamoto and S. Nishio, "Fast methods with magic sampling for knowledge discovery in deductive databases with large deduction results", Proceedings of ER'98, the Workshops on Data Warehousing and Data Mining: Advances in Database Technologies, pp.14-28, 1999.

[9] C. Domingo, R. Gavaldà and O. Watanabe, "Adaptive sampling methods for scaling up knowledge discovery algorithms", Data Mining and Knoledge Discovery, Kluwer Academic Publishers, Vol. 6 No. 2, pp.131-152, 2002.

[10] J. S. Vitter, "Random sampling with a reservoir", ACM Transactions on Mathematical Software, Vol. 11 No. 1, pp.37-57, 1985.

[11] D. Suryakumar, "The Critical Dimension Problem – No Compromise Feature Selection", Ph.D. Dissertation, New Mexico Institue of Mining and Technology, 2013.

[12] National Research Council, "Frontiers in Massive Data Analysis", The National Academies Press, 2013.

Semantic Indexing of Big Data Using a Hierarchical, Multidimensional Scheme

R. J. Wroblewski
SSC Pacific, San Diego, CA, USA

Abstract: *Semantic information can be captured after a fashion in the form of RDF triples or quads. Often these triple or quad stores can be billions of statements or more. Having thus passed into the realm of "big data", there is an acute need for efficient methods of searching these stores to extract subgraphs for processing. This paper outlines such a method that takes advantage of multidimensional indexing schemes combined with a hierarchical (i.e. semantic) ordering to form a semantic index. A proof-of-concept demonstration was conducted with a simulation creating random RDF triples drawn from a simple ontological taxonomy. As expected, semantically similar triples indexed close together.*

Keywords: Semantic Indexing, Multidimensional Indexing, Space-Filling Curves, Z-Order Indexing, Big Data

1 Introduction[1]

The Resource Description Framework (RDF) was developed to capture semantic information of web and computer-stored objects to facilitate the transfer of this metadata. Although an RDF triple is referred to as subject/predicate/object (SPO), this is properly an object-name/attribute/value. However, this format is conveniently extended by many to the natural usage suggested by the SPO designation in the realm of real-world objects.

Such triples are used to build social-networking graphs, to capture extracted metadata from unstructured text files, or to organize business analytics, to name a few instances. Data stores of these triples can quickly grow into billions of elements and become unwieldy to search or process. Typical big-data databases are NoSQL-based using some form of a key/value design for storage. Efficient search and retrieval of this data is facilitated by an efficient indexing key.

[1] This technology may be the subject of one or more invention disclosures assignable to the U.S. Government.

A typical way to achieve this is merging multiple keys into a single indexing key. Often these multidimensional indexes are formed with the support of space-filling curves [1, 2]. That is, a curve that maps discrete points spanning an n-dimensional volume into a one-dimensional series. Common examples are Z-order curves, Hilbert curves, and Gray-code curves. Roughly, the efficiency of using multidimensional indexing comes from the feature that things that are close in some sense in n-dimensional space will then be close in the one-dimensional mapping, at least on average.

While efficient, such indexing doesn't fully capture the semantic content available. In particular, the RDF elements are generally constructed as elements derived from an ontology. The hierarchical structure of the ontology (i.e. the taxonomy) expresses the core of the semantic information embedded in the ontology. A hierarchical-indexing scheme would allow searching of all the children of a parent element at once. On the other hand, mixing the RDF elements into a key index in a structureless way would require searching over all of the combinations of the sets of children of each of the elements to achieve the same result. Hence, merging ontology-based hierarchical-indexing with multidimensional indexing would give us a workable and efficient semantic-indexing method.

2 Multidimensional Indexing

There are any number of multidimensional-indexing designs available. A simple one is just concatenating the elements. A major drawback with this method is that it is strongly biased toward the first dimensions in searches and not particularly efficient for more than three or so dimensions. Principally, this is because the "closeness" of later dimensions is limited by needing to index through all the elements of the previous dimensions before incrementing.

Some recent schemes [3, 4, 5] for indexing RDFs, including Rya, use this concatenation method. Their approach to working the closeness issue is to generate multiple indexes such as SPO, PSO, OSP, etc. This produces workable indexes that perform significantly better than individual S, P, or O searches, but are still inefficient

as multidimensional indexes. Furthermore, as mentioned above, this simple mixing of elements doesn't capture the semantic content of the RDF.

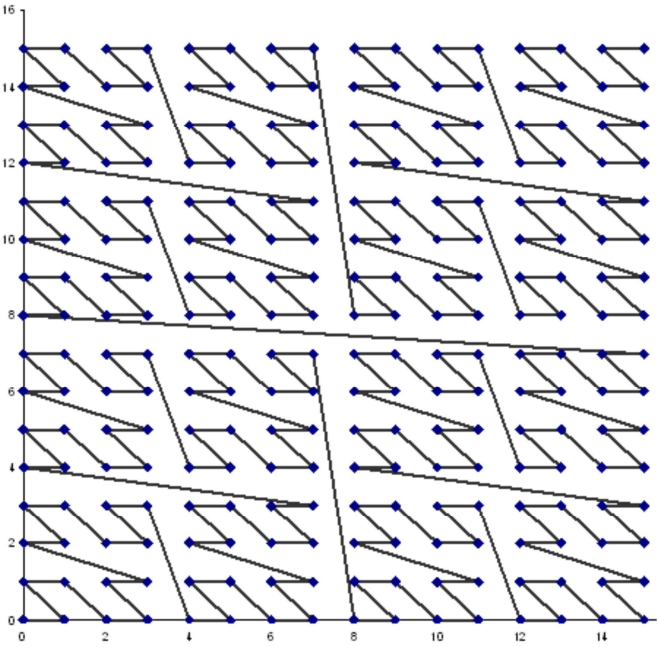


Figure 1: Z-Order Indexing (2-D).

A simple mitigation for the closeness problem is to interleave the elements at a more atomistic level — characters or digits for example. Although still biased towards the earlier dimensions, reducing the span of the dimensions improves the closeness criteria, particularly for the later dimensions. Taking this to the limit involves representing each dimensional element in binary format and interleaving the bits. This is often called Z-order indexing. Figure 1 illustrates the two-dimensional Z-order "curve" as it discretely spans an area defined by x, y ∋ Int[0, 15].

In two dimensions, the transform proceeds as follows: The pair of integers is converted into binary representations. Then the two are combined into a single number by interleaving the digits such that the lowest significant digits of each are consecutive, then the next, etc. The resultant representation can be expressed in a high base, 32 in this case, for a more compact index. As the mapping is one-to-one, this is a reversible operation. By incrementing the index by one, one steps to the next point along the curve. E.g.:

$$
\begin{aligned}
105_{10} &=> 1\ .\ 1\ .\ 0\ .\ 1\ .\ 0\ .\ 0\ .\ 1\ .{}_2 \\
70_{10} &=> .\ 1\ .\ 0\ .\ 0\ .\ 0\ .\ 1\ .\ 1\ .\ 0{}_2 \\
& 1\,1\,1\,0\,0\,0\,1\,0\,0\,1\,0\,1\,1\,0{}_2 \ => e4m_{32} \\
(105, 70) &=> e4m
\end{aligned}
\tag{1}
$$

This is easily generalized to N dimensions. Here's a four-dimensional example:

$$
\begin{aligned}
105_{10} &=> 1\ .\ .\ .\ 1\ .\ .\ .\ 0\ .\ .\ .\ 1\ .\ .\ .\ 0\ .\ .\ .\ 0\ .\ .\ .\ 1\ .\ .\ .{}_2 \\
70_{10} &=> .\ 1\ .\ .\ .\ 0\ .\ .\ .\ 0\ .\ .\ .\ 0\ .\ .\ .\ 1\ .\ .\ .\ 1\ .\ .\ .\ 0\ .\ .{}_2 \\
84_{10} &=> .\ .\ 1\ .\ .\ .\ 0\ .\ .\ .\ 1\ .\ .\ .\ 0\ .\ .\ .\ 1\ .\ .\ .\ 0\ .\ .\ .\ 0\ .{}_2 \\
99_{10} &=> .\ .\ .\ 1\ .\ .\ .\ 0\ .\ .\ .\ 0\ .\ .\ .\ 0\ .\ .\ .\ 1\ .\ .\ .\ 1\ .\ .\ .\ 0{}_2 \\
& 1\,1\,1\,1\,1\,0\,0\,0\,0\,0\,1\,0\,1\,0\,0\,0\,0\,1\,1\,1\,0\,1\,0\,1\,1\,0\,0\,0{}_2 \ => 7o51qo_{32} \\
(105, 70, 84, 99) &=> 7o51qo
\end{aligned}
\tag{2}
$$

The reversal process is literally just reading these examples from bottom to top. That is, represent the index in binary format and de-interleave the digits over the number of dimensions. Finally, reconstruct the representation of the component in the desired base, 10 in this case.

A feature of the Z-order curve is that every other step along the curve maps to a greater-than-minimum step in the n-dimensional space. In fact, the jumps at these bit boundaries can be arbitrarily large. (Hilbert curves overcome this, but at the cost of more complexity in computing the index.) This issue manifests itself when searching. Unless a search box exactly straddles bit boundaries in each dimension, the run of the Z-order curve that fully covers the search box will also contain values from outside the box, sometimes in significant numbers.

The solution is to partition the search box along the bit boundaries, starting with the highest-order boundary in each dimension. The exact solution is likely to lead to a surfeit of short-count runs of the Z-order index. Fortunately, an optimally efficient solution is generally obtained with just a couple of partitionings, especially in light of typical buffer sizes of retrieves in distributed big-data stores. (E.g. 64 MB in Hadoop.) That is, partition along the highest-order boundary for each dimension and maybe the next, retrieve based on those Z-index runs, and filter the excess retrieves with map-reduce tasks.

3 Hierarchical Indexing

The quintessential hierarchical-indexing system example is outline numbering. However, the generalized form is far more ubiquitous than might be supposed. Hierarchical implies a tiered structure, i.e. parent/child relationships. Furthermore, we also need a uniqueness condition enforced between siblings. Indexing implies that

the unique siblings are also ordered. That is, they can be sorted.

There are a few classes of sibling sets that can be distinguished, falling into two broad types. One type has a finite "step" size, but is potentially unbounded in range. I'll call this: Categories. The other type is finite in range, but has potentially limitless resolution. I'll call this: Discretely Sampled Continuum (DSC).

The Categories can be further subdivided. A Category with a finite number of elements is one. Examples of this are the set of letters or alphanumeric characters. This implies *a-priori* knowledge of the categorical span. Adding another item to the set will disturb the ordering; items on one side of the add are now a different number of steps away from those on the other side.

Allowing adds only to the end of the set leads to another subcategory, a single-sided unbounded set. An example is the set of positive integers. You may add as many items as you wish, but you may not arbitrarily sort them into the existing set. Yet another subcategory is the double-sided unbounded set, where you may add items to either end, with the same sorting restrictions. The principal distinction between these two is that the latter requires an extra bit of information for representation.

On the other hand is the DSC, where you may arbitrarily insert items between others without affecting the span between existing items. Examples include the set of fractions spanning the finite range, [0 1), or a binary tree. Table 1 summarizes these.

A hierarchical index then is just the product space of one or more of these classes. As such, it is also a form of multidimensional indexing. However, unlike the previous design, the tiering is important, that is, we want to keep the dimensional bias. As shown in the table, examples include things like the Dewey Decimal system (or floating-point numbers in general), quadtrees, dates and times, and of course, taxonomies.

In principle, separators are not needed for hybrids consisting solely of CAT_F patterns, or with at most one of CAT_1, CAT_2, or DSC. However, they are useful for readability and implicit typing.

Table 1: Hierarchical Indexing.

Rank 1			
Categories			Finite step size
CAT_F	Finite	E.g. [A .. Z]	*a-priori* knowledge (no adds)
CAT_1	Infinite (1-sided)	E.g. [1, 2, ..., ∞)	Unlimited, but unsorted adds
CAT_2	Infinite (2-sided)	E.g. (-∞ ... -2, - 1] ⊕ [0] ⊕ [1, 2, ..., ∞)	
Discretely Sampled Continuum			Finite range, e.g. [0, 1)
DSC	E.g. [0, .5, .25, .75, .125, .375, ...]		Unlimited, sorted adds (Binary and B-Trees)
Rank 2			
Hybrid			
$CAT_1 \otimes DSC$	E.g. Dewey Decimal		Unlimited range, unlimited stepping
$DSC \otimes DSC$	E.g. Quadtree		
Rank n			
Other Hybrids			
$CAT_1 \otimes CAT_F \otimes CAT_F$	E.g. Dates		2013 Sep 10
$CAT_F \otimes CAT_F \otimes CAT_F \otimes DSC$	E.g. Times		19:42:23.56
$CAT_F \otimes CAT_F \otimes CAT_F \otimes CAT_F$	E.g. IPv4 addresses		129.0.0.1
$CAT_1 \otimes CAT_1 \otimes ... \otimes CAT_1$	E.g. Outline		1.A.2.b.5
$CAT_1 \otimes CAT_1 \otimes ... \otimes CAT_1$	E.g. Taxonomy		Thing.Agent.Person.Male

4 Semantic Indexing

To create a semantic index, we need to express each element of the RDF as a hierarchical-index value and then combine the three in a multidimensional index. For this to be effective though, it is important that the tiers of the hierarchical index "line up" over all instances and across the dimensions. That is, the bit-level position of each tier should be exactly the same in each hierarchal index, which implies a known, finite length for each of those tiers. So, we are led to using CAT_F formatting for the tiers.

As previously discussed, CAT_F is formally restrictive in terms of span and sorting that, once implemented, makes it difficult to modify or extend with new information. In

practice though, we can use the time-honored tradition of choosing a large span with lots of "reserved for future use" elements between our currently defined ones.

Small, incremental changes to the taxonomy can be tolerated. For example, moving a subtree to another parent would require retrieving only those indexes related to that subtree, which, as will be shown, is efficiently feasible via this indexing scheme. Those indexes are then updated with the new subtree information and added to the index store, while the old indexes are deleted.

For illustration, one could choose 10-bit numbers represented by two base-32 characters allowing for 1024 elements for each tier. This would allow for a relatively compact index while providing reasonable flexibility to modify. Powers-of-two sizings are chosen to help searches straddle bit boundaries.

To demonstrate semantic indexing, we have fashioned a simulation. A sample taxonomy was created and used in generating a large batch of random RDFs. Table 2 shows a subset of this taxonomy. The indexing uses the illustrative 10-bit system described above. Some 300+ terms were spread over four levels, sparsely populating a tera-term structure, leaving lots of room for expansion and modification.

A modest attempt was made to group tier siblings suggestively, kind of a fuzzy tiering, at least for the upper levels. This has the effect of reducing the actual number of tiers needed. While not particularly relevant for this demonstration, this technique may be valuable for making searches across only a subsection of a tier a little more efficient.

Here, predicate-like terms are grouped to the beginning of the tree, then physical entities, and finally descriptors. Hence, the RDFs created by the simulation were not completely random, having an entity/predicate/entity structure.

The hierarchal indexes are eight, base-32 characters – four tiers of two base-32 characters. For computational purposes, the upper tiers are *right* padded to ensure tier alignment, and *left* padded internal to a tier to ensure bit alignment. E.g. 'relation' in Table 2 left pads within the tier to 01 and right pads to 01000000 for the full index.

The multidimensional index then takes the three hierarchal elements and mixes them to create a semantic index of 24 base-32 characters. Specifically, the triple is taken in PSO order, giving the predicate the slight bit-level dimensional bias, followed by the subject.

Table 2: Subset of the Sample Taxonomy.

Index	Term
01	Relation
0101	SameAs
01G1	Contains
	…
21	Action
2101	Move
210101	Walk
210141	Run
	…
2141	Use
2181	Communcate
21K1	Transact
21K1C1	TransferMoney
21K1G1	Donate
	…
	…
C1	Agent
C101	Person
C10101	Professional
C101A1	GroupLeader
C101C1	PersonOfInterest
	…
C1G1	Organization
	…
G1	Location
G141	SpaceTime
G14181	LatLonAltTime
	…
G1M1	SpotFeature
G1M161	School
G1M1C1	Business
G1M1C101	CoffeeShop
G1M1C141	RetailShop
	…
K1	Structure
K101	Building
	…
M1	Equipment
M101	Vehicle
M1O1	Weapon
M1O101	Gun
M1O141	Grenade
M1O1K1	Bomb
	…
	…
S1	Descriptor
S1K1	Size
S1O1	Shape
S1S1	Color
	…
…	

Figure 2 presents the results of the simulation. The left plot shows the distribution of the semantic-index values. For plotting purposes, just the first three base-32 digits were pulled off of the semantic index and expressed in base 10.

Although the randomization of the RDF was based on a uniform distribution, other than the previously mentioned, predicate/entity split, the taxonomy had varying numbers of children for parents and as well as for the number of tiers. Hence, the creation of random triples results in a non-uniform index distribution. Additionally, the very sparse taxonomy used here in turn results in a very sparsely filled index. But this is sufficient for the proof-of-concept demonstration.

The right part of Figure 2 breaks out a small section of the index to examine details. Here, just off to the top is an upper-level triple, Agent/Transact/Weapon. As we move towards higher indexes down the plot, we find related triples with one or more of the SPO items at lower levels. For example: Merchant/Trade/Weapon and Person/Purchase/Bomb. Only about half of the labels are displayed to avoid unreadable overlaps, but all of the triples in this segment are within the Agent/Transact/Weapon triple hierarchy. That is, no other random triple encodes to an index within this range.

It is illustrative to examine the results in more detail. For example, take the upper-level triple, Agent/Transact/Weapon. Agent, c1000000, Transact, 21k10000, and Weapon, m1o10000, encode to a PSO semantic index of: hd8007m20005000000000000.

The semantically related triples: PersonOfInterest, c101c100, Donate, 21k1g100, Grenade, m1o14100 => hd8007m200075c0007000000; Professional, c1010100, Sell, 21k14100, Mortar, m1o1c100 => hd8007m200072a0007000000; Merchant, c1012100, Purchase, 21k10100, Mine, m1o1g100 => hd8007m20007g0g007000000; and, Merchant, c1012100, Trade, 21k18100, Sword, m1o1o100 => hd8007m20007igg007000000; all group together with semantic indexes starting with hd8007m2.

"Cousin" triples that have some semantic closeness, but are outside of the Agent/Transact/Weapon hierarchy, index close, but outside the hd8007m2 range. Agent, c1000000, Purchase, 21k10100, PickupTruck, m10101g1 => hd8007420005000005g00004; Agent, c1000000, Sell, 21k14100, Sedan, m1010141 => hd80074200050200050800004; Agent, c1000000, Walk, 21010100, Weapon, m1o10000 => hd8007i00005000001000000; Agent, c1000000, Praise, 21818100, Weapon, m1o10000 => hd8007ig00050g0001000000; Person, c1010000, Manufacture, 21o10000, IED, m1o1s100 => hd8007mg0007i80004000000; MilitaryLeader, c101a1g1, Shoot, 21s1o1c1, Gun, m1o10100 => hd8007mi00075gg0078i0003.

Because the simulation used triples created randomly based on the taxonomy, many of them are nonsensical in content. However, they do group together throughout the span of the indexing based on their hierarchical positioning similar to the example shown above. The exceptions come at the major bit boundaries, which can be effectively handled by the search-partitioning method described above.

Some examples of unrelated triples: HeatWave, 61g14100, Ambush, 21s1o1k1, Contractor, c1018100 => 2co007ci00076k0007420001; Agent, c1000000, Attack, 21s1o100, Police, c1g1g100 => 3c8007ki0005kg0005000000; Interval, g1010101, Has, 01810000, Election, 618101g1 => 8900072g0007000006g00006; Country, g1q10100, Discuss, 21817100, LifeEvent, 61010000 => 8980079gg007029003000000; Polygon, u1018100, Sell, 21k14100, Election, 618101g1 => 9do007620007120007g00004 ; Truck, m1018101, Kill, 21s1o1o1, Agent, c1000000 => aco0074i00035g00034g0003; Earthquake, 61g1k100, Hijack, 21s1o181, SpaceTime, g1410000 => g4o007cq0007ck00030g0001; Bridge, g1o10100, Contains, 01g10000, Motorcycle, m1010101 => o90007d00007000006000004; Equipment, m1000000, At, 01k10000, StorageBuilding, k101m100 => ocg007420005g90004000000; Point, u101g100, Spy, 21c10100, SpaceTime, g1410000 => p4o0070q0007800003000000.

As can be seen, these triples, some of them quite whimsical, are semantically well removed from the sense of Agent/Transact/Weapon. Correspondingly, they index well removed from the hd8007m2 range. This is true, despite the fact that many of these share a term in common, such as Agent or Sell.

In comparison, consider how a concatenated-triple index would function. In an SPO triple, Agent/Purchase/PickupTruck would index closer to Agent/Praise/Weapon than Agent/Sell/Sedan, assuming a default alphabetical ordering. Multiple indexes such as PSO, SOP, etc. would be needed to maintain some level of efficiency for differing search biases.

Furthermore, without exploiting the semantic information contained in an ontological taxonomy to order the terms, you just have a collection of unrelated words. Triples starting with Agent will index far from its children: Person, Merchant, or Victim. If you wanted to find all the triples semantically related to Agent/Transact/Weapon, you would need to search across all of the combinatorics of Agent and its subtree with Transact and its subtree with Weapon and its subtree. This does not scale well with a growing ontology, let alone with a growing triplestore database.

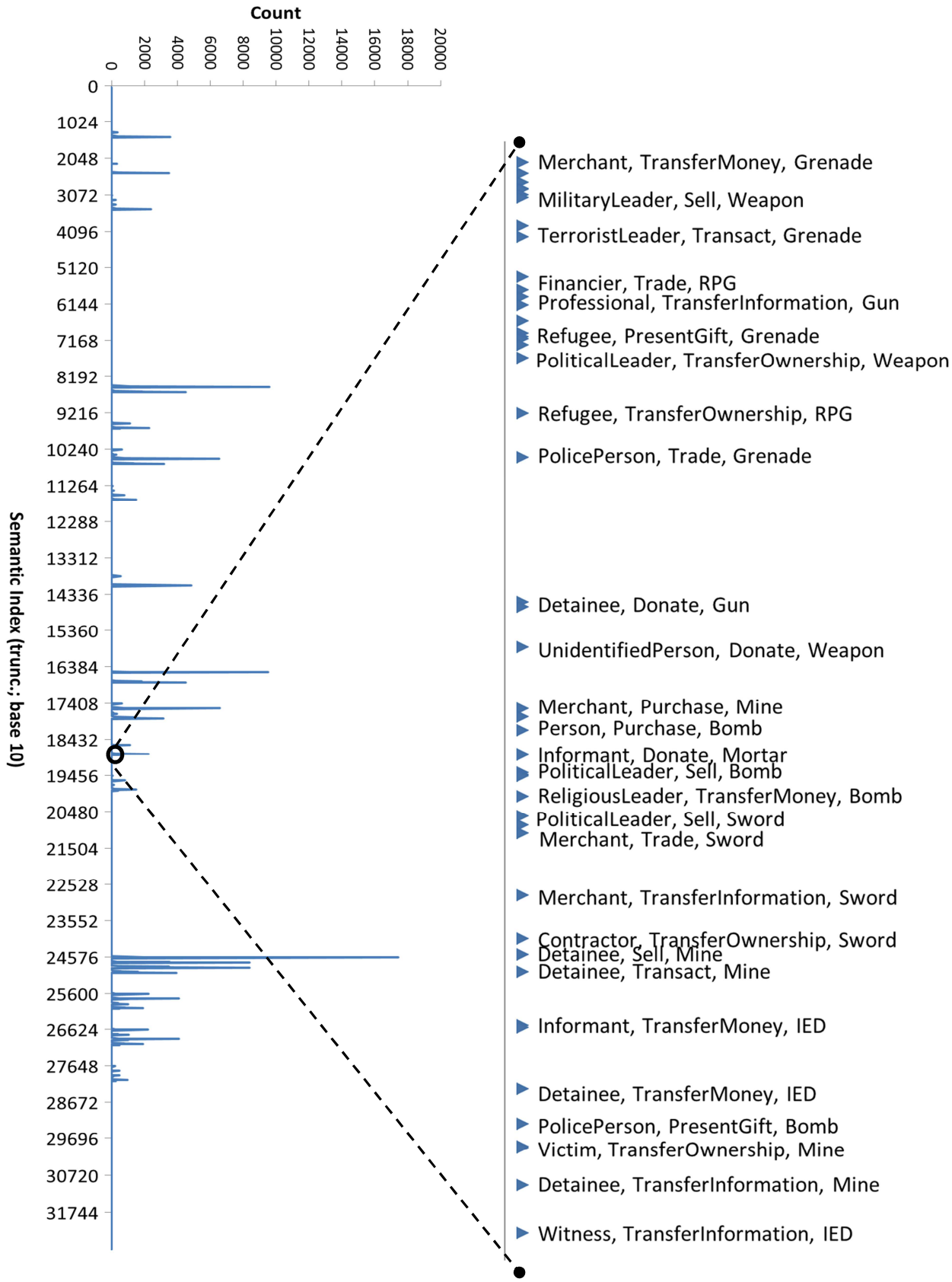


Figure 2: Overview and Detail of Semantic Index.

5 Summary and Conclusions

We have created an index to encode the semantic information contained in RDF triples. This consists of creating a hierarchical index, based on an ontology, for each element of the triple. Then these dimensions are combined using a multidimensional-indexing method.

The indexing method was exercised using a sample ontology to randomly generate RDF triples. As expected, upon detailed inspection of the indexing we find that semantically related triples encode closely.

The importance of this result is in facilitating efficient searches over large stores of triples. The "closeness" feature for semantic relatedness in the index allows for dramatically reducing the fraction of the database needed to be retrieved during a search. The inexactness of "closeness" requires some filtering of excess retrieves. But the extent of these can be mitigated with the search method and the design of the indexing.

On a speculative note, this index may be a useful foundation for an alternate or complementary semantic-hashing [6] method. Semantic hashing attempts to index a document semantically by statistically analyzing the text, typically with latent semantic analysis. Studying the distribution of the document's indexed triples may yield similarly useful results.

6 Future Work

We have demonstrated semantic indexing over RDF triples. This can be extended to cover quads and beyond that are in common use. Often temporal and/or geolocation tags are important in RDF searches. It is a simple matter to extend the above scheme to include these.

Specifically, one could create a multidimensional index for the spatial or space-time tag in addition to the RDF semantic index. Then the two can be blended together with a second multidimensional step. Searches can then proceed not only over the semantic content but also over the when and where using an efficient one-dimensional index.

Our plans include conducting benchmarking tests on big-data retrieves for both simulated and real data. Additionally, we plan to investigate details of the index design (number of dimensions and bits) and search method (number of partitions, index-run lengths, etc.) to optimize searching in generic and specific big-data stores (e.g. Accumulo, MongoDB).

7 Acknowledgement

The author would like to thank the Office of Naval Research (ONR) for financial support. We thank Ben Migliori for support with simulations and Scott McGirr for paper editing. This paper is the work of US Government employees performed in the course of employment and copyright subsists therein.

8 References

[1] Lawder, J. K., *The Application of Space-Filling Curves to the Storage and Retrieval of Multi-Dimensional Data*, Ph.D. thesis, School of Computer Science and Information Systems, Birkbeck College, University of London, 2000.

[2] Lawder, J. K. and P. J. H. King, "Using Space-Filling Curves for Multi-Dimensional Indexing", In: Lings, B. and K. Jeffrey (eds.), *Proceedings of the 17th British National Conference on Databases (BNCOD 17)*, vol. 1832, ser. *Lecture Notes in Computer Science*, pp. 20-35, Springer Verlag, July 2000.

[3] Shah, A., A. Farooq, S. Ahsan, & M. Imran, "An Indexing Technique for Web Ontologies", Journal of Computing, vol. 2, iss. 7, Jul 2010.

[4] Punnoose, R., A. Crainiceanu, and D. Rapp, "Rya: A Scalable RDF Triple Store for the Clouds", Proc. 1st Intl. Workshop on Cloud Intelligence, Aug 2012.

[5] Punnoose, R., A. Crainiceanu, and D. Rapp, "SPARQL in the Cloud using Rya", Information Systems, Jul 2013.

[6] Salakhutdinov, R., G. Hinton, "Semantic Hashing," International Journal of Approximate Reasoning, vol. 50, iss. 7, pp. 969-978, Jul 2009.

Social Event Radar (SERv2.0)：Efficient Dynamic Adjustment Mechanism of Distributed Web Crawlers in Social Networks

Tsun Ku
Department of Computer Science & Information Engineering, National Central University Taoyuan, Taiwan, R.O.C.
cujing@gmail.com

Cheng-Hung Tsai, Ping-Yen Yang, Ming-Jen Chen
Institute for Information Industry, Innovative DigiTech-Enabled Applications &Service Institute, Taipei, Taiwan, R.O.C.
{jasontsai, michaelyang, mjchen }@iii.org.tw

***Abstract*—Along with the popularity of the Internet, users can link to all kinds of social networking sites anytime and anywhere to interact and discuss with others. This phenomenon indicates that social networking sites have become a platform for interactions between companies and customers so far. This paper proposes an improved architecture of web crawlers which we proposed previously in [5]. We extend the concept of master-slave architecture of web crawlers and propose a new distributed architecture. Also we add a method of dynamic adjustment of the number of web crawlers into this architecture to efficiently promote the speed and performance of data collection. The goal of this proposed system is to collect multiple kinds of comments on the social network, aiming at those popular sites and forums to provide enterprises and users with thorough feedback data.**

Keyword— Social Network; Web Crawler; Service Module; Information Retrieval; Web Ming

I. INTRODUCTION

Recently, the flourish of information technology caused people to think how they can promote the chances of interaction with others by information technology. With the progress of information technology and the extension of the concept of interaction, virtual communities (e.g. Facebook, Weibo, Blogspot, PTT, forums, news…) had begun forming and growing. The information on the social network is one of the crucial sources from which enterprises collect customers' feedback to analyze their behavior. "People" plays a quite important role in the social network. Also, because of the connection between people, the social platforms become an enormous hub of information, scattering text comments toward every kinds of events. By the collection of these comments, it will have a notable effect on the analysis of a certain event. Therefore, we focus on how to efficiently collect these comments. The research of this paper is to propose a novel architecture of web crawlers and promote the performance of the former architecture in [5]. Then we collect data from different social networking sites(Facebook、PTT、Forum、News) by the use of five modules in the proposed architecture and carrying out multi-tasking according to the algorithm in the modules. After experiment we found that the proposed architecture can efficiently collect data from different social networking sites for future work. The method will be discussed in the following sections.

The remainder of this paper is organized as follows. Section II investigates the related works in the past. Section III presents the method and technique of the proposed architecture. The results and analysis of the implementation are drawn in Section IV. Section V is our conclusion and future work.

II. RELATED WORK

According to the estimation of the international market research authority eMarketer, the number of global users of social network has reached 1.5 billion. The number of active user of Facebook, Twitter, Weibo and Tencent were one billion, 0.5 billion and 0.3 billion at 2012, respectively. Furthermore, the penetration rate of social network services among the global users of all ages has reached 79%. Even those once considered the main composition of digital divide has a rise of penetration rate up to 9.3%. These facts show that the use of social network is very popular.

According to the statics of Institute for Information Industry[6], Taiwanese citizens are already used to searching for others' shopping experiences as a reference to their decisions before purchasing (there are 80% of internet users will browse for evaluations from others, and 74% of them will be affected by the comments.) Moreover, many customers are willing to share their own experiences with others. (52.3% of internet users are willing to do so.) For this reason, the brand owners are still focusing on conveying word-of-mouth of customers on social networking services and considering social media as an marketing channel of conveying information about their brands. The issues are therefore placed on the patterns and efficiency of word-of-mouth conveyance. The brand owners believe that they will benefit from appropriate social strategies which enhance the exposure of their products and services. As a result, according to the statics of DMA, there are more and more business owners and advertisers invest on word-of-mouth marketing[8]. In domestic market, research data indicates that the market value of social

media marketing focusing on word-of-mouth has gradually increased to 0.946 billion NTD. In comparison with that in 2011, the growth rate is 13.20%.[7] If we further predict the possible market size in 2013, there is expected to be a growth rate of 18%, which is significantly more than that of common website ads and keyword ads, and make a breakthrough in market value to 1.1 billion NTD.

III. Efficient Dynamic Adjustment Mechanism of Distributed Web Crawlers in Social Networks

This paper implements the system on the cloud service platform of Amazon, and it can be divided into five modules: DCM, TPEM, JPM, CCM, and DPPM, to collect, manage and store the data. The architecture of the system is shown in figure 3.1:

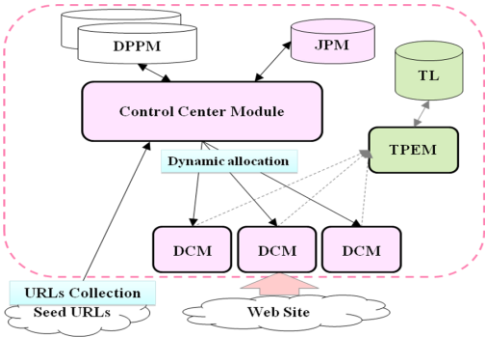


Fig 3.1. Architecture of the system

3.1 Distributed Crawler Module, DCM

■ Web Crawler Service, WCS

The DCM is in passive service mode, which is namely a web crawler service (WCS). The main work is done in the DCM. The Url_List and Job_List are dispatched through CCM and JPM for the system to collect data. The main purpose of this system is to collect the posts and comments from the social networking sites, while there are tens to hundreds of URLs. Regarding this large quantity of webpage information, we adopt the concept of depth-first search and distributed capture. When a web crawler captures an URL in the webpage, it will immediately return the link to CCM and call JPM to schedule the next task.

As shown in Fig 3.2, we abbreviate Web Crawler as WC to carry on the description below. The depth of a webpage on the social networking sites can be classified into multiple hierarchies. When JPM dispatch a task to WC_01 to do data collection, if it capture the information of URL_Lv1, WC_01 will return the information to JPM and call it to schedule a new task for WC_02. In this way, when WC_02 capture the information of URL_Lv2, it will return the information to JPM and schedule the next task.

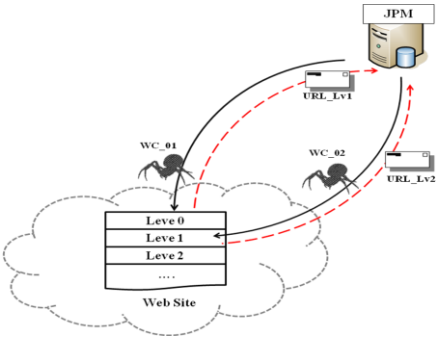


Fig 3.2, A distributed architecture of data collection in WCS

As described above, the concept of distributed cycle can significantly boost the speed of data collection and can fully load every WC to efficiently use them. The details of job scheduling in JPM will be described in section 3.3.

3.2 Template Parsing Engine Module, TPEM

■ Web Parser Service, WPS

The main work of this service is done in TPEM. By the information of URLs collected by WCS, the webpages will be parsed based on their webpage templates. As shown in table 3.1, we connect to the target websites through the internet and analyze the webpage templates to get the optimal data path.

■ Template Library, TL

The main purpose of this service is to collect the webpage information in a fast, precise way when WCS is collecting data. Therefore, how to provide the information of optimal data path to WCS will be discussed in this chapter. We propose a concept of cache which stores the information of webpage templates analyzed by WPS. The information of these webpages (e.g: authors, posts, comments, elements of webpages) will be classified and stored in the template database through the path of webpage components in the template database. This method can help the WCS collect data in a fast, precise way by the archived information.

Table 3.1. Analysis of data path on webpages

Website	URL	Path of posts	Path of comments	Path of authors	Path of components
Mobile01_HTC (Android)	http://www.mobile01.com/topiclist.php?	li[id~=post_.*].postcontainer.old > div.posthead	td.post_msg_wrap > div.post_msg >	div.jive-username-link-wrapper > a	span.prev_next > a
Mobile01_Android (Tablet)	http://www.mobile01.com/topiclist.php?f=605	table[id~=post.*].postbit > tr > td.thead:eq(0)	table > td [id~=postmessage_.*]	table.itemTable >td.author_detail	form.pagination > span > a[href~=/forum]

3.3 Job Pool Module, JPM

The JPM is an active service, which can schedule

the tasks for each web crawler according to different social networking sites and the work load of each web crawler. The goal of this service is to make each Web Crawler reach a full workload, and fully use each of them to collect data. We design essential amounts of Web Crawlers for each different social networking site and test the deviations based on the rule of thumb-anchoring and adjustment theory. Furthermore, we use the anchoring theory and the regression model proposed by [4] to adjust the scheduling and amounts of the Web Crawlers.

The research use the data of webpages on each social networking sites in January 2014 as a sample (O_{final}), then we can predict the deviations $FR_{(n)}$ based on different social networking sites and the correction $FERR_{(n)}$ at each time. The details are shown below (3-1~3-3).

$$FR_{(n)} = O_i - E(O_i) \quad (3\text{-}1)$$

$$FERR_{(n)} = O_i - O_{final} \quad (3\text{-}2)$$

$$FERR_{(n)} = \alpha_{(n)} + \varphi_{(n)} FR_{(n)} + \varepsilon \quad (3\text{-}3)$$

$FR_{(n)}$: The degree of data correction on each site. $FERR_{(n)}$: The error of data prediction on each site. n: The nth observation value before 2014/01. There will be an observation value for each month. O_i: The actual number of samples on the month of observation. $E(O_i)$: Estimation of the number of samples. $\alpha_{(n)}$ & $\varphi_{(n)}$: Parameters of the regression test. ε: The residual of the nth month. O_{final}: The final amount of observed data on the sites.

Using the regression test based on the model [4], as shown in (3-3), we can find out the margin of corrections of these social networking sites. Among the formula, the independent variables $FR_{(n)}$ are called the margin of corrections, which is the difference of predictions between the n months before the social networking sites released and the previous stage. As shown in formula 3-1, the difference between the expected and the actual amount of data on the day of observation is drawn after the estimation of margin of corrections $FR_{(n)} = O_i - E(O_i)$. Based on our assumption, the amount of observed data P_i at each phase is the predicted value at each phase, and $E(O_i)$ is the predicted value of the previous phase. Besides, the dependent variable $FERR_{(n)}$ is the predicted error, which means the difference between the expected value and the predicted value. We conduct the estimation of predicted error by formula 3-2. The amount of data in each month O_i is the predicted value at each phase, and O_{final} is the final actual value. Hence, formula 3-2 is in accordance with the context..

3.4 Control Center Module, CCM

The service of this module is in active mode. Its main job is to coordinate with JPM and TPEM, to provide with a backup mechanism, and to monitor the status of each service module and the information of each Web Crawler (e.g. : ID, IP, name of service, Port, status etc.). The details are shown in table 3.2.

Table 3.2. Information of each module

id	Name of service	IP	Port	status
01	Http_Crawler	192.148.1.2	3579	On
09	Http_Parser	192.148.1.8	3580	On
02	Http_Url	192.148.1.8	3582	On
05	Waiting	192.148.1.27	3589	Off
07	Null	192.148.1.36	3587	Miss

According to table 3.2, CCM classifies the current working status of each Web Crawler into three categories: On, Off and Miss. When the status is On, it means the Web Crawler is executing its service. If it's Off, the Web Crawler is waiting JPM for a new job. If it's Miss, the Web Crawler has lost its connection and CCM will start the backup mechanism for JPM to schedule a new job.

■ Backup mechanism

This mechanism is applied when CCM and JPM issue request packets to a Web Crawler but it does not reply to them in a certain period of time, then we define the Web Crawler has lost its connection. At this time, the backup mechanism will be turn on to resolve the problem of insufficient amount of Web Crawlers. We create a table (BM_List) in CCM to store the spare Web Crawlers, as shown in table 3.3:

Table 3.3. The list of backup web crawlers

id	Name of service	IP	Port	Status
B_01	null	null	null	Off
B_02	null	null	null	Off
B_03	null	null	null	Off

When CCM start the backup mechanism, it will provide JPM with a new Web Crawler, and JPM will schedule a new job for the new one. Then CCM will update the information of working status of the new Web Crawler, as shown in table 3.2. This paper designed the backup mechanism by this method to resolve the problem that when a Web Crawler lost its connection, and provide with a new Web Crawler to continue with the remaining job.

3.5 Data Processing Platform Module, DPPM

This module is constructed inside a MYSQL database based on the architecture designed by

ourselves. There are three hierarchies: data preprocessing, data collection, demands classification. By these functions in the three hierarchies, the collected data will be classified and stored. Further they will be processed based on their categories.

3.5.1 Data Pre-Processing

The goal of this service is to provide required information for the front-end DCM to collect and store data from the four main social networking sites (Facebook, PTT, Forum, News). The details are shown below:

- Url_List

It's for storing the name, URL, time of data collection and updating of each social networking site. Its goal is to provide WCS with the starting page of the social networking site to collect data through the paths in Site_Pattern.

- Site_Pattern

It's for storing the site pattern of each social networking site. Url_List and Site_Pattern will be transmitted to WCS through CCM for data collection. On the other hand, the amount of data collected by WCS from social networking sites can be configured at here.

- Data_table

It's for storing the collected data through DCM from each social networking site according to different sites, classifying, storing the data and modifying the time format of the data into consistency.

3.5.2 Data Collection

It's for integrating and arranging the preprocessed data collected from the four main social networking sites. The details are shown below:

- All_Data

It's for storing the collected data from the four main social networking sites. At first, it will integrate, classify the data through formula 3-4~7 based on the concept of association table, and store them into the database to make sure that the data are in correctness and relevance.

$$\bigcup_{i=1}^{n} D_{\delta} = D_1 \cup D_2 \cup D_3 \cup .... \cup D_n \quad (3\text{-}4)$$

$$\sum_{\delta=1}^{n} Da = \bigcup_{i=1}^{n} D_{\delta 1} + \bigcup_{i=1}^{n} D_{\delta 2} + ... + \bigcup_{i=1}^{n} D_{\delta n} \quad (3\text{-}5)$$

$$dom(TP) = \left\{ \sum_{\delta=1}^{n} Da1, \sum_{\delta=1}^{n} Da2, \sum_{\delta=1}^{n} Da3, ..., \sum_{\delta=1}^{n} Dan \right\} \quad (3\text{-}6)$$

$$EMP_n \subseteq dom(TP_1) \times dom(TP_2) \times ... \times dom(TP_5) \quad (3\text{-}7)$$

D_n: The data collected from each site. $\bigcup_{i=1}^{n} D_{\delta}$: Total amount of collected data from the site. $\sum_{\delta=1}^{n} Da$: Total amount of collected data from all sites. $dom(TP)$: A defined data set based on its attributes. EMP_n: The relevance of the data sets.

3.5.3 Demand Classification of Data

We conduct data extraction at the phase of data integration by the planning of demands. Through the service at this phase, there will be fifteen set of planning for social networking sites based on the categories of sources from different sites. The set of planning can be rearranged on demand to conduct further analysis and manipulations..

IV. EXPERIMENT

To verify the performance of the proposed system, in this section we will compare the method of SER v2.0 with the previous one of SER v1.0 and Nutch, which is an open source software. We plan to conduct the experiment aiming at four main social networking sites－Facebook (Fans Page), PTT, Forum and News. The content of the planned experiment is performance test and statics verification. The performance test is classified as writing speed of data and memory usage. The categories of each social networking site are shown in table 4.1:

Table 4.1. The four main social networking sites－number of categories

	Number of sites	Number of categories
Facebook	1	7045
PTT	1	227
Forum	73	1628
News	592	6465

The system is now constructed and running on the cloud service platform of Amazon. To obtain a complete performance test, the experiment environment is set at the local machine to conduct a simulation test. The details of the environment are shown in table 4.2:

Table 4.2. Details of the environment for experiment

CPU	Intel i7-3520M 2.90 GHz
Memory	16G
Hand Drive	10TB
Database	MySQL
TCP Channels	10G-bps
Number of Web Crawlers	20~300

The performance test is based on the categories of the four main social networking sites. The goal of

this test is to calculate the time of collecting data and responses from the four main social networking sites and assess the performance of each system. In this section we verify the performance of SER v2.0, SER v1.0 and Nutch by our experiment.

4.1 System performance－data collection

■ Collecting speed of small amount of data

Under the same environment, the data collection time is set at an hour and there will be twenty Web Crawlers collecting data from a social networking site. Assume that the time spent collecting each URL is one second and that there is an anti-crawler mechanism on this site; the crawlers will collect data at each site every five second. The list of target social networking sites for experiment is shown in table 4.3.

Table 4.3. List of social networking sites

	Web site	URL
Facebook	HTC	https://www.facebook.com/HTC
PTT	NTU_PTT	telnet://ptt.cc
Forum	Mobile01	http://www.mobile01.com/
News	Yahoo News	http://tw.news.yahoo.com/

Based on the above setting, we verify the speed of data collection by the amounts of data shown in figure 4.1.1 and calculate the average speed of data collection by formula 4-1.

$$V_{(av)} = \frac{D_{(tav)}}{dT} \tag{4-1}$$

$V_{(av)}$: Average speed of data collection. $D_{(tav)}$: Total average amount of collected data. dT : The difference of consumed time .

By formula 4-1 and the amount of data shown in figure 4.1.1, we calculate and find that the average speed of data collection of SER v2.0 is 4 items per second, the one of SER v1.0 is 0.2 items per second, and the one of Nutch is 3.2 items per second.

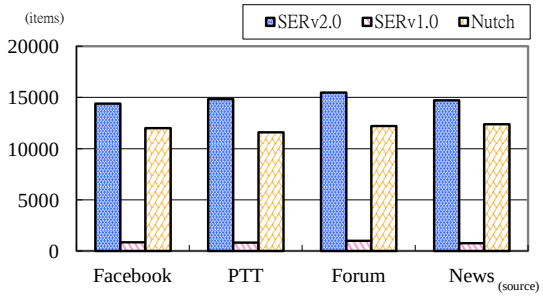


Fig 4.1.1. Amount of collected data in an hour

Through the experiment, because of the centralized architecture of SER v1.0 and the condition of 20 web crawlers on the same site, the remaining web crawlers will be idle while there is only one crawler collecting data. Hence, the average speed of SER v1.0 is 0.2 items per second. To improve the performance, we proposed a method of distributed dynamic adjustments to make each web crawler reach a full workload. The performance of SER v2.0 is higher than Nutch by 25%.

■ Collecting speed of large amount of data

In this section we limit the time of data collection from 2014/1/1 to 2014/1/31, to conduct an experiment lasting one month with 300 web crawlers on the sites, as shown in table 4.1. The goal of this experiment is to verify the speed of data collection of SER v2.0, SER v1.0 and Nutch when there is an enormous amount of data.

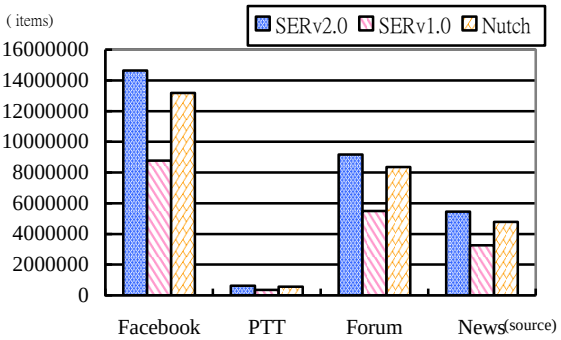


Fig 4.1.2. Amount of collected data in a month

The time of data collection is set at one month, and we conduct the data collection proposed based on the system proposed in this paper. The amount of data collected by SER v2.0 in one month from each social networking site is 14635418 items(Facebook), 619159 items(PTT), 9179003 items(Forum), and 5443267 items(News). The total amount of collected data is 29876847 items per month. The amount of data collected by SER v1.0 in one month from each social networking site is 8773845 items(Facebook), 363951 items(PTT), 5499918 items(Forum), and 3260329 items(News). The total amount of collected data is 17898043 items per month.The amount of data collected by Nutch in one month from each social networking site is 13171963 items(Facebook), 557367 items(PTT), 8357241 items(Forum), and 4776339 items(News). The total amount of collected data is 26862910 items per month.

By the statics of data collection in one month, our design and planning of the system can effectively collect a massive amount of data. The amount of collected data of SER v2.0 is higher than that of SER v1.0(11978804 items) and that of Nutch(3013937 items). The further analysis of the data in one month is shown below.

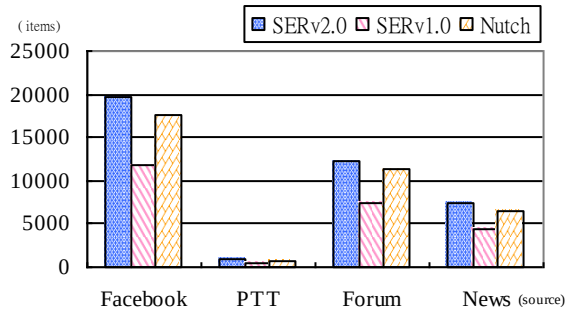


Fig 4.1.3. Average amount of collected data in an hour

This verification is the analysis of average amount of data collection in an hour, as shown in figure 4.1.3. The total average amount of data collected by SER v2.0 is 40160 items per hour, which is equal to 11.2 items per second. The total average amount of data collected by SER v1.0 is 24059 items per hour, which is equal to 6.7 items per second. The total average amount of data collected by Nutch is 36107 items per hour, which is equal to 10 items per second. According to the above statics, the performance of SER v2.0 is higher than SER v1.0 and Nutch, by 66.9% and 11.2% respectively. The key of promotion in performance is that the method of data collection in SER v2.0 is a distributed dynamic structure, and that the URLs are distributed scheduled by dynamic adjustment.

Investigating into the experiments of previous section and this section, it's noteworthy that both SER v2.0 and Nutch use distributed architecture, so the performance of them will increase when the number of web crawlers increases. The difference between them is that SER v2.0 collects URLs with a distributed architecture, which will assess the number of URLs of each social networking site, and make each web crawler reach a full workload based on dynamic job scheduling. Yet SER v1.0 uses a centralized architecture, which has a pitfall. When the number of social networking sites is smaller than that of the web crawlers, there will be some idle web crawlers and decrease the overall performance.

4.2 System performance－memory usage

In this section we will verify the usage of memory. Through the process of data collection mention in previous section, we conduct a test of memory usage, the results are shown in figure 4.2. The research analyzes the memory usage of data collection in one month, and obtains statics of SER v2.0, SER v1.0, Nutch, which is 0.371GB, 9.54GB, and 0.584GB, respectively. Because the web crawlers in SER v2.0 collect URLs in a dynamic distributed structure and return them to JPM for job scheduling, the memory usage is significantly reduced. Compared with Nutch and SER v1.0, the memory usage of SER v2.0 is less than both of them by 36.5% and 96.1%, respectively. By this experiment, we can verify that SER v2.0 will use much less memory and enhance the performance of web crawlers when conducting a data collection.

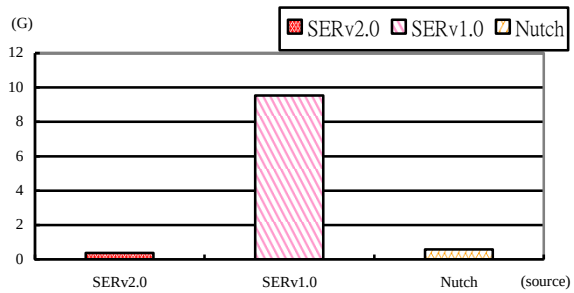


Fig 4.2. Comparison of memory usage

4.3 Verification of data

This research conducts a verification on the collected posts and responses in the database to verify the correctness of the amount of collected data. This experiment is under the same environment with the same method of data collection. The date of collected data is set at 2014/1/1 and the data is being verified manually one by one in thirty minutes. The goal of this experiment is to verify the correctness of collected data. The detail of verification is to randomly choose and number the categories of the four main social networking sites, and select one category of each site to verify its statics.

The results of random sampling are shown in table 4.4. We verify the correctness of numbers of the collected posts and comments by analyzing the results of random sampling, as shown in fig 4.3. The gray interval in the chart is the data which is not captured by our system. The statics are shown in table 4.5.

Table 4.4. Results of random sampling

Type of social network	Category of social network	URL
Facebook	JTV LIVE	https://www.facebook.com/matrixchannel
PTT	DSLR	telnet://ptt.cc
Forum	ck101	http://ck101.com/
News	CD NEWS	http://www.cdnews.com.tw/cdnews_site/

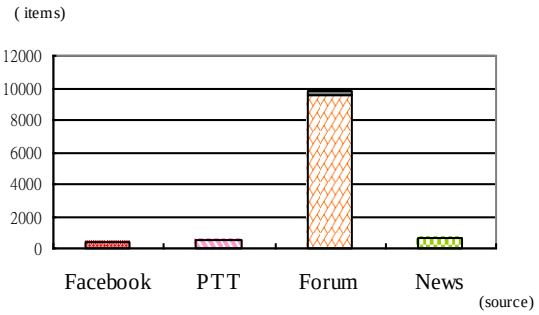


Fig 4.3. Verification of collected data

Table 4.5. Statics of collected data from the four main social networking sites

	Facebook	PTT	Forum	News
Category of social network	JTV LIVE	DSLR	ck101	CD NEWS
Amount of data collected by SERv2.0	384	531	9541	589
Correct data after verification	379	523	9524	589
The actual amount of data after verification	398	556	9753	595
Precision	98.7%	98.5%	99.8%	100%
Recall	95.2%	94.1%	97.7%	98.9%

We add a condition of one-time data (un-updated) into this research. After a manual verification of the data, we find three phenomenon below:

1.) On Facebook (Fans Page), PTT or Forum, when a moderator deletes a post or a comment, parts of them still show the deleted contents and the number of deleted ones. Even so, the proposed system will not collect the deleted posts or comments. Therefore, there will be some errors in the verification. But when we further investigate the collected data, the results are correct. 2.) In the process of data collection, when a moderator deletes the post or comment, the deleted content, which is in the database, will not show on the webpages. Because of the problem, there will be errors when conducting data verification. 3.) On Facebook (Fans Page), PTT or Forum, when there are new posts or comments, these new contents will possibly not be collected during data collection and manual verification.

By table 4.5, we find the value of precision & recall on PTT is relatively lower than that of the other categories. This is due to the difference in the number of people's discussions. Through the overall verification, the average precision of data collection is 99.3%, and the average recall rate of data collection is 96.5%.

V. CONCLUSION

This research is a project of Research on Intelligence Techniques and Service Modes of Social Media conducted by Institute for Information Industry. The proposed social event radar (SER v2.0) optimizes the hybrid of breadth-first optimal priority mechanism and depth-first recrawl mechanism, and analyzes social networking sites with the concept of anchoring effect and the dynamic distributed architecture. SER v2.0 also has dynamic adjustment of number of web crawlers, dynamic job scheduling, and backup mechanism. The performance of the proposed planning and design of system architecture can be verified by our implementation and experiments, and there is a significant increase in performance (Precision: 99.3%, Recall rate: 96.5%). Results of the experiment show that it can effectively and rapidly collect the posts and comments from the four main social networking sites, and promote the quality of social network analysis.

VI. ACKNOWLEDGEMENT

This study is conducted under the Social Intelligence Analysis Service Platform of the Institute for Information Industry which is subsidized by the Ministry of Economy Affairs of the Republic of China .

REFERENCE

[1] Mirtaheri, S.M. , Di Zou , Bochmann, GJourdan, G.-V. ,Onut, I.V. , "Dist-RIA Crawler: A Distributed Crawler for Rich Internet Applications" 3PGCIC,2013

[2] Mesbah, A. ; van Deursen, A. ; Roest, D. , "Invariant-Based Automatic Testing of Modern Web Applications", IEEE Transactions on Software Engineering, Vol.38 ,2013

[3] Jingtian Jiang ; Xinying Song ; Nenghai Yu ; Chin-Yew Lin, "FoCUS: Learning to Crawl Web Forums" , IEEE Transactions on Knowledge and Data Engineering, Vol.25 , Iss.6 ,2013

[4] Amir, E., Ganzach, Y. "Overreaction and underreaction in analysts forecasts" Journal of Economic Behavior and Organization, 37, 333–347,1998

[5] Cheng-Hung Tsai, Ping-Yen Yang,"Social Event Radar(SERv1.0)：Design and Implementation of a Web Crawlers Based in Social Networks", KC,2014

[6] Innovative DigiTech-Enabled Applications & Services Institute, Institute for Information Industry, "Report on Internet users' usage behavior in Taiwan on social network", Department of Industrial Technology, Ministry of Economic Affairs, Taipei,2012

[7] Innovative DigiTech-Enabled Applications & Services Institute, Institute for Information Industry, "Report on advertisers in Taiwan utilizing mobile and social media", Department of Industrial Technology, Ministry of Economic Affairs, Taipei, 2012

[8] Institute for Business Value, IBM, "A new marketing power under the digital trends ─ Report on global marketing officers in 2013", Department of Global Business Services, IBM, Taipei,2013

[9] Noor, T.H. ; Sheng, Q.Z. ; Alfazi, A. ; Ngu, A.H.H. ; Law, J. , "CSCE: A Crawler Engine for Cloud Services Discovery on the World Wide Web", ICWS,2013

[10] Moraes, M.C. ; Heuser, C.A. ; Moreira, V.P. ; Barbosa, D. , "Prequery Discovery of Domain-Specific Query Forms: A Survey", IEEE Transactions on Knowledge and Data Engineering, Vol.25 , Iss.8 , 2013

[11] Jingtian Jiang ; Xinying Song ; Nenghai Yu ; Chin-Yew Lin, "FoCUS: Learning to Crawl Web Forums" , IEEE Transactions on Knowledge and Data Engineering, Vol.25 , Iss.6 , 2013

Finding Local and Periodic Association Rules from Fuzzy Temporal Data

F. A. Mazarbhuiya, M. Shenify, Md. Husamuddin
College of Computer Science and IT
Albaha University, Albaha, KSA
fokrul_2005@yahoo.com
mshenify@yahoo.com
mdhusamuddin@gmail.com

Abstract. *The problem of finding association rules from a dataset is to find all possible associations that hold among the items, given a minimum support value and a minimum confidence. This involves finding frequent sets first and then the association rules that hold within the items in the frequent sets. The problem of mining temporal association rules from temporal dataset is to find association rules between items that hold within certain time intervals but not throughout the dataset. This involves finding frequent sets that are frequent at certain time intervals and then association rules among the items present in the frequent sets. In some of the applications the time of transaction is imprecise; we call the associated dataset as fuzzy temporal dataset. In such datasets, we may find set of items that are frequent in certain fuzzy time intervals. We call these as locally frequent sets over fuzzy time intervals and the associated association rules as local association rule over fuzzy time intervals. These association rules cannot be discovered in the usual way because of fuzziness involved in temporal features. Normally these association rules are periodic in nature. We call such rules as periodic association rules over fuzzy time interval. We propose modification to the A-priori algorithm to compute locally frequent sets and to extract periodic frequent sets and periodic association rules from fuzzy temporal data.*

Kew-words: Core of a fuzzy number, Data mining, frequent sets, fuzzy membership function, α-cut.

1 Introduction

The problem of mining association rules has been defined initially [15] by R. Agarwal *et al* for application in large super markets. Large supermarkets have large collection of records of daily sales. Analyzing the buying patterns of the buyers will help in taking typical business decisions such as what to put on sale, how to put the materials on the shelves, how to plan for future purchase etc.

Mining for association rules between items in temporal databases has been described as an important data-mining problem. Transaction data are normally temporal. The market basket transaction is an example of this type.

In this paper we consider datasets, which are fuzzy temporal i.e. the time in which a transaction has taken place is imprecise or approximate and is attached to the transactions. In large volumes of such data, some hidden information or relation ship among the items may be there which cannot be extracted because of some fuzziness in the temporal features. Also the case may be that some association rules may hold in certain fuzzy time period but not throughout the dataset. For finding such association rules we need to find itemsets that are frequent at certain time period, which will obviously be imprecise due to the fact that the time of each transaction is fuzzy. We call such frequent sets locally frequent over fuzzy time interval. From these locally frequent sets, associations among the items in these sets can be obtained. Since a periodic nature is there in any natural event this kind of associations normally hold periodically. And if such locally frequent sets also have the property that they become frequent in certain fuzzy time intervals i.e.

they are periodic in nature then we call these sets periodic frequent sets and the associated associations as periodic association rules over fuzzy time interval.

In section 2 we give a brief discussion on the recent works in Temporal Data Mining and fuzzy temporal data mining. In section 3 we describe the terms and notations used in this paper. In section 4, we give the algorithm proposed in this paper for mining locally frequent sets over fuzzy time interval and local association rules over same. In section 5, we discuss about periodic association rule over fuzzy time interval. We conclude with conclusion and lines for future work in section 6.

2 Recent works

The problem of discovery of association rules was first formulated by Agrawal *et al* in 1993. Given a set I, of items and a large collection D of transactions involving the items, the problem is to find relationships among the items i.e. the presence of various items in the transactions. A transaction t is said to support an item if that item is present in t. A transaction t is said to support an itemset if t supports each of the items present in the itemset. An association rule is an expression of the form $X \Rightarrow Y$ where X and Y are subsets of the itemset I. The rule holds with confidence τ if τ% of the transaction in D that supports X also supports Y. The rule has support σ if σ% of the transactions supports $X \cup Y$. A method for the discovery of association rules was given in [15], which is known as the A priori algorithm. This was then followed by subsequent refinements, generalizations, extensions and improvements. As the number of association rules generated is too large, attempts were made to extract the useful rules ([13], [16]) from the large set of discovered association rules. Attempts are also made to make the process of discovery of rules faster ([12], [14]). Generalized association rules ([9], [17]) and Quantitative association rules ([18]) were later on defined and algorithms were developed for the discovery of these rules. A hashed based technique is used in [11] to improve the rule mining process of the A priori algorithm.

Temporal Data Mining is now an important extension of conventional data mining and has recently been able to attract more people to work in this area. By taking into account the time aspect, more interesting patterns that are time dependent can be extracted. There are mainly two broad directions of temporal data mining [7]. One concerns the discovery of causal relationships among temporally oriented events. Ordered events form sequences and the cause of an event always occur before it. The other concerns the discovery of similar patterns within the same time sequence or among different time sequences. The underlying problem is to find frequent sequential patterns in the temporal databases. The name sequence mining is normally used for the underlying problem. In [8] the problem of recognizing frequent episodes in an event sequence is discussed where an episode is defined as a collection of events that occur during time intervals of a specific size.

The association rule discovery process is also extended to incorporate temporal aspects. In temporal association rules each rule has associated with it a time interval in which the rule holds. The problems associated are to find valid time periods during which association rules hold, the discovery of possible periodicities that association rules have and the discovery of association rules with temporal features. In [10], [19], [20] and [21], the problem of temporal data mining is addressed and techniques and algorithms have been developed for this. In [10] an algorithm for the discovery of temporal association rules is described. In [2], two algorithms are proposed for the discovery of temporal rules that display regular cyclic variations where the time interval is specified by user to divide the data into disjoint segments like months, weeks, days etc. Similar works were done in [6] and [22] incorporating multiple granularities of time intervals (e.g. first working day of every month) from which both cyclic and user defined calendar patterns can be achieved. In [1], the method of finding locally and periodically frequent sets and periodic association rules are discussed which is an improvement of other methods in the sense that it dynamically extract all the rules along with the intervals where the rules hold. In ([23], [24]) fuzzy calendric data mining and fuzzy temporal data mining is discussed where user specified ill-defined

fuzzy temporal and calendric patterns are extracted from temporal data.

Our approach is different from the above approaches. We are considering the fact that the time of transactions are not precise rather they are fuzzy numbers and some items are seasonal or appear frequently in the transactions for certain ill-defined periods only i.e. summer, winter, etc. They appear in the transactions for a short time and then disappear for a long time. After this they may again reappear for a certain period and this process may repeat. For these itemsets the support cannot be calculated in the usual way ([1], [10]), it has to be computed by the method defined in section 3.2. These items may lead to interesting association rules over fuzzy time intervals. In this paper we calculate the support values of these sets locally in a α- cut of a fuzzy time interval where a fuzzy time interval represents a particular season in which the itemset is appearing frequently and if they are frequent in the fuzzy time interval under consideration then we call these sets locally frequent sets over that fuzzy time interval. The large fuzzy time gap in which they do not appear is not counted. We also define periodic frequent sets and periodic association rules over fuzzy time intervals. As mentioned in the previous paragraph similarly works were also done in [23], [24] but in non-fuzzy temporal data. But in all these methods they discuss the association rule mining of non-fuzzy temporal data. Our approach although little bit similar to the work of [1], is different from others in the sense that it discovers association rules from fuzzy temporal data and finds the association rules along with their fuzzy time intervals over which the rules hold automatically.

3 Terms, Notations and Symbols used

3.1 Some Definitions related to Fuzziness

Let E be the universe of discourse. A fuzzy set A in E is characterized by a membership function $A(x)$ lying in [0,1]. $A(x)$ for $x \in E$ represents the grade of membership of x in A. Thus a fuzzy set A is defined as

$$A=\{(x, A(x)), x \in E\}$$

A fuzzy set A is said to be normal if $A(x) = 1$ for at least one $x \in$ E

An α-cut of a fuzzy set is an ordinary set of elements with membership grade greater than or equal to a threshold α, $0 \le \alpha \le 1$. Thus a α-cut A_α of a fuzzy set A is characterized by

$$A_\alpha=\{x \in E; A(x) \ge \alpha\}$$ [see *e.g.* [4]]

A fuzzy set is said to be convex if all its α-cuts are convex sets [see *e.g.* [5]].

A fuzzy number is a convex normalized fuzzy set A defined on the real line R such that

1. there exists an $x_0 \in R$ such that $A(x_0) = 1$, and
2. $A(x)$ is piecewise continuous.

A fuzzy number is denoted by $[a, b, c]$ with $a < b < c$ where $A(a) = A(c) = 0$ and $A(b) = 1$. $A(x)$ for all $x \in [a, b]$ is known as left reference function and $A(x)$ for $x \in [b, c]$ is known as the right reference function. Thus a fuzzy number can be thought of as containing the real numbers within some interval to varying degrees. The α-cut of the fuzzy number $[t_1\text{-}a, t_1, t_1+a]$ is a closed interval $[t_1+(\alpha\text{-}1).a, t_1+(1\text{-}\alpha).a]$.

Fuzzy intervals are special fuzzy numbers satisfying the following.

1. there exists an interval $[a, b] \subset R$ such that $A(x_0) = 1$ for all $x_0 \in [a, b]$, and
2. $A(x)$ is piecewise continuous.

A fuzzy interval can be thought of as a fuzzy number with a flat region. A fuzzy interval A is denoted by $A = [a, b, c, d]$ with $a < b < c < d$ where $A(a) = A(d) = 0$ and $A(x) = 1$ for all $x \in [b, c]$. $A(x)$ for all $x \in [a, b]$ is known as left reference function and $A(x)$ for $x \in [c, d]$ is known as the right reference function. The left reference function is non-decreasing and the right reference function is non-increasing [see *e.g.* [3]].

Similarly the α-cut of the fuzzy interval $[t_1\text{-}a, t_1, t_2, t_2+a]$ is a closed interval $[t_1+(\alpha\text{-}1).a, t_2+(1\text{-}\alpha).a]$.

The core of a fuzzy number A is the set of elements of A having membership value one i.e.

$$\text{Core}(A) = \{(x, A(x); A(x) = 1\}$$

For every fuzzy set A,

$$A = \bigcup_{\alpha \in [0,1]} {}_{\alpha}A$$

where ${}_{\alpha}A(x) = \alpha.\ {}_{\alpha}A(x)$, and ${}_{\alpha}A$ is a special fuzzy set [4]

For any two fuzzy sets A and B and for all $\alpha \in [0, 1]$,

i) ${}_{\alpha}(A \cup B) = {}_{\alpha}A \cup {}_{\alpha}B$

ii) ${}_{\alpha}(A \cap B) = {}_{\alpha}A \cap {}_{\alpha}B$

For any two fuzzy numbers A and B, we say the membership functions $A(x)$ and $B(x)$ are similar to each other if the slope of the left reference function of $A(x)$ is equal to the that of $B(x)$ and the slope of right reference of $A(x)$ is equal that of $B(x)$. Obviously for any two fuzzy numbers A and B having similar membership functions

$$\left| {}_{\alpha}A \right| = \left| {}_{\alpha}B \right|, \forall \alpha \in [0, 1]$$

3.2 Some Definitions related to Association Rule Mining over Fuzzy time period

Let T = $<t_o, t_1, \ldots\ldots\ldots>$ be a sequence of imprecise or fuzzy time stamps over which a linear ordering < is defined where $t_i < t_j$ means t_i denotes the core of a fuzzy time which is earlier than the core of another fuzzy time stamp t_j. For the sake of convenience, we assume that all the fuzzy time stamps are having similar membership functions. Let I denote a finite set of items and the transaction database D is a collection of transactions where each transaction has a part which is a subset of the itemset I and the other part is a fuzzy time-stamp indicating the approximate time in which the transaction had taken place. We assume that D is ordered in the ascending order of the core of fuzzy time stamps. For fuzzy time intervals we always consider a fuzzy closed intervals of the form $[t_1\text{-}a\ t_1, t_2, t_2+a]$ for some real number a. We say that a transaction is in the fuzzy time interval $[t_1\text{-}a, t_1, t_2, t_2+a]$ if the α-cut of the fuzzy time stamp of the transaction is contained in α-cut of $[t_1\text{-}a, t_1, t_2, t_2+a]$ for some user's specified value of α.

We define the local support of an itemset in a fuzzy time interval $[t_1\text{-}a, t_1, t_2, t_2+a]$ as the ratio of the number of transactions in the time interval $[t_1+(\alpha\text{-}1).a, t_2+(1\text{-}\alpha).a]$ containing the itemset to the total number of transactions in $[t_1+(\alpha\text{-}1).a, t_2+(1\text{-}\alpha).a]$ for the whole data base D for a given value of α. We use the notation $Sup_{[t_1-a,t_1,t_2,t_2+a]}(X)$ to denote the support of the itemset X in the fuzzy time interval $[t_1\text{-}a, t_1, t_2, t_2+a]$. Given a threshold σ we say that an itemset X is frequent in the fuzzy time interval $[t_1\text{-}a, t_1, t_2, t_2+a]$ if $Sup_{[t_1-a,t_1,t_2,t_2+a]}(X) \geq (\sigma/100)$* tc where tc denotes the total number of transactions in D that are in the fuzzy time interval $[t_1\text{-}a, t_1, t_2, t_2+a]$. We say that an association rule $X \Rightarrow Y$, where X and Y are item sets holds in the time interval $[t_1\text{-}a, t_1, t_2, t_2+a]$ if and only if given threshold τ,

$Sup_{[t_1-a,t_1,t_2,t_2+a]}(X \cup Y) / Sup_{[t_1-a,t_1,t_2,t_2+a]}(X) \geq \tau / 100.0$

and $X \cup Y$ is frequent in $[t_1\text{-}a, t_1, t_2, t_2+a]$. In this case we say that the confidence of the rule is τ.

For each locally frequent item set we keep a list of fuzzy time intervals in which the set is frequent where each fuzzy interval is represented as [*start-a, start, end, end+a*] where *start* gives the approximate starting time of the time interval and *end* gives the approximate ending time of the time-interval. *end – start* gives the length of the core of the fuzzy time interval. For a given value of α of two intervals [$start_1$-a, $start_1$, end_1, end_1+a] and [$start_2$-a, $start_2$, end_2, end_2+a] are non-overlapping if their α-cuts are non-overlapping.

4 Algorithm proposed:

4.1 Generating Locally Frequent Sets

While constructing locally frequent sets, with each locally frequent set a list of fuzzy time-intervals is maintained in which the set is frequent. Two user's specified thresholds α and *minthd* are used for this. During the execution of the algorithm while making a pass through the database, if for a particular itemset the α-cut of its current fuzzy time-stamp, [${}_{\alpha}Lcurrent$, ${}_{\alpha}Rcurrent$] and the α-cut, [${}_{\alpha}Llastseen$, ${}_{\alpha}Rlastseen$] of its fuzzy time, when it was last seen overlap then the current transaction is included in the current time-interval under consideration which is extended with replacement of ${}_{\alpha}Rlastseen$ by ${}_{\alpha}Rcurrent$; otherwise a new time-interval is started with ${}_{\alpha}Lcurrent$ as the starting point. The support count of the item set in the previous time interval is checked to see whether it is frequent in that interval

or not and if it is so then it is fuzzified and added to the list maintained for that set. Also for the locally frequent sets over fuzzy time intervals, a minimum core length of the fuzzy period is given by the user as *minthd* and fuzzy time intervals of core length greater than or equal to this value are only kept. If *minthd* is not used than an item appearing once in the whole database will also become locally frequent a over fuzzy point of time.

Procedure to compute L_1, the set of all locally frequent item sets of size 1.

For each item while going through the database we always keeps an α-cut $^{\alpha}$*lastseen* which is [$^{\alpha}$*Llastseen, *$^{\alpha}$*Rlastseen*] that corresponds to the fuzzy time stamp when the item was last seen. When an item is found in a transaction and the fuzzy time-stamp is *tm* and if its α-cut $^{\alpha}$*tm*=[$^{\alpha}$*Ltm, *$^{\alpha}$*Rtm*] has empty intersection with [$^{\alpha}$*Llastseen, *$^{\alpha}$*Rlastseen*], then a new time interval is started by setting *start* of the new time interval as $^{\alpha}$*Ltm* and *end* of the previous time interval as $^{\alpha}$*Rlastseen*. The previous time interval is fuzzified provided the support of the item is greater than *min-sup*. The fuzzified interval is then added to the list maintained for that item provided that the duration of the core is greater than *minthd*. Otherwise $^{\alpha}$*Rlastseen* is set to $^{\alpha}$*Rtm*, the counters maintained for counting transactions are increased appropriately and the process is continued.

Following is the algorithm to compute L_1, the list of locally frequent sets of size 1. Suppose the number of items in the dataset under consideration is n and we assume an ordering among the items.

Algorithm 4.1

C_1 = {(i_k,tp[k]) : k = 1,2,.....,n}
where i_k is the k-th item and tp[k] points to a list of fuzzy time intervals initially empty.}
for k = 1 to n do
set $^{\alpha}$lastseen[k]=ϕ;
set itemcount[k]and transcount[k] to zero for each transaction t in the database with fuzzy time stamp tm
do
{for k = 1 to n do
{ if {i_k} $\subseteq$ t then
{ if($^{\alpha}$lastseen[k] == ϕ)
{$^{\alpha}$lastseen[k] = $^{\alpha}$firstseen[k] = $^{\alpha}$tm;
itemcount[k] = transcount[k] = 1;
}
else
if([$^{\alpha}$Llastseen[k],$^{\alpha}$Rlastseen[k]]$\cap$ [$^{\alpha}$Ltm[k], $^{\alpha}$Rtm[k]]$\neq\phi$)
{$^{\alpha}$Rlastseen[k]=$^{\alpha}$Rtm[k]; itemcount[k]++;
transcount[k]++;
}
else
*{ if (itemcount[k]/transcount[k]*100 $\geq$ σ)*
fuzzify([$^{\alpha}$Llastseen[k],$^{\alpha}$Rlastseen[k]],$\forall\alpha\in$[0, 1])
if(|core(fuzzified interval)|$\geq$ minthd)
add(fuzzified interval) to tp[k];
itemcount[k] = transcount[k] = 1;
lastseen[k] = firstseen[k] = tm;
}
}
else transcount[k]++;
} // end of k-loop //
} // end of do loop //
for k = 1 to n do
*{ if (itemcount[k]/transcount[k]*100$\geq\sigma$)*
fuzzify([$^{\alpha}$Llastseen[k], $^{\alpha}$Rlastseen[k]], $\forall\alpha\in$[0, 1])
if(|core(fuzzified interval)|$\geq$ minthd)
add(fuzzified interval) to tp[k];
if(tp[k] != 0) add {i_k, tp[k]} to L_1
}
fuzzify([$^{\alpha}$a,$^{\alpha}$b], α)
{ fuzzified interval= $\bigcup_{\alpha\in[0,1]}{}_{\alpha}[a,b]$;
where $_{\alpha}$[a, b](x) = α. $^{\alpha}$[a, b](x)
return(fuzzified interval)
}

Two support counts are kept, *itemcount* and *transcount*. If the count percentage of an item in an α-cut of a fuzzy time interval is greater than the minimum threshold then only the set is considered as a locally frequent set over fuzzy time interval.

L_1 as computed above will contain all 1-sized locally frequent sets over fuzzy time intervals and with each set there is associated an ordered list of fuzzy time intervals in which the set is frequent. Then A priori candidate generation algorithm is

used to find candidate frequent set of size 2. With each candidate frequent set of size two we associate a list of fuzzy time intervals that are obtained in the pruning phase. In the generation phase this list is empty. If all subsets of a candidate set are found in the previous level then this set is constructed. The process is that when the first subset appearing in the previous level is found then that list is taken as the list of fuzzy time intervals associated with the set. When subsequent subsets are found then the list is reconstructed by taking all possible pair wise intersection of subsets one from each list. Sets for which this list is empty are further pruned.

Using this concept we describe below the modified A-priori algorithm for the problem under consideration.

Algorithm 4.2

Modified A priori

Initialize

k = 1;

C_1 = all item sets of size 1

L_1 = {frequent item sets of size 1 where with each itemset $\{i_k\}$ a list tp[k] is maintained which gives all time fuzzy intervals in which the set is frequent}

*L_1 is computed using algorithm 1.1 */*

for(k = 2; $L_{k-1} \neq \phi$; k++) do

{ C_k = apriorigen(L_{k-1})

/ same as the candidate generation method of the A priori algorithm setting tp[i] to zero for all i*/*

prune(C_k);

drop all lists of fuzzy time intervals maintained with the sets in C_k

Compute L_k from C_k.

//L_k can be computed from C_k using the same procedure used for computing L_1 //

k = k + 1

}

$$Answer = \bigcup_k L_k$$

Prune(C_k)

{Let m be the number of sets in C_k and let the sets be $s_1, s_2, \ldots, s_m$. Initialize the pointers tp[i] pointing to the list of fuzzy time-intervals maintained with each set s_i to null

for i = 1 to m do

{for each (k-1) subset d of s_i do

{if d $\notin L_{k-1}$ then

{C_k = C_k - {s_i, tp[i]}; break;}

else

{ if (tp[i] == null) then set tp[i] to point to the list of fuzzy time intervals maintained for d

else

{ take all possible pair-wise intersection of fuzzy time intervals one from each list,one list maintained with tp[i] and the other maintained with d and take this as the list for tp[i]

delete all fuzzy time intervals whose core length is less than the value of minthd if tp[i] is empty then {C_k = C_k - {s_i,tp[i]};

break;

}

}

}

}

}

}

4.2 Generating Association Rules

If an itemset is frequent in a fuzzy time-interval [t_1-a, t_1, t_2, t_2+a] then all its subsets are also frequent in the fuzzy time-interval [t_1-a, t_1, t_2, t_2+a]. But to generate the association rules as defined in section 3, we need the supports of the subsets in fuzzy time-interval [t_1-a, t_1, t_2, t_2+a], which may not be available after application of the algorithm as defined in 4.1. For this one more scan of the whole database will be needed. For each association rule we attach a fuzzy interval in which the association rule holds.

5 Extracting Periodic Patterns

In the algorithm proposed in section 4 for each locally frequent set, a list of fuzzy time intervals is maintained in which the set is frequent. In this list if we find that the distance between the cores of fuzzy intervals is almost equal (up to a small variation) then we call these frequent sets periodic frequent sets over fuzzy time intervals. Association rules that hold periodically over fuzzy time period are called periodic association rules over fuzzy time interval.

6 Conclusion and Lines for future work

An algorithm for finding frequent sets that are frequent in certain fuzzy time periods from fuzzy temporal data, in the paper is given. The algorithm dynamically computes the frequent sets along with their fuzzy time intervals where the sets are frequent. These frequent sets are named as locally frequent sets over fuzzy time interval. The technique used is similar to the A priori algorithm. From these locally frequent sets interesting rules may follow. Some of these locally frequent sets may be periodic in nature. Then we call these sets periodic frequent sets over fuzzy time interval and the associated rules periodic association rule over fuzzy time interval.
In the level-wise generation of locally frequent sets, for each locally frequent set we keep a list of all fuzzy time-intervals in which it is frequent. For generating candidates for the next level, pair-wise intersection of the intervals in two lists are taken. The same algorithm can be implemented with both real life as well as synthetic datasets.

REFERRENCES

[1] A. K. Mahanta, F. A. Mazarbhuiya and H. K. Baruah; Finding Locally and Periodically Frequent Sets and Periodic Association Rules, Proceeding of 1st Int'l Conf on Pattern Recognition and Machine Intelligence (PreMI'05),LNCS 3776 (2005), 576-582.

[2] B. Ozden, S. Ramaswamy and A. Silberschatz; Cyclic Association Rules, Proc. of the 14th Int'l Conference on Data Engineering, USA (1998), 412-421.

[3] D. Dubois and H. Prade; Ranking fuzzy numbers in the setting of possibility theory, *Information Science* 30(1983), 183-224.

[4] G. J. Klir and B. Yuan; Fuzzy Sets and Fuzzy Logic Theory and Applications, Prentice Hall India Pvt. Ltd.(2002).

[5] G. Q. Chen, S. C. Lee and E. S. H. Yu; Application of fuzzy set theory to economics, In: Wang, P. P., ed., Advances in Fuzzy Sets, Possibility Theory and Applications, Plenum Press, N. Y.(1983), 277-305.

[6] G. Zimbrao, J. Moreira de Souza, V. Teixeira de Almeida and W. Araujo da Silva; An Algorithm to Discover Calendar-based Temporal Association Rules with Item's Lifespan Restriction, Proc. of the 8th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining (2002) Canada, 2nd Workshop on Temporal Data Mining, v. 8 (2002), 701-706.

[7] J. F. Roddick, M. Spillopoulou; A Biblography of Temporal, Spatial and Spatio-Temporal Data Mining Research; ACM SIGKDD (June 1999).

[8] H. Manilla, H. Toivonen and I. Verkamo; Discovering frequent episodes in sequences; KDD'95; AAAI, 210-215 (August 1995).

[9] J. Hipp, A. Myka, R. Wirth and U. Guntzer; A new algorithm for faster mining of generalized association rules; Proceedings of the 2nd European Symposium on Principles of Data Mining and Knowledge Discovery (PKDD '98), Nantes, France (September 1998).

[10] J. M. Ale and G.H. Rossi; An approach to discovering temporal association rules; Proceedings of the 2000 ACM symposium on Applied Computing (March 2000).

[11] J. S. Park, M. S. Chen and P. S. Yu; An Effective Hashed Based Algorithm for Mining Association Rules; Proceedings of ACM SIGMOD (1995), 175-186.

[12] M. J. Zaki, S. Parthasarathy, M. Ogihara and W. Li; New algorithms for the fast discovery of association rules; Proceedings of the 3rd International Conference on KDD and data mining (KDD '97), Newport Beach, California (August 1997).

[13] M. Klemettinen, H. Manilla, P. Ronkainen, H. Toivonen and A. I. Verkamo; Finding interesting rules from large sets of discovered association rules; Proceedings of the 3RD international Conference on Information and Knowledge Management, Gathersburg, Maryland (29 Nov 1994).

[14] R. Agrawal and R. Srikant; Fast algorithms for mining association rules, Proceedings of the 20th International Conference on Very Large Databases (VLDB '94), Santiago, Chile (June 1994).

[15] R. Agrawal, T. Imielinski and A. Swami; Mining association rules between sets of items in large databases; Proceedings of the ACM SIGMOD '93, Washington, USA (May 1993).

[16] R. Motwani, E. Cohen, M. Datar, S. Fujiware, A. Gionis, P. Indyk, J. D. Ullman and C. Yang; Finding interesting association rules without support pruning, Proceedings of the16th International Conference on Data Engineering (ICDE), IEEE, (2000).

[17] R. Srikant and R. Agrawal; Mining generalized association rules, Proceedings of the 21st Conference on very large databases (VLDB '95), Zurich, Switzerland (September 1995).

[18] R. Srikant and R. Agrawal; Mining quantitative association rules in large relational tables, Proceedings of the 1996 ACM SIGMOD Conference on management of data, Montreal, Canada (June 1996).

[19] X. Chen and I. Petrounias; A framework for Temporal Data Mining; Proceedings of the 9th International Conference on Databases and Expert Systems Applications, DEXA '98, Vienna, Austria. Springer-Verlag, Berlin; Lecture Notes in Computer Science 1460 (1998), 796-805

[20] X. Chen and I. Petrounias; Language support for Temporal Data Mining; Proceedings of 2nd European Symposium on Principles of Data Mining and Knowledge Discovery, PKDD '98, Springer Verlag, Berlin (1998), 282-290

[21] X. Chen, I. Petrounias and H. Healthfield; Discovering temporal Association rules in temporal databases; Proceedings of IADT'98 (International Workshop on Issues and Applications of Database Technology (1998), 312-319

[22] Y. Li, P. Ning, X. S. Wang and S. Jajodia; Discovering Calendar-based Temporal Association Rules, In Proc. of the 8th Int'l Symposium on Temporal Representation and Reasonong (2001)

[23] W. J. Lee and S. J. Lee; Discovery of Fuzzy Temporal Association Rules, IEEE Transactions on Systems, Man and Cybenetics-part B; Cybernetics, Vol 34, No. 6 (Dec 2004), 2330-2341.

[24] W. J. Lee and S. J. Lee; Fuzzy Calendar Algebra and It's Applications to Data Mining, Proceedings of the 11th International Symposium on Temporal Representation and Reasoning (TIME'04), IEEE, 2004.

Exploring big-data analysis using integrative systems biology approaches for kidney renal clear cell carcinoma studies

William Yang and Kenji Yoshigoe
Department of Computer Science, University of Arkansas Little Rock College of Engineering and Information Technology, Little Rock, Arkansas 72204 USA
wxyang1@ualr.edu and kxyoshigoe@ualr.edu

Andrzej Niemierko and Jack Y. Yang
Division of Biostatistics and Biomathematics, Department of Radiation Oncology, Massachusetts General Hospital and Harvard Medical School, Boston, MA 02140 USA
aniemieko@partners.org & jyang@hadron.mgh.harvard.edu

Xiang Qin
Human Genome Sequencing Center
Baylor College of Medicine
Houston, Texas 77345 USA
xqin@bcm.edu

Yunlong Liu
Center for Computational Biology and Bioinformatics
Indiana University School of Medicine
Indianapolis, Indiana 46202 USA
yunliu@indiana.edu

Jun S. Liu
Department of Statistics, Harvard University,
Cambridge, MA 02138, USA
jliu@stat.harvard.edu

Zhongxue Chen
Department of Epidemiology and Biostatistics,
Indiana University School of Public Health,
Bloomington, Indiana 47405 USA ZC3@indiana.edu

A Keith Dunker
Center for Computational Biology and Bioinformatics
Indiana University School of Medicine
Indianapolis, Indiana 46202 USA
kedunker@indiana.edu

Liangjiang Wang
Department of Genetics and Biochemistry
Clemson University,
Clemson University, SC 29634
liangjw@clemson.edu

Youping Deng
Rush University Cancer Center,
Rush University, Chicago, IL 60612, USA
youping.deng@rush.edu

Dong Xu
Department of Computer Science
University of Missouri, Columbia, MO 65211 USA
xudong@missouri.edu

Weida Tong
Division of Bioinformatics and Biostatics,
National Center for Toxicological Research,
U.S. Food and Drug Administration (FDA),
3900 NCTR Road, Jefferson, Arkansas 72079 USA
Weida.Tong@FDA.hhs.gov

Hamid R. Arabnia
Department of Computer Science
University of Georgia, Athens, GA 30603 USA
hra@cs.uga.edu

Mary Qu Yang
Joint Bioinformatics Ph.D. Program
University of Arkansas Little Rock College of Engineering & Information Technology
and University of Arkansas for Medical Sciences
2801 South University Avenue, Little Rock, Arkansas 72204 USA
Mary.Yang@NIH.hhs.gov

I. INTRODUCTION

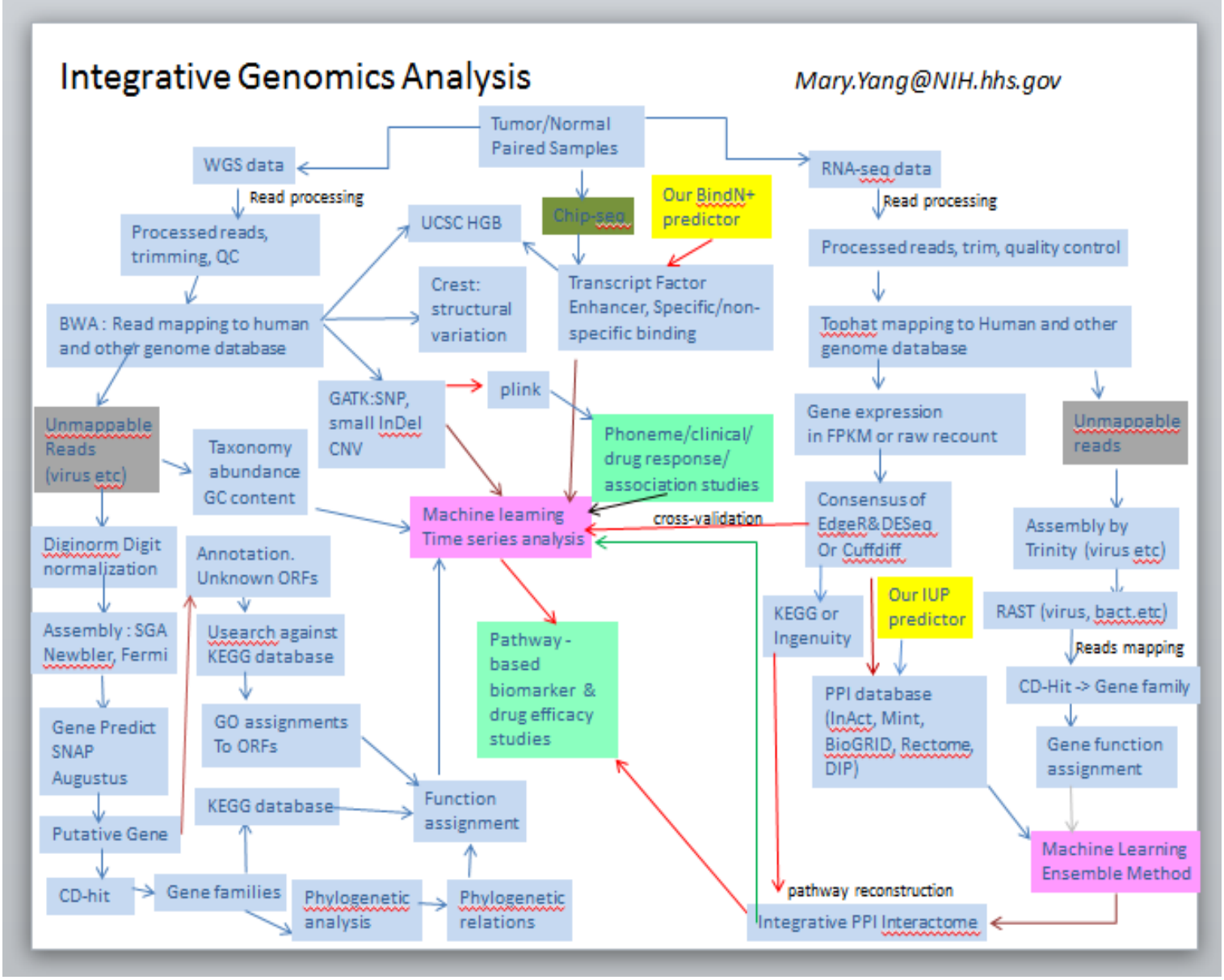
Synergistic integrating multiple genomic big-data at systems biology level can provide deeper insight on the molecular mechanisms relating to disease initiation and prognosis, and also guide many pathway-based biomarker identification and drug deliveries. NIH has initiated large number of genome-wide association study (GWAS), and whole genome/whole exome sequencing projects along with The Cancer Genome Atlas (TCGA) trying to identify the genomic mutations that are associated with cancer and other diseases. Those studies have produced rich data sets and have successfully identified large number of genomic mutations, including Single Nucleotide Polymorphisms (SNP), copy number variations (CNV) and structure variations (SV). However, most genomic mutations identified in those studies either have only a small disease risk effect, or are only present in a small fraction of the population in complex diseases such as cancer due the complexity and inter-patient heterogeneity. We leveraged the research by combining different genomic information including eQTL mapping, differential expression of genes and protein-protein interactions (PPI), to construct high-level gene networks for integrative genome-phoneme studies at a higher systems biology level.

We use Kidney Renal Clear Cell Carcinoma (KRIC) in this exemplary study. KIRC is the eighth most common cancer and is known to be the most lethal of all the genitourinary tumours with an estimate of ~65K new cases and ~13K deaths yearly. The cancer is a cortical tumour in kidney often characterized by malignant epithelial cells with clear cytoplasm and a compact-alveolar or acinar growth character scattered with intricate, arborizing vasculature. Knowing risk factors include use of drug NSAIDs (excluding aspirin) such as ibuprofen and naproxen; obesity; faulty genes; family history; and virus such as hepatitis C virus. This cancer has common presence of abnormalities on chromosome 3 and was suspected with translocations or somatic mutations in tumour suppressor genes on 3p. Genomic copy number alterations were suspected in chromosome arms 1p-, 3p-, 3q-, 4q-, 5p+, 6q-, 7p+, 8p-, 9p-, 9q-, 12p+, 13q-, 14q-, and 20q+ [1]. Regions frequently lost include 3p12-14, 3p21 and 3p25. KIRC may require loss of at least two regions in 3p and loss of 3p21 is obligatory. However if a tumour has only one deletion at 3p, either 3p14 or 3p25, it is designated as common type renal cell adenomas. This may indicate that KIRC has profound genetic alternations than Renal Cell Carcinoma (RCC). Although familial KIRC has been reported, KIRC is considered as sporadic tumours but also syndromic in patients with the von Hippel-Lindau (VHL) disease (germ line mutations in the VHL tumour suppressor gene assigned to 3p25) [2]. The tumour is reported to be resistant to radiation therapy and chemotherapy, although very few cases have been reported to response to immunotherapy. Targeted cancer therapies such as sunitinib, temsirolimus, bevacizumab, interferon-alpha, and sorafenib have slightly improved the outlook, although survival rate may not be improved. Immunotherapy including interferon and interleukin-2 only works on very few patients with limited efficacy. If the cancer can be detected in very early stages, it is potentially curable by surgical resection with adjuvant therapy. However, there is no curative treatment for metastatic KIRC. Therefore, a further investigation of the genomic alternations and underlying molecular mechanisms are essential for early diagnosis and treatment planning.

Successful identification of mutation-based biomarkers has also generated unwanted side effects of inhibitor treatment that often cause resistance to the drug. Our past studies on IUP (intrinsically unstructured protein) showed that due to differences in the post-translational circuitry such as the phosphorylation networks, where phosphorylation sites typically within IUP regions of motifs are dynamically un-conserved during evolution and cancerous cells often have aggregated mutations in the target kinase. Genetic interactions between kinases and substrates are unlikely conserved [13], amplification of MET in transactivation of EGFR and PI3K signalling can cause resistance toward EGFR inhibitors [4]. Therefore evolutionary divergence of phosphorylation and functional alternations in protein kinases are likely correlated. This may indicate that signalling networks have adapted to the relative low sequence specificity of tyrosine kinases by silencing the function. Furthermore, fewer sequence specificity found in tyrosine kinase domains has oncogenic propensity. Evolutionary conserved kinase substrate interactions in phosphoproteins are more likely mutated in cancer. Thus, while network connections between kinases and substrates are evolutionary dynamic, they are prone to be adapted to the constraint of a deterministic cellular context. Therefore rewiring of the signalling networks can be prone to cancer. When multiple diseases can target a conserved core network of kinases and substrates, conserved networks may affect one or more of the nodes. Crosstalk between EGFRvIII and other receptor tyrosine kinases can lead to resistance of EGFR inhibitor treatment in cancer and thus requires simultaneous use of several kinase inhibitors that result much less efficacy and more side-effects in general.

We consider that cancer is not only complex, in that many genetic variations contribute to malignant transformation, but also wildly heterogeneous, in that genetic mechanisms can vary significantly between patients. This gives great challenges on the efficacy of drug deliveries and treatment planning. New research advancement indicated that same type of cancer can have different subtypes with different genetic mechanisms and drug/treatment responses [3]. Therefore biomarkers derived from single genetic type such as SNPs, CNVs, DEG (differential expressions of genes), bidirectional promoters or protein structural changes / protein disorders usually are not well reproducible and often vary significantly with patience population. The identification of causal aberrant regulatory networks is crucial for early stage diagnosis of cancer but this task is confronted with lack of effective big-data methods to integrate different level of genomic data. We have developed methods to address the problem.

II. DATA AND METHOD

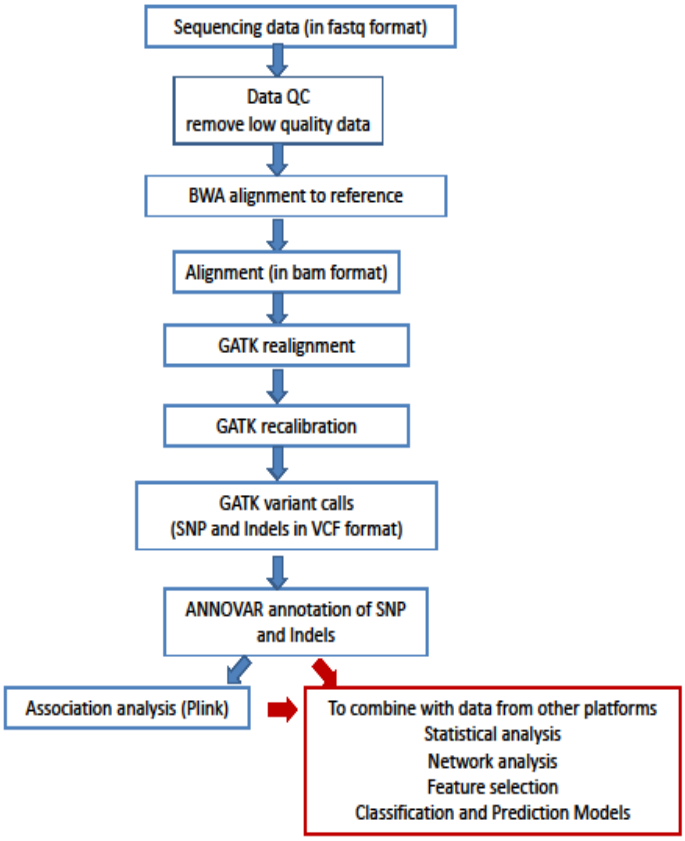


Because of the compensational effects in cells, loss of function in one part of signalling network can generate gain of function in another part of the network; thus genomic mutation can happen in a completely different node far from the DEG and therefore, mutation based biomarker identifications and drug design can have very limited efficacy. Given the situation, we must identify rewire signalling networks so that we can identify useful biomarkers and pinpointing the critical pathways at a higher systems biology level. To achieve this goal, we need to integrate multiple genomic data and perform further integrative investigations by developing new methods to identify causal genes and networks. We have designed our overall approach and data integration flowchart as shown in the above figure.

In the paper, we use Kidney Renal Clear Cell Carcinoma (KIRC) data to present a systems biology approach from the analysis of DEG. The TCGA contains comprehensive next generation sequencing data of KIRC. The data were downloaded from TCGA Data Portal Bulk Download (https://tcga-data.nci.nih.gov/tcga/). There were over 500 downloadable patient samples. RNA-seq Version 2 data was originated from the University of North Carolina GCC using Illumina Hi-seq 2000 platform. To test our methods in the first step, we use the data of 64 normal and 64 tumour paired tissues and 469 cancer tissues of KIRC patients and 4 normal kidney tissues as summarized below:

- 68 normal
- 68 tumor matched
- 469 cancer tissues from KIRC patients
- 4 Normal kidney tissues

Each sample contains over 20K genes from whole genome sequences. Because the dataset is very large, we wrote scripts using high performance computing techniques to process the big data as shown below:



The identification of driving genomic mutations is essential for understanding mechanisms of complex disease initialization and progression. Genome Wide Association Studies (GWAS) have been demonstrated capable of discovering genetic loci that contribute to disease risk and progress[15]. However due to linkage disequilibrium, individual genomic loci may contain up to hundreds of genes [16, 17], and genetic loci associated with diseases are often absent of transcription factors and not enriched in any biological functional categories [18, 19]. Many studies have found that most SNPs either have only a small effect on disease risk [20-22], or are only presented in a small fraction of the population [23, 24]. Consequently, clinical effective biomarkers and drug targets are not directly apparent from GWAS data [6, 7, 25]. Efforts that integrate multi-layer data to uncover the disease mechanisms and pathways have been made [26-29] but often yielded high rate of false positives. These efforts focused on the DEG, but DEG has shown to have very low reproducible rate across distinct patient cohorts [30-32]. Additionally, regulatory pathways connecting causal genes and differentially expressed genes were created based on expression correlation between genes, whereas such correlation may be lost in cancer progression [33, 34]. To overcome these problems, we have developed integrative approaches to identify disease causal genes and affected pathways.

For differential gene expression analysis from the TCGA RNA-seq data, we use EdgeR [6] and DESeq [7] in commercial free R biconductor package. We performed row count normalization and tag wise dispersion estimation. The genes with significant different level of expression and Log (fold change) > 5 were selected as differentially expressed gene set between KIRC tumour and normal tissues.
The high-throughput sequencing of RNA has many advantages over traditional solid-phase chip systems that include:

- not rely on prior knowledge of gene annotation
- greater dynamic range to detect gene expression
- capability to detect novel transcripts
- capability to detect splice variants
- detection of alternative transcription start and stop exons
- identification of 5' UTR and 3' UTR
- detection of fusion transcripts
- RNA-editing

Our ultimate experimental goals include detecting:

- differential gene expression
- rare transcript expression
- transcriptome assembly
- non-coding RNAs

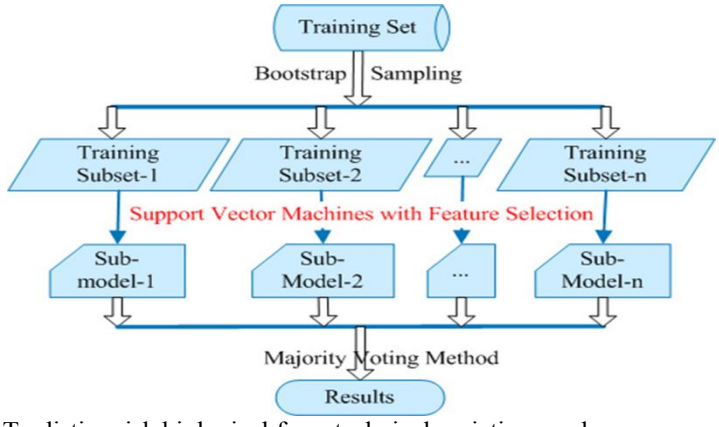
Discovering driving mutations requires identifying differential gene networks. Genomic mutations including SNPs, CNVs and SVs in disease development can be naturally perturbed, and the gene expression data can provide clues to infer the molecular interactions. However, traditional eQTL which tests single mutation and trait at a time suffers the pitfall of multiple testing, and does not take phenotype information into account. We use eQTL mapping to identify regulatory regions or genes harbored the genomic mutations, and then the information from eQTL are used to infer the network in accordance with phenotypic information. In this way, genomic variants were examined from sub-networks that are associated with particular phenotypes by combining expression and protein-protein interaction data. We therefore use eQTL mapping in combination with the protein interaction networks. The approach guides our construction of subnetworks from differentially expressed genes. Each subnetworks are used in our construction of the Variant of Self-Organizing Global Ranking Feature Maps [25] we developed before to further identify the overall links to phenotypic information. The phenotypes of tissue samples are permuted for statistical tests to assign significance. Regulations of differentially expressed genes and interconnectivities in the network maps are under investigation utilizing GO, KEEG and PPI annotations that lead to the identification of hallmarks of underlying mechanisms of cancer.

In addition, we consider that incorporating lncRNA (long non-coding RNA) expression data with disease-disrupted networks can offer new insight into the regulatory structure of the disease-associated networks. Although it is not yet known whether most lncRNA are functional or what their functions are, experiments have confirmed that some lncRNAs are functional and may function as regulators in a modular regulatory fashion. Recent studies have suggested that

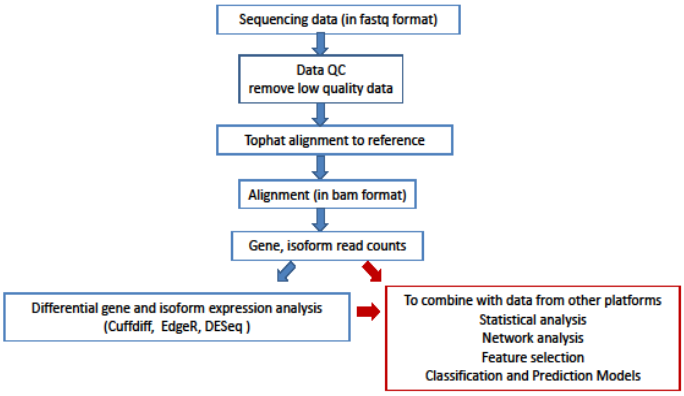
lncRNAs play important roles in oncogenesis and the roles of lncRNA in cancer have been identified that lncRNA can be the targets of somatic copy-number amplification. We consider that mutations on driver genes will likely change the expressions of a common set of diseased-associated lncRNAs, and we characterize the relationship between driver mutations and lncRNA expression, which may offer new insight into the regulatory structure of the disease-associated modules and the roles of bidirectional promoters in cancer.

We consider cancer as a highly heterogeneous disease; however, convergence to the same phenotypes suggests the existence of common molecular mechanisms in cancer development. We therefore
integrate transcriptome and protein interactome data and derive differentially expressed genes from expression profiles as well. The reproducibility of distinct types of molecular signature can be compared across different patient cohorts if available, where reproducibility is measured by the degree of overlap of molecular signatures across all independent patient groups. The results are compared with known disease genes in the Catalogue of Somatic Mutations in Cancer (COSMIC) and the Online Mendelian Inheritance in Man (OMIM) databases. Our work provides a systematic and comprehensive assessment of the reproducibility of biomarkers. In addition, molecular signatures that are recurrent over all patient cohorts is further studied using our previously generated network mapping to reveal their causative mutations and relevant pathways. By studying recurrent biomarkers, we can reveal causative genes and pathways for therapeutic targets that could potentially be used to inhibit or activate pathways related to the recurrent biomarkers.

This paper presents the identification of differentially expressed genes between the normal and tumour samples. We designed consensus Support Vector Machines (SVM) [5] with feature selection and bootstrap algorithms to classify tumour samples as shown



To distinguish biological from technical variations and identification of differential gene and transcript expression, we use pairwise and multi-factor experiment designs that include the over-dispersion model MDS (multidimensional scaling) plot to inspect the data. We use MDS to visualize the level of similarity which refers to a set of related ordination techniques used in information visualization, and display the information contained in a distance matrix. Hierarchical clustering algorithms were further performed and used to visualize the data. We group the differentially expressed genes between tumor and normal samples into four catalogues: overly expressed genes, under expressed genes, weakly over expressed genes, and weakly under expressed genes. Gene Ontology (GO) analyses were performed. Differentially expressed genes were compared based on GO annotations that include the biological process, molecular function, and cellular component of gene products and used as attributes of gene products for the construction of predication models. Therefore, our analysis of RNA-seq flowchart is illustrated below:



A fivefold cross-validation approach was used to evaluate the performance of our SVM classifiers. Positive and negative instances were distributed randomly into five folds. In each of the five iterative steps, four of the five folds were used to train a classifier, and then the classifier was evaluated using the holdout fold (test data). The predictions made for the test instances in all the five iterations are combined and used to compute the following performance measures:

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN}$$

$$\text{Sensitivity} = \frac{TP}{TP+FN}$$

$$\text{Specificity} = \frac{TN}{TN+FP}$$

$$\text{Strength} = \frac{\text{Sensitivity}+\text{Specificity}}{2}$$

where *TP* is the counts of true positives; *TN* is the true negatives; *FP* is the false positives; and *FN* is the false negatives. Because there are fewer paired normal/tumour samples than tumour samples alone, the dataset is imbalanced, both sensitivity and specificity are computed. The average of sensitivity and specificity, referred to as strength is evaluated for the performance. The strength is used as an accuracy indicator for the SVMs. We use Matthews Correlation Coefficient (MCC) as a measurement of the quality of binary classifications. It measures the correlation between predictions and the actual class labels. However, for imbalanced datasets, different trade-offs of sensitivity and specificity may give rise to different MCC values for a classifier. MCC is defined as:

$$\mathrm{MCC} = \frac{TP \times TN - FP \times FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}$$

The Receiver Operating Characteristic (ROC) curve is used for classifier evaluation and comparison. The ROC curve is drawn by plotting the true positive rate (*i.e.*, sensitivity) against the false positive rate, which equals to (1 – specificity). In this way, the ROC curve is generated by changing the output threshold of a classifier and plotting the true positive rate against false positive rate for each threshold value. The area under the ROC curve (AUC) can be used as a reliable measurement of classifier performance. Since the ROC plot is within a unit square, the maximum value of AUC is 1, which is achieved by a perfect classifier. Random classifiers have AUC values close to 0.5. To further improve the performance of week classifier, we developed an algorithm called Boosting with bagging [35] which emphasizes on weaker learner for each boosting run. If $e_t < ½$ on each boosting round t, then the training set error after T rounds of boosting, d_t, satisfies

$$\delta_T \le \prod_{t=1}^{T} 2\sqrt{\varepsilon_t(1-\varepsilon_t)}$$

Our analysis include unmatched samples of 469 KIRC and 4 normal tissues, we achieved heterogeneous clusters based on expression level of differentially expressed genes. Pathways of differentially expressed genes were investigated using KEGG (Kyoto Encyclopedia of Genes and Genomes) [8], which is a database resource for understanding high-level functions and utilities of the biological system. Significant pathways were identified using hypergeometric test. P-vales were ranked according to their significance. Significant pathways were further investigated by protein-protein interactions (PPI), obtained from four major databases: intact [9], Mint [10], BioGRID [11] and DIP [12]. We wrote a script to remove redundant PPIs, leaving about 120K unique human protein-protein interactions that were used in the subsequent analysis. Gene-wise normalization was performed from the expression profiles. We use the average Pearson Correlation Coefficient (PCC) to quantify the modularity differences between two phenotypes as follows:

$$\nabla_{r_{H,I}} = \frac{\sum_j (I_{j,N}-\bar{I}_N)(H_{j,N}-\bar{H}_L)}{(n_N-1)S_{I_N}S_{H_N}} - \frac{\sum_j (I_{j,T}-\bar{I}_T)(H_{j,T}-\bar{H}_T)}{(n_T-1)S_{I_T}S_{H_T}}$$

$$<\mathrm{PCC}> = \frac{1}{m-1}\sum_{i=1}^{m} |\nabla_{r_{H,I_i}}|$$

III. Results and Discussions

We used the SVMs and randomly select about 4/5 tissue samples for training and 1/5 for testing. We designed 5 consensus network training machines and repeat the random sampling 50 times. The performance is reported below:

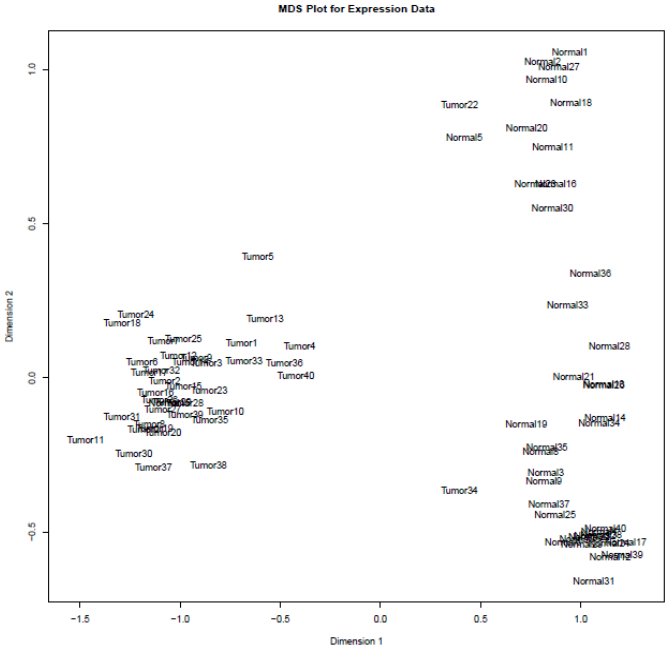
68 paired +469 KIRC+4 Norm	SVM Sensitivity	SVM Specificity	Area ROC
Mean(expectation)	96.5%	97%	0.987
Standard deviation	0.036	0.036	0.015

Gene expressions from over 20K genes were investigated. Levels of expression of all genes in all samples were sorted according to logFC, logCPM, P-value, and false discovery rate (FDR). Examples of significantly expressed differently of genes from tagwise dispersion are shown below:

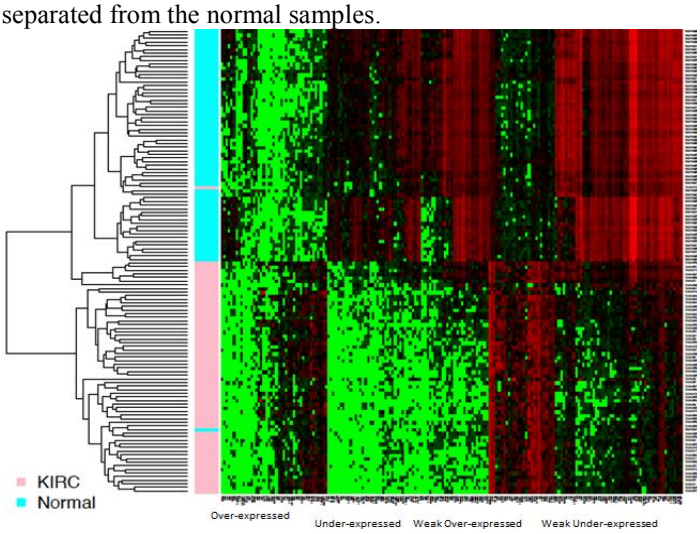
Gene	logFC	LogCPM	Pvalue	FDR
TRPV6	6.76	2.94	2.56E-73	4.21E-69
VEGFA	-3.57	10.02	1.19E-72	9.81E-69
PRR15	4.96	3.8	7.58E-70	4.16E-66
LOC284578	7.31	5.17	5.73E-66	2.36E-62
MFSD4	5.66	8.01	1.07E-61	3.54E-58
ACPP	5.88	4.71	1.13E-60	3.09E-57
NOL3	-3.39	5.97	8.25E-59	1.94E-55
MECOM	2.88	7.11	8.28E-58	1.71E-54
GATA3	3.91	5.36	3.39E-57	6.20E-54
USP44	4.02	0.66	5.30E-56	8.73E-53
NHLRC4	3.6	2.84	3.65E-55	5.48E-52
TYRP1	7.95	3.29	4.24E-55	5.83E-52
DDB2	-2.03	4.85	9.93E-55	1.26E-51
TMEM91	-4	5.36	1.17E-54	1.38E-51
SLC15A4	-1.64	5.92	1.64E-54	1.81E-51
CADPS2	1.75	6.24	2.32E-54	2.39E-51
EGLN3	-4.08	8.4	4.90E-54	4.75E-51
KRBA1	-2.85	5.26	6.03E-54	5.52E-51
C20orf46	-3.55	1.68	7.84E-53	6.8E-50
C12orf34	3.23	3.5	3.56E-51	2.9E-48

The tagwise dispersion showed less differentially expressed genes (DEG) than Poisson dispersion but more DEG than normal dispersion.

Below is the MDS (multidimensional scaling) applied to visualize patterns of tumour and normal samples.



The heatmap of DEG is reported below. Both heatmap and MDS show that normal samples have distinct patterns from tumour samples. Gene expressions in tumours are well separated from the normal samples.



As showing the above figures, overall gene expressions in tumours are well separated from the normal samples.

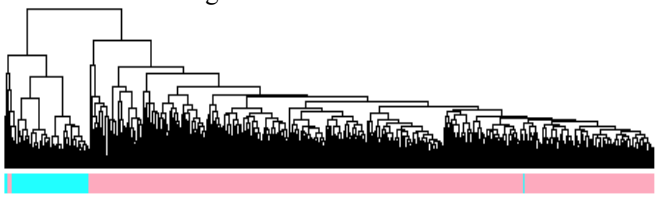
Consensus networking algorithms were used for edgeR and DESeq results. Differentially expressed genes are divided into up-regulators and down-regulators.

The gene ontology (GO) analyses were performed. Differentially expressed genes were grouped into 4 categories. Each category showed distinct pattern of different pathways as reported below:

Gene Group	GO Term (P < 0.01, Fisher's Exact Test with Benjamini multiple test correction)
over-expressed	Defense response Response to environmental stimulus
under-expressed	organismal physiological process system development organ development
weak_over-expressed	cell-cell signalling lipid metabolism signal transmission across a synapse organismal physiological process
weak_under-expressed	excretion secretion transmembrane transporter activity

The results resonate some of current research findings. Some top up-regulated genes are known to have tumour relevancy. For example TNFAIP6 is a tumour necrosis factor which may mean that the tumours in the patients have developed pretty badly that may significantly affect their kidney function. Another example SLC6A3 is known in lung and breast cancer but has not been reported in kidney cancer. However many of known oncogenes are not differentially expressed. Those down and up regulated genes warrant an investigation at a systems biology level. From our comprehensive examinations, we can conclude that biomarkers derived from differentially expressed genes alone may not be well reproducible for out-of-sample data and can vary significantly among patient population. We must further investigate the gene interactions and networks.

We identified 185 significantly differentially expressed genes. Using the expression of these genes, we found 128 (64 normal and 64 KIRC paired) tissues were well clustered into two groups. One normal and one tumour tissue were mis-clustered, and both tissues are from the same patients. This suggests that they are likely to be mislabelled or outliers. We may contact TCGA for further clarification of the outliners. Furthermore, we included our analysis to unmatched tissues by adding 469 KIRC and 4 normal tissues. We achieved two heterogeneous clusters based on expression level of differentially expressed genes; only 4 out all tissues were mis-clustered as showing below:



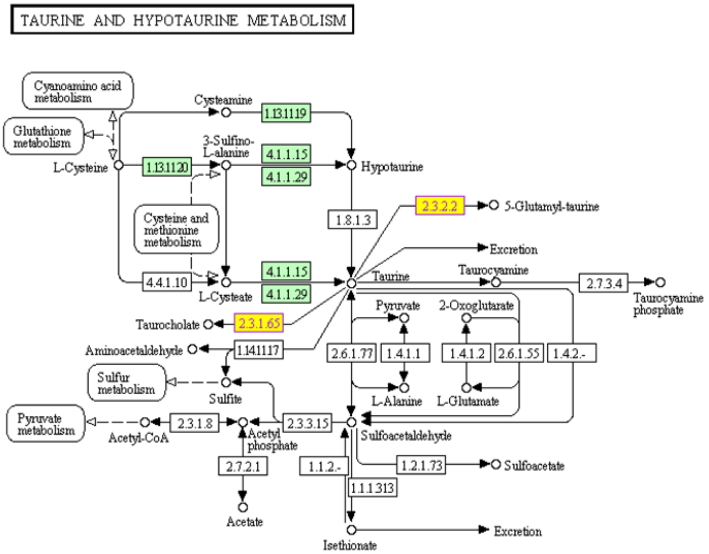
The hierarch clusters of 469 KIRC and 68 normal tissues. Cyan represents normal tissues and pink represents kirc tissues.

Differentially expressed genes and KEGG pathways were investigated. Hypergeometric test was performed. Summary of examples of significant pathways from differentially expressed genes is shown below:

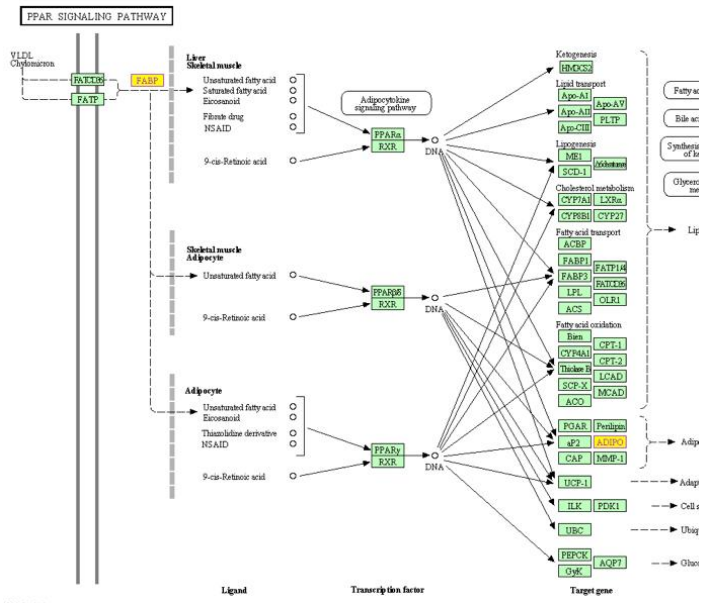
Differential Genes and KEGG Pathways

P-value (hypergeometric test)	Pathway
0.002	Taurine and hypotaurine metabolism
0.016	Neuroactiveligand-receptor interaction
0.025	Glycosaminoglycan biosynthesis - heparan sulfate
0.033	PPAR signaling pathway
0.0346	Hepatitis C
0.039	Gastric acid secretion

Those pathways related to DEG can lead clues to identify causal genes and networks. For examples, the Taurine and hypotaurine metabolism is shown below:



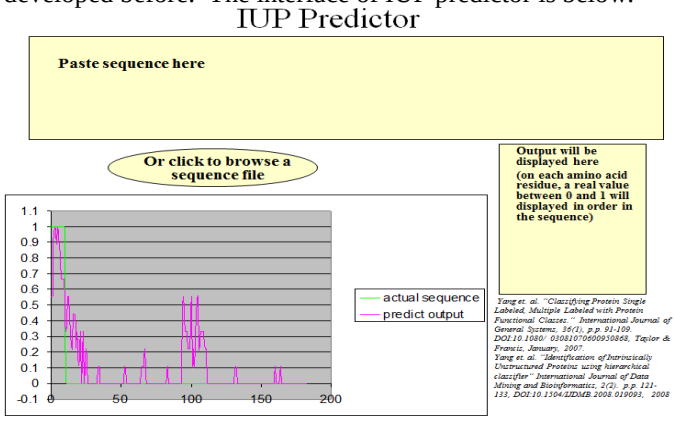
The PPAR signalling and pathway is shown below:



Based on protein interactions, we identified a number of pathways involved in cancer development. Examples of two networks containing significant differentially expressed genes are reported as:

- Molecular Transport, Heredirary Disorder, Metabolic Disease
- Network of Renal and Urological Disease

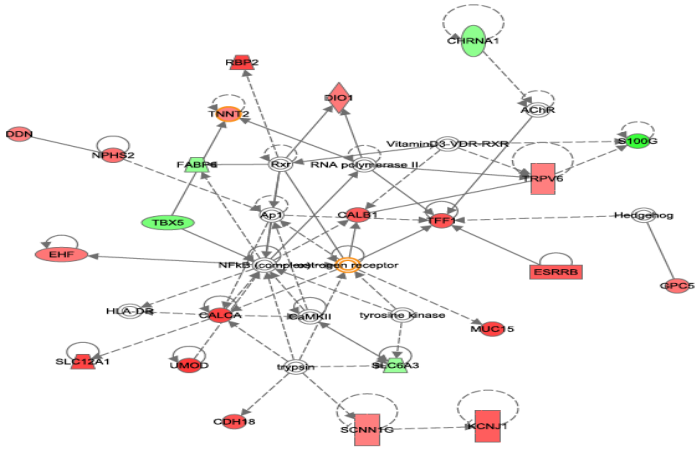
Proteins that bind a specific DNA or RNA sequence may also bind non-specific sequences. For example, for transcriptional regulators, a binding is often intrinsic to function; in other cases such as tRNAs, the binding is not productive. To identity the mechanism of malignant transformation, we need to identify all proteins that distinguish one DNA or RNA from another more accurately and need the information of IUP (intrinsically unstructured proteins)[13]. In order to know the driving causes of malignant transformation, we explored the IUP, and Protein-DNA/RNA interactions and their roles in DEG. We used our IUP and BindN+[14] predictors we developed before. The interface of IUP predictor is below:



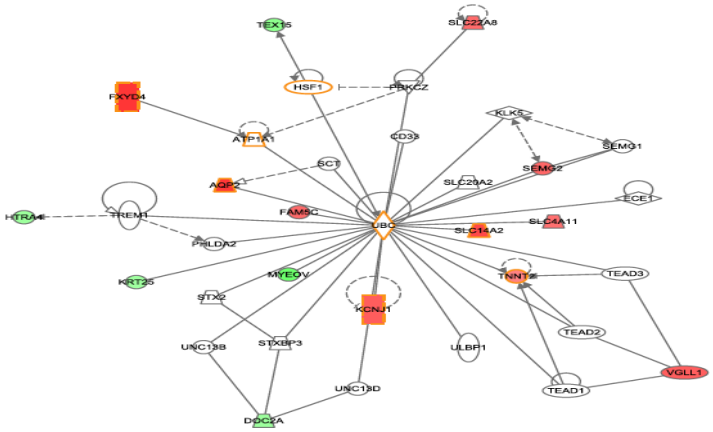
Results showed that we need to identify protein interaction networks that can provide a method to detect novel disease causative genes and pathways. They can provide guidance for personalized drug design and treatment planning. They can offer new frameworks for downstream functional genomics and systems biology analysis for large-scale integrative analysis of next generation sequence data and other -omics data that include:

- reveal novel biomarkers and pathways in disease;
- identify master regulator or driver genes/mutations attributed to pathology and progression of disease;
- Design new drugs with improved efficacy building intelligent medical systems for early diagnosis, and treatment panning

Therefore protein interactions were also investigated. We assembled interacting network databases using protein-protein interaction, protein-DNA interaction, and protein phosphorylation between protein and kinase. Using our pathway analysis, we found a number of networks that contain significant numbers of differentially expressed genes. Over 90% of subnetworks were significantly enriched in at least one biological process term. The network approach enhances our ability to discover clusters of genes that function in pathways affected by disease and drug response. For example: Network of Molecular Transport, Heredirary Disorder, Metabolic Disease is shown below:



Another pathway of renal and urological disease is shown below:



The pipelines we have developed support major categories of next generation sequencing data process and analysis. A number of driver genes and causal pathways were identified. Results showed that integrating the information of the underlying network structure as a function of the interactions of individual molecular components in a living organism can lead to a higher systems biology level view of the cellular behaviors such as malignant transformation and the research

initiative is part of our development of integrative genomic big-data analysis. Since systems biology approaches can reveal a number of mechanisms from gene regulatory networks, and protein-DNA/RNA interactions, to the ultimate cellular behaviors such as malignant formulation, our approaches include the development of novel high throughput, sensitive, reliable and integrative analytical methods for the characterization of DEG, IUP, Protein-Protein/DNA interactions and their products, thus lead to the identification of pathway based molecular biomarkers and their potentials applications for the early diagnosis of cancer and better design of resistance-free drug deliveries.

References

1. L Moore, E Jaeger, M Nickerson, P Brennan, S Vries, R Roy, J Toro, H Li, S Karami, P Lenz, D Zaridze et. al. "*Genomic copy number alterations in clear cell renal carcinoma: associations with case characteristics & mechanisms of VHL gene inactivation"* Oncogenesis, 1e14; doi:10.1038/oncsis.2012.14, 2012.
2. E Maher, H Neumann, S Richard *"von Hippel–Lindau disease: A clinical and scientific review"* European Journal of Human Genetics 19, 617–623, 2011
3. N Navin, J Kendall et al. *Tumour evolution inferred by single-cell sequencing.* Nature; 472:90-4. 2011
4. C Chong, P Jänne "*The quest to overcome resistance to EGFR-targeted therapies in cancer"* Nature Medicine, Nov 7, 2013
5. C Cortes, V Vapnik, *"Support-vector networks"* Mach. Learning 20(3): doi:10.1007/BF00994018. 1995
6. M Robinson, D McCarthy, G Smyth "edgeR: a Bioconductor package for differential expression analysis of digital gene expression data." Bioinformatics, 26, pp. 1, 2010
7. A Anders, W Huber "Differential expression analysis for sequence count data." Genome Biology, 11, pp. R106. http://dx.doi.org/10.1186/gb-201, 2010
8. M Kaneshisa, S Goto "KEGG (Kyoto Encyclopedia of Genes and Genomes" Nucleic Acids Research, 28(1), pp 27-30 2000
9. S Orchad et al "The MIntAct project--IntAct as a common curation platform for 11 molecular interaction databases." Nucl. Acids Res. doi: 10.1093/nar/gkt1115 PMID: 24234451, 2013
10. L Licata1, L Briganti, D Peluso, L Perfetto, M Iannuccelli, E Galeota, F Sacco, A Palma, A Nardozza, E Santonico et. al. "MINT, the molecular interaction database: 2012 update" Nuc. Acids Res. doi: 10.1093/nar/gkr930 November 16, 2011
11. C Stark, B Breitkreutz , T Reguly, L Boucher, A Breitkreutz, M Tyers M "BioGRID: A General Repository for Interaction Datasets". Nucleic Acids Research, 34 (database issue): D535–D539. doi:10.1093/nar/gkj109. PMC 1347471, 2006
12. I. Xenarios, D Rice, L Salwinski, M Baron, E Marcotte, D Eisenberg "DIP: the database of interacting proteins." Nucleic acids research, 28(1), pp. 289-291, doi:10.1093/nar/28.1.289, Jan 1, 2000
13. C. Oldfield, AK Dunker "Intrinsically Disordered Proteins and Intrinsically Disordered Protein Regions." Rev Biochem. March 2014.
14. L Wang, C Huang, M Yang, J. Yang. "BindN+ for accurate prediction of DNA and RNA-binding residues from protein sequence features." *BMC Systems Biology*.; doi: 10.1186/1752-0509-4-S1-S3. PubMID: 20522253, May 28, 2010
15. D Altshuler, M Daly, E Lander, *Genetic mapping in human disease.* Science, 2008. **322**(5903): p. 881-8.
16. H Brunner, M van Driel, *From syndrome families to functional genomics.* Nat Rev Genet, **5**(7) p545-51. 2004
17. D Botstein, N Risch, *Discovering genotypes underlying human phenotypes: past successes for mendelian disease, future approaches for complex disease.* Nat Genet, 2003. 33 Suppl: p. 228-37.
18. E Schadt, *Mol networks as sensors & drivers of common human diseases.* Nature, 2009. 461(7261): p. 218-23.
19. G Yvert et al., *Trans-acting regulatory variation in Saccharomyces cerevisiae and the role of transcription factors.* Nat Genet, 2003. **35**(1): p. 57-64.
20. P Kraft, D Hunter, *Genetic risk prediction--are we there yet?* N Engl J Med, 2009. **360**(17): p1701-3.
21. D Goldstein, *Common genetic variation and human traits.* N Engl J Med, 2009. **360**(17): p. 1696-8.
22. K Christensen, J Murray, *What genome-wide association studies can do for medicine.* N Engl J Med, 2007. **356**(11): p. 1094-7.
23. T Manolio, et al., *Finding the missing heritability of complex diseases.* Nature, 2009. **461**(7265): p. 747-53.
24. E Eichler, et al., *Missing heritability and strategies for finding the underlying causes of complex disease.* Nat Rev Genet, 2010. **11**(6): p. 446-50.
25. J Hardy, A Singleton, *Genomewide association studies & human disease.* N Eng J Med, 2009. **360**(17): p. 1759-68.
26. Z Tu et al., *An integrative approach for causal gene identification and gene regulatory pathway inference.* Bioinformatics, 2006. **22**(14): p. e489-96.
27. S Suthram et al., *eQED: an efficient method for interpreting eQTL associations using protein networks.* Mol Syst Biol, 2008. **4**: p. 162.
28. E Yeger-Lotem et al., *Bridging high-throughput genetic and transcriptional data reveals cellular responses to alpha-synuclein toxicity.* Nat Genet, 2009. **41**(3)p316-23.
29. 15. Kim, S Wuchty, T Przytycka, *Identifying causal genes and dysregulated pathways in complex diseases.* PLoS Comput Biol, 2011. **7**(3): p. e1001095.
30. W Symmans et al., *Breast cancer heterogeneity: evaluation of clonality in primary and metastatic lesions.* Hum Pathol, 1995. **26**(2): p. 210-6.
31. S Tomlins, et al., *Recurrent fusion of TMPRSS2 and ETS transcription factor genes in prostate cancer.* Science, 2005. **310**(5748): p. 644-8.
32. L Ein-Dor et al., *Outcome signature genes in breast cancer: is there a unique set?* Bioinformatics, 2005. **21**(2): p. 171-8.
33. I Taylor, et al., *Dynamic modularity in protein interaction networks predicts breast cancer outcome.* Nat Biotechnol, 2009. **27**(2): p. 199-204.
34. A Barabasi, Z Oltvai, *Network biology: understanding the cell's functional organization.* Nat Rev Genet, 2004. 5(2): p. 101-13.
35. M Yang, *Predicting protein structure and function using machine learning methods.* Ph.D. Dissertation, Purdue University, ISBN:0-542-59618-0, ACM Digtial Library.

Exploring and Analyzing Big Data Processing Technology Based on the EMU

Qiming Niu [12], Feng Liu [1], Jie Liu [3], and Chun Zhang [1]
[1] School of Computer Science, Beijing Jiaotong University, Beijing, China
[2] Computing center, Hebei University, Baoding, China
[3] Department of Computer Science, Western Oregon University, Monmouth, Oregon, USA

Abstract - *In the era of big data, the development of the railway industry faces new challenges, and brings new opportunities. Big Data processing technology is a powerful tool to deal with Electric Multiple Unit(EMU) big data analysis. Application of Big Data processing technology is a critical development stage of intelligent EMU. In the process of EMU's operation, repair and maintenance, it generate a large number of heterogeneous, multi-state data. Only the data are effectively dealt with before it can be more quickly to access and manipulate. Therefore, the method of processing data become more and more important. combining the current development situation of EMU, This paper explores and analyses Big Data processing technology based on EMU.*

Keywords:EMU, Big Data Integration, Big Data Transmission, Big Data Storage, Big Data Mining, Big Data Visualization.

1 Introduction

Electric Multiple Unit(EMU) consists of a number of electric traction motor train units and a corresponding number of trailer units. It is suitable for fast passenger transportation. At present, some countries have carried out research on intelligent EMU. An important prerequisite is the application of the big data for the smart EMU safety, reliable operation. The application includes real-time data acquisition, data integration, transmission, storage, big data rapid analysis, security, authentication, and visualization of intelligent EMU. With the intelligent EMU construction scale increasing, the amount of data generated in the intelligent EMU operation, repair and maintenance are growing exponentially, which attracts the attention of many researchers and stake holders. The processing of large data cannot do without the advanced in big data processing technology based on EMU. The following will deeply explore and analyze big data processing technology based on intelligent EMU from different perspectives.

2 The technology of processing multiple heterogeneous data

Intelligent EMU in the next generation require organic combination of different processes such as manufacture, assembly, operation, repair, maintenance, etc. The technology of processing multiple heterogeneous data can effectively carry on the comprehensive collection of information, and support the information transmission and processing effectively. It integrates the data flow, information flow, and business flow. Through large-scale integration of multi-source heterogeneous data, the technology provides analysis data for the intelligent EMU applications. For a large number of heterogeneous data, it is necessary through the establishment of a unified data model to achieve the purpose of data fusion, at the same time, it is also necessary to further improve these heterogeneous data storages, queries and analysis, to ensure timely access to these heterogeneous data. Normally, information systems of different types of different functions of EMU are developed on the actual demand of the corresponding departments, all these information systems work in all kinds of platform, also have a variety of data format. Therefore, it brings a series of problems, such as the scattered data, heterogeneous data, not sharing and not easily achieving the related data. Specifically, the intelligent EMU mainly involves the following systems: PDM (Product Data Management), CAD (Computer Aided Design), CAM (Computer Aided Manufacturing), ERP (Enterprise Resource Planning), MES (Manufacturing Execution System) [1], MRO (Maintenance, Repair, Overhaul Management System) [2], sales management information system, management information system, and maintenance management information system. They are independent of each other, it is difficult to realize data sharing. In this case, we should adopt the technology of processing multiple heterogeneous data, with the help of big data processing platforms, we can effectively integrate the different information system data, and make full use of them.

3 The technology of data transmission and storage

With the rapid development of intelligent EMU, both intelligent EMU operation data and on-line equipment state monitoring data can be recorded. This caused the intelligent EMU big data transmission and storage problem. On the one hand, huge amounts of data for intelligent EMU's impact on normal, high efficiency, stable operation. On the other hand, big data, also promote the healthy development of the intelligent EMU. In this situation, with the aid of data compression technology, it can reduce the amount of network data transmission to a certain extent, and significantly enhance the efficiency of data storage. However, data compression technology requires support of CPU resources. At the same time, Big Data based EMU can be used through a Distributed File System[3], usually with the help of the HDFS (Hadoop Distributed File System)[4] for storage.

4 The technology of data analysis

At present, EMU manufacturing enterprise information system are confined in the business process layer, providing a fixed reporting function. Both managers and decision makers can only obtain limited static business information, which is independent of each other, and lacks of association. It cannot satisfy the enterprise decision-making only through displaying the statistical information, because EMU manufacturing companies have a huge impact on the organizational structure and complex business systems, and complex social and international situation. Therefore, in the process of implementing strategy of big data, the EMU manufacturing enterprise is necessary to make full use of data mining technology to associate the isolated data, such as data, people, environment, policy, and business, in order to provide more comprehensive analysis, to provide more accurate prediction, and to provide more practical significance decision making.

Firstly, EMU production. Using large amounts data from each EMU information system, combining with geographic information data and other external environment data, considering different regional level of productivity, energy distribution, terrain advantage, the feature of climate, we may use data mining technology to get decision-making information of the EMU planning construction, in order to realize the optimal allocation of resources, avoid the EMU marketing issues. In this way, it can effectively help manufacturing enterprises to optimize asset and capital expenditures.

Secondly, EMU Marketing. We analyze various types of data, including contract management data, account management data, inspection and maintenance data, business data, market management data, customer relationship management data, and other marketing data, combining with various types of data related to the national policy, the economic development, the natural environment, and we get transportation law and transport behavior in different regions and different industries. There are a lot of marketing analysis applications, such as customer segmentation, anomaly detection, customer credit rating forecasting, demand forecasting and so on.

Thirdly, EMU Security Operations. We use distributed processing and data mining technology, analyse real-time large amounts of data from train monitoring systems, sensor networks and surveillance cameras. In this way we can improve the level of security detection EMU, timely detect EMU failure and give an effective solution.

Finally, EMU Equipment Maintenance. In the equipment maintenance system, we use the equipment repair records, including repair time, repair costs, repair personnel number, and associated equipment information, analyze and achieve positioning fault in time, assess potential risks. We can predict which device needed maintenance in a specific time, and gives a corresponding detailed maintenance scheme using the maintenance model.

In brief, to make good use of each business system contained in the production data, sales data, operation data and repair data, maintenance data, guided by the business objectives, we use data mining algorithm to analyze the data deeply, find the value behind big data, finally get reasonable solution effectively, in order to promote the development and progress of enterprise and customer.

5 The technology of data visualization

This is a particularly challenging task that we present the results of the analysis to users at all levels through an intuitive and easy to understand way in a limited time. Visualization method is an effective method for large-scale data analysis [5]. In practice, this method also has obtained the good effect. Big Data from all kinds of application system mainly includes high precision data, high resolution data, time-varying data and multivariate data, etc. A typical data set can reach the level of magnitude of TB. How quickly deal with large and complex data, is becoming increasingly important. It is also a very important technical difficulty in the smart application of EMU. Some big data analysis system [6] [7] can use a variety of complex algorithms to plot data into a high precision, high resolution images, at the same time, provide interactive operation, which can change the data and algorithm parameters in real time. Thus we can observe data in a timely and effective manner, and can analyze the data qualitatively and quantitatively.

6 Conclusion

In summary, this paper has carried on the exploration and analysis of big data processing technology based on Intelligent EMU. At present, the research on the processing technology of big data based on Intelligent EMU has made some progress, however, we are facing a lot of challenges in real-time analysis, data consistency, security, privacy, ect.

We require more experts and scholars to further explore data processing technology based on the EMU.

7 Acknowledgements

This work is supported by The National High Technology Research and Development Program of China (Grant No.2012AA040912). The authors would like to thank reviewers for their review and helpful comments on a draft of this paper.

8 References

[1] http://en.wikipedia.org/wiki/Big_data
[2] http://mro.thss.tsinghua.edu.cn
[3] http://en.wikipedia.org/wiki/Distributed_file_system
[4] http://hadoop.apache.org/docs/r2.4.0/hadoop-project-dist/hadoop-hdfs/Federation.html
[5] http://datavlab.org
[6] http://www.vertica.com/resources/white-papers
[7] http://www.gopivotal.com/big-data/pivotal-big-data-suite

Big Data architecture for large-scale scientific computing

Benoit Lange
Project OPALE
INRIA Grenoble
38334 Sant-Ismier, France
benoit.lange@inria.fr

Toan Nguyen
Project OPALE
INRIA Grenoble
38334 Sant-Ismier, France
toan.nguyen@inria.fr

Abstract—**ABDA'14: POSITION PAPER.**

Today, the scientific community uses massively simulations to test their theories and to understand physical phenomena. Simulation is however limited by two important factors: the number of elements used and the number of time-steps which are computed and stored. Both limits are constrained by hardware capabilities (computation nodes and/or storage).

From this observation arises the VELaSSCo project[1]. The goal is to design, implement and deploy a platform to store data for DEM (Discrete Element Method) and FEM (Finite Element Method) simulations. These simulations can produce huge amounts of data regarding to the number of elements (particles in DEM) which are computed, and also regarding to the number of time-steps processed. The VELaSSCo platform solves this problem by providing a framework fulfilling the application needs and running on any available hardware.

This platform is composed of different software modules: a Hadoop distribution and some specific plug-ins. The plug-ins which are designed deal with the data produced by the simulations. The output of the platform is designed to fit with requirements of available visualization software.

Keywords*—*Big Data architecture, Scientific simulation, VELaSSCo, Hadoop

I. Introduction

The data production rate has followed a path similar to computation hardware (based on Moores law). The amount of information has an exponential growth while hardware storage capabilities does not follow a similar path. Moreover, the data produced has also an impact on which architecture is needed. This amount of data is extracted by several sources: sensors, simulations, users, etc. For example, the LSST produces 30 terabytes of astrophysics data every night [1]. Simulations can also create large amount of data as in [2], where the authors present a parallel implementation of the Denhen algorithm [3], an astrophysical N-body simulator. This implementation produces 500 Megabytes of data in 1.19 seconds (for a plummer distribution with 10 M particles, and only one time-step). HPC facilities, which are used by scientists to perform simulations, are not currently designed to store such important amounts of data: these systems are only suitable to provide efficient computation capabilities.

[1]http://www.velassco.eu

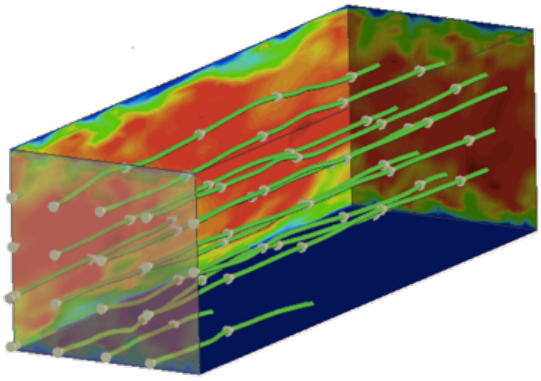


Fig. 1. Visualization of FEM simulation (Air flow), produced by GID (CIMNE).

This paper presents the VELaSSCo project: it provides a BigData architecture to store the data produced by various simulation engines. This data must be visualized by specific tools. For this purpose, two visualization software are targeted: GID[2] from CIMNE[3] and I-FX[4] from Fraunhofer IGD[5] .

The project is also focused on specific data produced by two different simulation engines: FEM and DEM data. An example of visualization of FEM simulation data is presented in Figure 1. This simulation deals with the decomposition of space using a mesh structure, and it is used to understand the dynamic of specific objects. For the DEM, a particle example is presented in Figure 2 (The figure has be produced by the University of Edinburg[6]). Both of these solutions produce important amounts of information: for 10 millions particles and 1 billion of time-steps, DEM uses 1 Petabytes of data, or 1 billion elements with 25000 time-steps, whereas FEM produces 50 TB of data. Currently, all the data produced by these simulations data are simply not stored, and several time-steps are deleted from storage device.

This paper focuses on the Big Data architecture designed for the VELaSSCo project. The platform is designed to be scalable regarding to which IT capabilities are available (HPC,

[2]http://www.gidhome.com
[3]http://www.cimne.com
[4]http://www.i-fx.net
[5]https://www.igd.fraunhofer.de
[6]http://www.ed.ac.uk

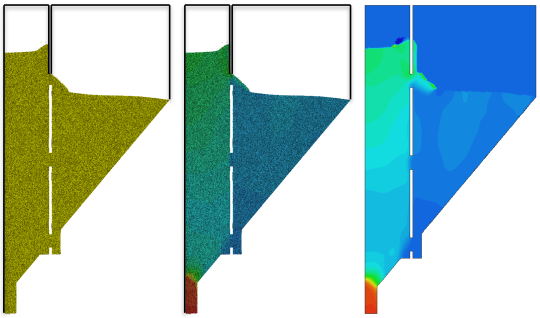


Fig. 2. Visualization of DEM data (UEDIN), silo discharge.

Clouds, etc.). It also interfaces visualization tools. It must also interface some commercial tools specifically designed to deal with engineering data.

Section II is an overview of related work on Big Data. Section III addresses the architecture of the VELaSSCo platform. Section IV is a conclusion.

II. Related Work

This section is mainly focused on Big Data for engineering applications. Problematics linked with this field are not widely developed in the literature. Most of Big Data related problems concentrate on Web crawling and analytics. Further, simple visualization queries for engineering simulations are similar to Web crawling. Therefore, we assume that using solutions provided for Web search can enhance engineering applications and visualization queries.

MapReduce computation has been massively studied and developed recently. Traditional Big Data approaches are mainly based on MapReduce computations to extract information. Strict implementations have been proposed for this computation model. But evolutions have also been presented and follow two different paths: Hadoop compliant and none Hadoop compliant software. Hadoop[7] is an open-source project which implements all the needs with respect to distribute processing systems for large-scale data. This project was mainly inspired by Google papers [4] and [5].

At the same time, non-Hadoop compliant solutions have been developed, which have been designed by database providers, e.g., to propose a BigData platform based on existing products. In other cases, these solutions are developed to deal with other requirements than Hadoop does. An example is to store big data on HPC facilities without dedicate storage, and run MapReduce jobs on the HPC nodes. These strategies have been designed to provide solutions for running big data applications on traditional data-centers.

Also, the MapReduce programming model has been ported to HPC facilities, while Hadoop is mainly developed to run on a dedicated storage nodes. In [6], [7], authors present two implementations of MapReduce dedicated to HPC facilities. Their strategies allow to apply MapReduce jobs on POSIX compliant file systems, and an abstract layer is not necessary (like HDFS). A deeper study of these methods is not possible, because the code source is not directly available and the extensibility of these solutions is not discussed.

Global frameworks like Hadoop have also been proposed by the scientific community. One of them is Dryad, [8]. This solution is designed to extend the standard MapReduce model by adding intermediate layers between the Map phase and the Reduce phase. Now, this implementation has been ported to the Hadoop ecosystem, and Dryad is a full extension of Hadoop using YARN. This software is available on the GitHub repository at Microsoft[8].

Regarding our needs, our interest is focused on the Hadoop ecosystem and more precisely on two extensions. The first one shows the usage of Hadoop over HPC, and the second one deals with an extension of the Hadoop storage with an existing database system.

The paper [9] presents how Hadoop is used over a traditional HPC system. This solution is decomposed as follows: Hadoop services are started, then the necessary files are transferred to HDFS, then the computation is run. After the computation, Hadoop services are stopped and the HDFS partition is destroyed. This solution highlights some bottlenecks: data transfers between HDFS and the HPC file system. Due to the HPC structure, authors do not use local storage of the HPC: indeed this storage can only be used as a temporary repository.

The second paper [10] presents a Hadoop extension which uses a RDBMS (Relational Data Base Management System) to store the data. This storage system is used instead of the traditional HDFS solution. The goal is to improve the query speed over Hadoop using the SQL engine of the RDBMS. Their example stores data into a Postgresql database, but any database system can be used instead.

III. Architecture

This section presents the architecture used for the VE LaSSCo platform. It is designed to fit with specific requirements of engineering data simulations. These requirements are:

- the platform has to be compatible with various computing infrastructure: HPC, clouds, grids, etc.,
- the data produced by the simulation engines can be computed by several nodes,
- the visualization queries can be simple or complex,
- the visualization queries will be performed in batch or in real-time,
- the architecture has to be extensible, scalable and supported by a large community of users.

For the first requirement, we are currently extending the solution presented in [9]. This tool provides a solution to deploy a Hadoop ecosystem on any kind of computation infrastructure, and moreover it reduces the bottleneck due to numerous file transfers between the virtual file system (FS) and the HPC FS. Regarding to our partners requirements, it is necessary to provide a solution which can be parameterized to deal with the specifics of their computing facilities. This solution is designed to be suitable for three different cases:

[7] http://hadoop.apache.org

[8] https://github.com/MicrosoftResearchSVC/Dryad

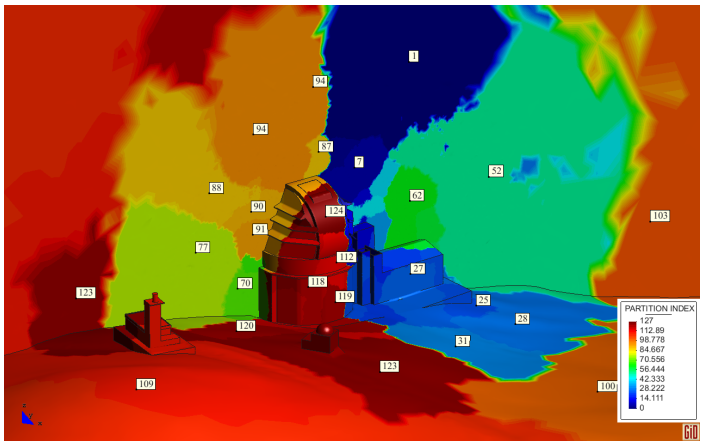


Fig. 3. Different partition of space for a FEM simulation (provided by CIMNE).

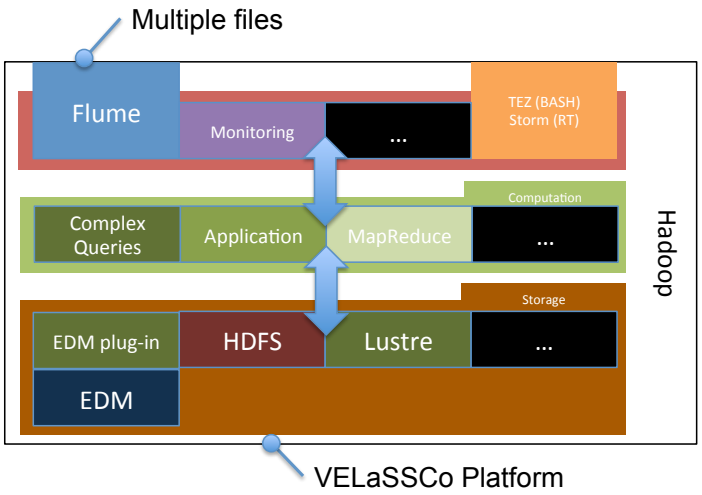


Fig. 4. Expected architecture of the VELaSSCo platform.

1) HPC and dedicated storage nodes,
2) HPC nodes with dedicated local storage,
3) HPC nodes with an existing distributed storage system.

For the first point, a HPC infrastructure coexists with storage facilities. This solution is quite new for users from data simulation. It implies to have two data-centers topologies with dedicated nodes for both sides (HPC and Hadoop). To avoid deployment of such an architecture, it is possible to extend an existing datacenter using external providers like Amazon (with EC2[9] and S3[10]).

The second approach uses dedicated storage for storing data. All the nodes in the HPC have a specific local storage dedicated to Big Data. A local hard disk is already used to store local data during computations. For these HPC facilities, it is possible to add a specific storage. In this architecture, we dedicate this local storage to all necessary information concerning the BigData architecture. This solution can only be implemented on private computing facilities, with possible hardware modifications.

The last solution uses the distributed FS of the actual HPC to store the data. This approach is the most suitable solution for public computing facilities without extensibility for users. With this approach the data transfers are an important bottleneck.

To fit with all these cases, we extend the myHadoop implementation [9], by providing all the necessary modules to deploy a VELaSSCo platform on all kind of computing facilities. This tool also provides the necessary interfaces to deal with visualization queries, using pre-installed extensions.

The second step of our project is to gather information from computation nodes. The computation of a specific job can imply splitting the data among different nodes: for example FEM simulations decompose the space into elements, which are distributed among the nodes. A representation of decomposition is presented in Figure 3, where each colored area is assigned to a particular node. Thus, for each time-step, it is necessary to gather all the information produced by each node. For this purpose, we use an Apache Flume agent which is in charge of storing information into the VELaSSCo platform.

The third and fourth points concern visualization, and more precisely queries. Visualization has two query layers: the simple one and the complex one.

Simple queries are very similar to traditional information search over Big Data sets. A query has to find a specific subset of information at a specific time-step. This model is well-known and can be efficiently translated into MapReduce jobs. To reduce the complexity of the query model (avoid to define the MapReduce jobs), we use Hive with Tez. But we also have to deal with more complex queries which imply complex computations. For this, specific scripts are developed. Examples of these computations are: extract spline, iso-surface, interpolate information, provide a multi-resolution models, etc. The fourth point concerns the queries rate: queries have to be performed in batch (SQL is well suited for this specific case), but queries can also be triggered dynamically from specific visualization points of view. Displacements of the camera in the 3D space thus produce a queries sent to the platform. For this specific case, we use Storm to stream the data. Different approaches have been proposed in the computer graphic literature, one of them is presented in [11]. This solution presents a continuous multi-resolution method for terrain visualization. Information is sent to the viewer in real-time depending on the camera location. To use efficiently this method, it is necessary to store data using a multi-level approach. In VELaSSCO, we plan to store the data at different resolutions to provide real-time answers to the visualization software. This decomposition of data will be inspired by the method presented by Hoppe in [12], where a base mesh is used to encode all information related to a higher resolution.

The storage architecture of the VELaSSCo platform has to deal with this multi-resolution characteristics and hierarchical decomposition. Moreover, the computational model used to extract information has also to be suitable with these assets. This part of the project is the trickier part, and most of our future contributions will be focused on these specific points.

[9] aws.amazon.com/fr/ec2/
[10] aws.amazon.com/fr/s3/

The last point concerns the extensibility and support. We are looking for an extensible framework which supports extensions for specific usage: queries, data locality and the management of specific storage. Hadoop is the best choice for this purpose. The framework already provides a large set of extensions, and scientific communities continuously provide new contributions. Moreover, this solution is well suited to our needs: we provide a plugin to store data into a partner database named EDM (Express Data Manager). The plug-in is inspired by the solution presented in [9]. The EDM database is an object-oriented database designed to store AP209 standard compliant files. It is a database dedicated to engineering applications.

To summarize the whole VELaSSCo platform is depicted in the Figure 4. It enhances the myHadoop software, with preinstalled plugins. It can be deployed on various IT architectures. This solution has been designed to store data from multiple sources using Flume. We plan to extend the current query engine, and improve it to support complex interactive visualization queries. Another part is dedicated to storage facilities using a specific database system for engineering data. In Figure 4, some extensions are not defined for example: applications and complex queries. The Application component is dedicated to specific computations which run on the storage nodes; for example the computation of multi-resolutions objects. For the complex queries, not all of them have been yet selected, thus the future plugin has not been yet chosen. As stated in this Figure 4, the platform also supports different file systems: HDFS and Lustre for example. We also use the EDM database system, and provide a wrapper between the abstract file system layer in Hadoop and EDM.

IV. Conclusion

We introduce the VELaSSCo project. Simulations produce exponentially growing volumes of data, and it is not possible to store them anymore into existing IT systems. Therefore, VELaSSCo aims to develop new concepts for integrated end-user visual analysis with advanced management and post-processing algorithms for engineering applications, dedicated to scalable, real-time and petabyte level simulations. Data in this project are produced by two simulation sources: DEM and FEM applications. VELaSSCo is a solution to provide a complete platform to answer these needs.

We introduce the architecture of the platform, which is composed of a specific Hadoop distribution related to engineering data processing. The choice was made with respect to some requirements: support complex architectures, support multi-sources aggregation, query lead by visualization, scalability and extensibility. It is be composed of an open-source Hadoop distribution, using myHadoop and preinstalled extensions and scripts for visual queries. We plan to extend the storage by providing a plugin to use the EDM commercial database system as a file system. This software is an engineering database which supports large and complex engineering applications.

Our future work will be mainly focused on complex visual queries on Big Data, and more precisely on real-time streaming queries according to dynamic camera locations.

Acknowledgments

This work was supported in part by the EU FP7 project VELaSSCo, project number: 619439, FP7-ICT-2013-11.

References

[1] C. F. Claver, D. W. Sweeney, J. A. Tyson, B. Althouse, T. S. Axelrod, K. H. Cook, L. G. Daggert, J. C. Kantor, S. M. Kahn, V. L. Krabbendam *et al.*, "Project status of the 8.4-m lsst," in *Astronomical Telescopes and Instrumentation*. International Society for Optics and Photonics, 2004, pp. 705–716.

[2] B. Lange and P. Fortin, "Parallel dual tree traversal on multi-core and many-core architectures for astrophysical n-body simulations," *Euro-Par 2014*, 2014.

[3] W. Dehnen, "A hierarchical o (n) force calculation algorithm," *Journal of Computational Physics*, vol. 179, no. 1, pp. 27–42, 2002.

[4] S. Ghemawat, H. Gobioff, and S.-T. Leung, "The google file system," in *ACM SIGOPS Operating Systems Review*, vol. 37, no. 5. ACM, 2003, pp. 29–43.

[5] J. Dean and S. Ghemawat, "Mapreduce: simplified data processing on large clusters," *Communications of the ACM*, vol. 51, no. 1, pp. 107–113, 2008.

[6] Z. Fadika, E. Dede, M. Govindaraju, and L. Ramakrishnan, "Mariane: Mapreduce implementation adapted for hpc environments," in *Grid Computing (GRID), 2011 12th IEEE/ACM International Conference on*. IEEE, 2011, pp. 82–89.

[7] E. Dede, Z. Fadika, J. Hartog, M. Govindaraju, L. Ramakrishnan, D. Gunter, and R. Canon, "Marissa: Mapreduce implementation for streaming science applications," in *E-Science (e-Science), 2012 IEEE 8th International Conference on*, Oct 2012, pp. 1–8.

[8] M. Isard, M. Budiu, Y. Yu, A. Birrell, and D. Fetterly, "Dryad: distributed data-parallel programs from sequential building blocks," *ACM SIGOPS Operating Systems Review*, vol. 41, no. 3, pp. 59–72, 2007.

[9] S. Krishnan, M. Tatineni, and C. Baru, "myhadoop-hadoop-on-demand on traditional hpc resources," *San Diego Supercomputer Center Technical Report TR-2011-2, University of California, San Diego*, 2011.

[10] A. Abouzeid, K. Bajda-Pawlikowski, D. Abadi, A. Silberschatz, and A. Rasin, "Hadoopdb: An architectural hybrid of mapreduce and dbms technologies for analytical workloads," *Proc. VLDB Endow.*, vol. 2, no. 1, pp. 922–933, 2009.

[11] P. Lindstrom, D. Koller, W. Ribarsky, L. F. Hodges, N. Faust, and G. A. Turner, "Real-time, continuous level of detail rendering of height fields," in *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*. ACM, 1996, pp. 109–118.

[12] H. Hoppe, "Progressive meshes," in *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*. ACM, 1996, pp. 99–108.

Graph Database for Agent Based Emergency Response Model

Safiye Sencer
Sakarya University
Sakarya, Turkey
Email:safiyesencer@yahoo.com,
{sencer@sakarya.edu.tr}

Kutlu Eren
Sakarya University
Sakarya, Turkey
Email: kutlu.eren26@gmail.com

Abstract **-** Developing computer technology provides to use the information system commonly. All to gather, big data processing is growing importance. Moreover to operate distributed and dynamic environments are inevitable in real life to realize process in a short time. Decision-making processes among autonomous agents can support to solve dynamic and large system problems. This paper presents an agent based emergency response coordination model that considers the graph data model and associated query language provides a unifying conceptual model. We used the Neo4j which is Cypher graph query language.

Keywords: Graph databases, Source-code queries, Multi agent system, Emergency Response Model

I. Introduction

Graph database uses the graph structures with nodes, edges and properties to represent and store data. It is a storage system that provides the index-free adjacency. General graph databases that can store any graph are distinct from specialized graph databases such as triple stores and network databases. Edges are the lines that connect nodes to nodes or nodes to properties and they represent the relationship between the two. Most of the important information is really stored in the edges (Neo4j, Mens; 2005, 2007). Meaningful patterns emerge when one examines the connections and interconnections of nodes, properties, and edges. In this study, emergency response model designed with graph database approach.

Emergency response model covers the nature events and the non-nature events. In particular, unexpected events cause the turmoil among the people most of time. To prevent the turmoil and to save their life in every time, in particular depend on the emergency responses. Appropriate responses are needed in the form of allocating resources to handle the effects of emergency responses. The form and remedy kind may help to people easily kind of the unexpected event. Also the early caution systems can able to stimulate people to this kind of events.

Determination of the unexpected events type is so important for help the people. Every time, people faces to nature events such earthquake, forest fires, terrorist attacks, war threads. Also to operate distributed and dynamic unexpected environments are inevitable and so important in real life to realize process in a short time. Multi-agent emergency response system has been extensively used in the different tasks of decentralized emergency response problem solving such as communication among agents, collective decision making, cooperation, collaborative planning in large scale that deals with uncertainty and conflicting information during emergency response management.

The paper is organized as follows: the next section presents an overview of works in the literature related to graph database approach, emergency response structure, multi agent systems dynamic modeling; the next section introduces overall architecture which attains the proposed aims and emphasizes the roles played by the agent based emergency system components; and describes the complete working flow and details theoretical approach on which relies the work; then, the next describes the process model applied to emergency response model case study. Conclusions and future works close the paper.

II. LITERATURE SURVEY

The graph data model unifies queries cross-cutting over various representations of source code, also demonstrate a prototype implementation based on Neo4j. It stores full source-code information and scales. The graph data model encodes entities and relationships amongst them using a directed graph structure (Angles, Gutierrez, 2008). It consists of sets of nodes and binary edges (and hyperedges but we do not use these here) representing entities and relationships between these entities. Nodes and edges can be labelled, typically with the name of the entity

they represent, but possibly with a tuple of name and further values.

Graph databases provide to combine relations based on a common attribute to retrieve connected records. It is faster and this potential source of inefficiency that is addressed in some graph databases which provide index facilities to map certain structures known ahead of time to a list of records (Vicknair et al. 2010). In addition, because graph databases do not depend on a schema, they can be more flexible when the structure of records needs to be changed.

Emergency management is so important for the sustainable qualify life conditions. In particular, to take precautions for unexpected emergency situations are so important for the human life. MAS may use in emergency situations resulting from natural and human made emergency responses, such as flood, tsunami, earthquake, terrorist attack, fire in building etc, represent complex and dynamic environments with high level of uncertainty; hence autonomous notification and situation reporting for emergency response management system will be done by multi-agent response system. In literature, Wang et al. (2013) suggested the emergency management response system structure for a city in China. Also, Basak (2011) et al., suggested the agent based disaster management and Chou et al. 2008, suggested the dynamic parking structure with agent based platform. Belief Desire Intelligence (BDI) is very important architecture for offer an agent with artificial capabilities. Some of the emergency response response systems have been developed based on multi-agents systems approach (such as: DrillSim developed by Balasubramanian et al. 2006, DEFACTO designed by Marecki et al. 2005, ALADDIN modeled by Adams et al. 2008 and Jennings et al. , RoboCup Rescue suggested by Kleiner et al. 2005, and FireGrid proposed by Berry and Usmani in 2005) and more are being developed. Our study covers the behavioral response structure which is influenced by objective (cognition or abilities) and subjective (feelings or reflexes) processes. It focuses on objective processes which may influence the behavioral response.

The autonomy is an ability of agent to achieve its goals without any supporting from other agents. On the other hand, the interaction of agents to get the global goal of the system is the social ability of agents. The reactivity, which is based on the relation between perception and action, is an ability of agents to respond to the environmental changes. The pro-activeness of agents is an ability to express the goal-directed behaviors. The reactions of agents to the environmental changes are the reactivity or pro-activeness that depends on what kind of architecture of agent is used to develop agents. Intelligence is the ability of the agent using its knowledge and reasoning mechanisms to make a suitable decision with respect to the environmental changes.

III. AGENT BASED EMERGENCY MODEL

This section includes overall architecture which achieves the proposed aims and emphasizes the roles played by all system components; and describes the complete working flow and details. Theoretical approach on which relies the work. The next subtitle of the Agent Based Model gives the detail of the emergency responsive model with components.

3.1 The Cycle of the Agent Model

Our suggested model could be done - situation assessment; understanding context surrounding; communication; collaboration properties. The suggested model is able to realize communication and collaboration with coordination, coalition and collaborative information distribution properties. Agents communicate in order to achieve better the tasks of them or of the society/system in which they exist. Communication can enable the agents to coordinate their actions and behavior, resulting in systems that are more balanced. For communication, agents are able to coordinate, coalition and collaborative information each other agents.

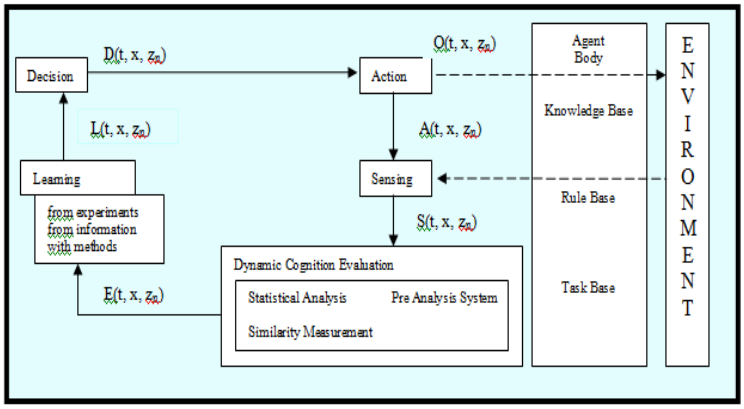


Figure1. Dynamic modeling of the abstract system

3.2 Emergency Response Agent Based Model

In this paper, the current emergency response management and response systems are analyzed and taking in consideration of domain requirements, agent-based design methodology and systems' comparative view. Suggested agent model covers the some parts, which are collective information, coordination, communication, coalition, cooperative distributed information sharing, problem solving and evaluation, decision making. Contract Net (CNET) is preferred to achieving efficient cooperation through task sharing in networks (Weiss, 2012) as well as used in Emergency response System for communicating problem solvers.

Multi-agent emergency response system includes the different tasks of distributed emergency response problem solving such as collective decision making, communication among agents, cooperation, collaborative planning in large scale that deals with uncertainty and conflicting information during emergency response management (Adams et al.). In detail, the suggested type of emergency response systems can be viewed on information and knowledge fusion and take the feedback from the existing agents for sensing, coordinating, decision making and acting. It must be able to achieve these objectives in environments in which: control is distributed; uncertainty, ambiguity, imprecision; multiple agents with different aims and objectives are present; and resources are limited and vary during the system's operation.

The suggested model may provide some benefits which are); more robust, interoperable, and priority sensitive communications, better situational awareness, improved decision support and resource tracking, greater organizational agility, better engagement of the public. Emergency response model covers the very wide area respects of the nature events (earthquake, floats, fires, hurricane, and thunder storms), terror events (considering of the coming information), fault of the people or devices (accidents, fires, explosions etc.).

Knowledge Base Agent: It includes the natural events, terror events and accidents, also natural events include the earthquake, flood, fire, hurricane, statistical data and current data. The knowledge base agent takes data from environment observation. Also system inputs connected with the knowledge base with simultaneously coming data information, historic data and weather data. At the same time, learning process shares the data with system inputs and knowledge base. It covers the database, which is related to the system components. It provides the using of the information whole system.

In order to deal with a lot of data, connection between stored data and making decisions on collected data, a graph, schema less database has been selected. In graph databases, every element contains a direct pointer to its adjacent elements and no index lookups necessary. It allows analyzing the data fast. Having schema less structure helps to store a lot of useful data on the nodes. Storing data this way allows making suggestions for action plan. In knowledge based agent, the applied area is stored as City, County, District, Street. These areas can be assumed as nodes and the connection between them locates the hierarch, as mentioned from right to left. Cities will also be connected to each other so the model can be completed and therefore search will be easier in the system. The applied area and event type modeled in Figure 2 and 3.

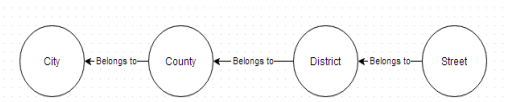


Figure 2 Main structure of the location

Thus the system will be able to store an event by address. Also events types will be nodes such as

Figure 3 Main structure of the event types

When an event occurs or gets reported, the system will take information and store the event according to the given address. Figure 4 shows how it gets stored.

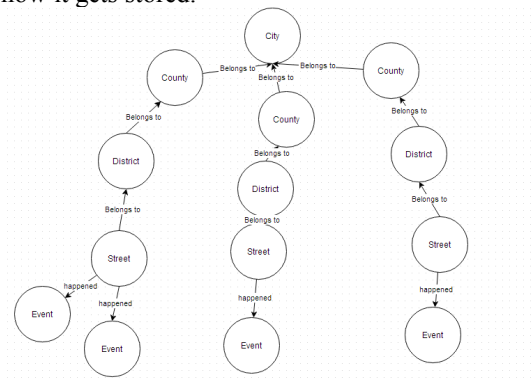


Figure 4. Taking information and store events diagram

The events are connected to the event types and also has a response action. From an event type, events in different cities can be reached and response actions can also be recommended for another event. Figure 5 shows the event and event type structure.

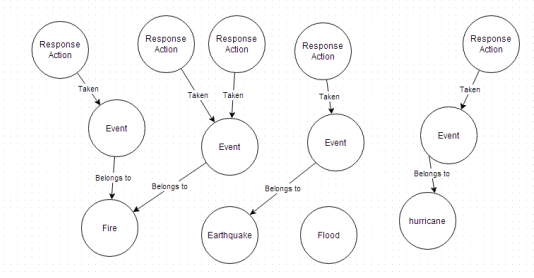


Figure 5 Event and event type structure

By storing the data in this structure, we are able to find events occurred in a street, district, county, city, what kind of actions has been taken, information about those taken actions. We can also suggest response actions for event according to its type as event types connected to different events and those events might have similar response actions. This structure will allow the system travel across the country easier and faster. By mean faster, the response will be quicker to resolve the event.

Performance Estimation Agent: It includes the information and coming data information. Processing of the information provides the coming data analysis and evaluation with proper decision making structure.

Service Provider Agent: The agent includes the service system which includes the ambulance service, logistic service, police linked service, combined emergency service, hospital based service.

Decision Making Agent: It includes the system outputs which are prediction of the event type, prediction of the location, prediction of the correct resource usage and correct resource coordination.

The system goal in this agent-based coordination network between emergency support system and people who are needs to support is divided into three parts: defining the initial negotiation strategy, considering the emergency assessment factors, and finding the cost functions and report results.

The architecture of an agent shown in Figure 6 consists of five modules: perception, decision making, knowledge, control, and communication. The perception module is responsible for data acquisition in the environment. The decision making module is in charge of the agent making a decision in an autonomous way.

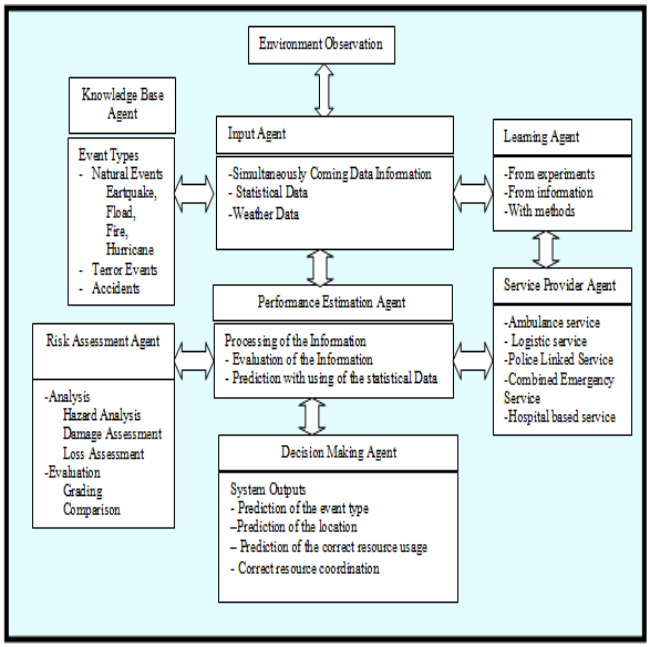


Figure 6 Agent Based Emergency Response Model

The control module processes the plan into tasks and executes the tasks to the environment. At the same time, this module sends the tasks to the communication module if the plan is processed by the agent team. The communication module is responsible for interactions between the agent and the agent community. It receives or sends messages, interprets them and transmits the tasks of the control module to other agents. The knowledge module contains intentions, and plans of the agent.

Control module is so important for the decision making agent. It covers the update, process of the task, evaluate of the alternatives, compare, learning, announce, negotiate, notify, request, notification and computing. The system considers the Table 2 control activities.

B	: Update (B, s);
D	: Process(task)
E	: Evaluate (alternatives)
T	: Compare(B,D,E)
L	: Learning (situation)
A	: Announce
Ne	: Negotiate
N	: Notify
R	: Request notification
C	: Computing

Table 2 Summary of the activity list in control module

The case study covers the some steps which are system analysis and design; building of conversation; system processes.

3.3 System analysis and design

The coordination network covers the system component, experts and people. FIPA –Contract-Net-Manager protocol is applied to analyze and design the system. Finally, the pattern of this system is built and illustrated. This study emphasizes the coordination of the network emergency assist system and who needs the assist one. The agent mechanism structure can be applied to establish such a negotiation system with the distinction from the emergency information system, and at the same time based on research as shown in the literature review.

3.3.1 System analysis

The agent can complete complicated negotiation with related agent. Agents work automatically without rest on coordination, enabling the drivers to seek out. As the agent architecture, the FIPA – Contract-Net-Manager protocol is preferred.

.

3.3.2 Agent classes

In negotiation, the agents engage in dialogue, exchanging proposals with each other, evaluating other agents' proposals and modifying their own proposals and then modifying their own proposals until all agents are satisfied with the set of proposals. Standard negotiation mechanisms adopted are based on game theory or on human-inspired negotiations. Every task also defines higher level, complex interaction protocols requiring coordination between multiple agents. First, the system checks emergency assist opportunities information for user when he inputs data into the system, and sends a message to emergency support agent. Second the user agent negotiates with the user agent through to emergency assist agent. The bidding continues until the driver agent accepts one bid or rejects them all.

3.3.3 Tasks

A task is a structured set of communications and actions. The ovals denote tasks that the role must execute in order to accomplish its goal. These concurrent tasks are defined as a finite state automation specifying messages sent between roles and tasks. The lines between nodes indicate protocols between tasks, which define a series of messages between the tasks that allow them to work cooperatively.

In this step, an agent class diagram is created, as depicted in Figure 6 from the viewpoint of the roles, documented. The agent class diagram depicts agent classes as boxes, and the conversations among them as lines connecting the agent classes. Four agent classes are defined: user agent; assist agent; information agent; decision making agent.

3.3.4 Roles

- Reporting any situation that requires a police officer at the scene (e.g. assaults, traffic accident, burglary report, damage to property, parking complaint, other ordinance violations, etc.)
- Calling an ambulance for medical assistance.
- Reporting fire, smoke or fire alarm.
- Reporting a crime in progress.
- Reporting suspicious or criminal activity. (shouts for help, glass breaking, vehicle or person that does not appear to belong in neighborhood).

3.3.5 Sequence Diagrams

A use case is a narrative description of a sequence of events defining desired system behavior. A sequence diagram depicts a sequence of events between multiple roles and defines the minimum communication between roles. Use cases can be drawn from the system requirements and users. Then the use cases can be restructured into sequence diagrams. The proposed system has five main sequences of events, which are described in the following by use cases and sequence diagrams.

Refining roles: The third step is to ensure that all the necessary roles have been identified, and to develop the tasks that define role behavior and communication patterns. Through applying the use case step, the roles of the proposed system have been defined roughly, so in this step they are refined, and tasks associated with each role are created. Fig.4 illustrates a agent architecture role model.

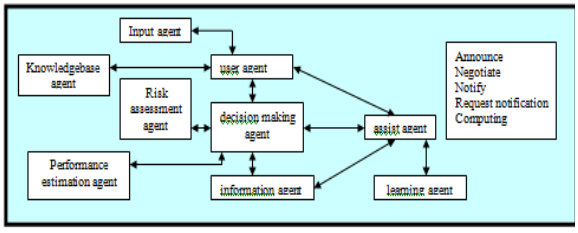


Figure 4 Agent based architecture

3.5.6 Conversations

With negotiation, the agents engage in dialogue, exchanging proposals with each other, evaluating other agents' proposals and then modifying their own proposals until all agents are satisfied with the set of proposals. Standard negotiation mechanisms adopted are based on markov chain approach and human inspired negotiations.

In negotiating phase, the roles which are emergency support agent, decision making agent, user agent includes the main messaging system. A communication diagram is a pair of finite state machines defining a conversation between two participant agent classes. The syntax of the communication class diagram is very similar to that of the roles include emergency response parameters.
The assist agent realizes the negotiation as follows:

- The initiator agent estimates the minimum cost and time resource point
- The respondent agent proposes the bid lately
- The initiator agent proposes the minimum cost
- The initiator agent proposes the minimum distance

The suggested model realizes the following steps:
Step 1: Determine the asking help choices and the assist alternatives
Step 2: Revise asking resource slightly
Step 3: Diagnose whether exceeding the end time and distance
Step 4: Consider the types of messages
Step 5: Control of the limitations
Step 6: Evaluate the final situation

The agent architecture defines the configuration of the system to be implemented. The overall system architecture is defined by deployment diagrams. The proposed system is divided into two subsystems. Subsystem 1 includes four agents: agent; assist agent; information agent; decision making agent that provides the negotiable spaces of the dynamic behavior. Also some of the system components uses the some input data such as calling system people's voice tone like calm, angry, excited, slow, rapid, soft, loud, vulgar, laughing, crying, normal, distinct, slurred, intoxicated, nasal, stutter, lisp, raspy, ragged, clearing throat, deep breathing, cracking voice, disguised, accent, electrically altered, familiar, rational, irrational; background voice frequency like, airport, animal noises, baby, clear, local, school, factory machinery, office machinery, restaurant, television, house noises, motor, music, street noises, kids, traffic, long distance, party.

3.5.7 System process

In multi agent decision making, the agents utilize classical artificial intelligence decision making methods to decision making their activities and resolve any foreseen conflicts. Emergency response criteria types are listed in Table 4.

Alternatives	Event Type	Event Distance	Emergency Size	How many people affected	Location

Table 4 Emergency response criteria types

A graph simulation has been created using Neo4j.The simulation model considers the just only fire events in Istanbul, Sisli.

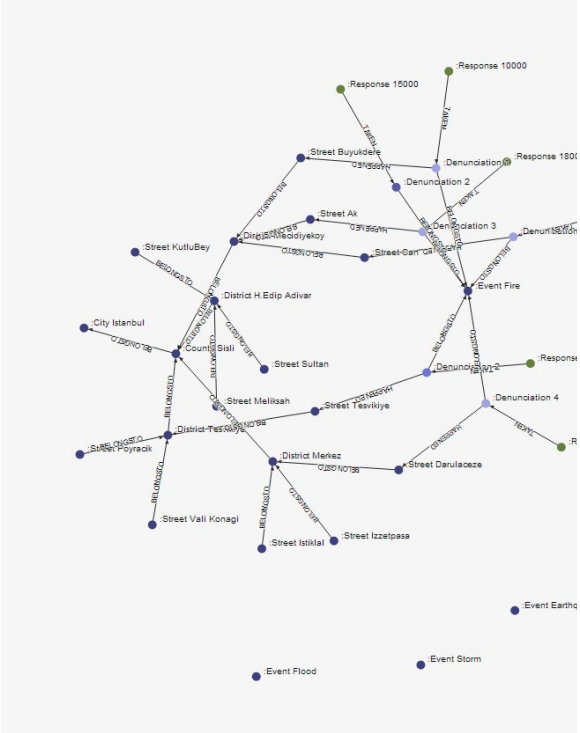


Figure 5 Designed model with Neo4j

In the simulation, Istanbul (Turkey) and Sisli (a county belongs to Istanbul) has been selected. There a lot of district in Sisli but some of them used in the simulation. As shown in the image above. There are event types, streets connected to the districts, district connected, district connected to the city. Denunciations occurred at random dates are connected to the streets as the main purpose is to store them by detail address (in Figure 5). There are responses taken for denunciations, and those denunciations connected to the events. For this simulation, it is only focused on fire event.

From the simulation, system is able to interpret the inputs. System now can find fire events occurred in Sisli, responses to those events. System now might evaluate the input, look for similar response and if there is a match then it could be use as new response to that event (Figure 6).

STEP 1: Fire events from system

```
START n=node(*)
MATCH r-[rel:BELONGSTO]->n
WHERE n.name='Fire'
RETURN r
```

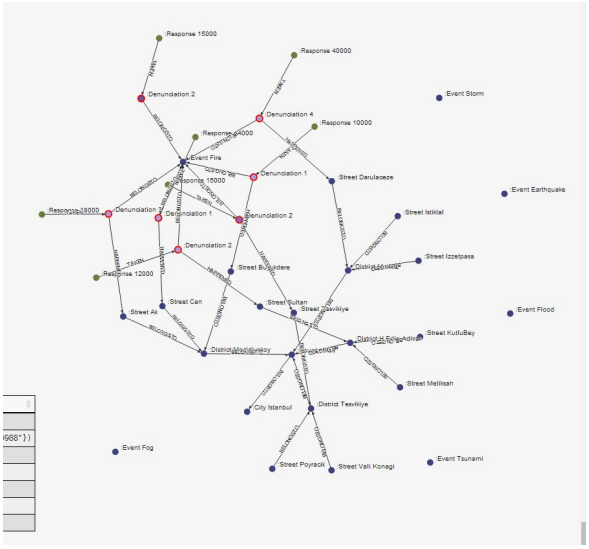


Figure 6 Sisli fire event simulation model

STEP 2: Denunciations happened in Sultan Street (in Figure 7)

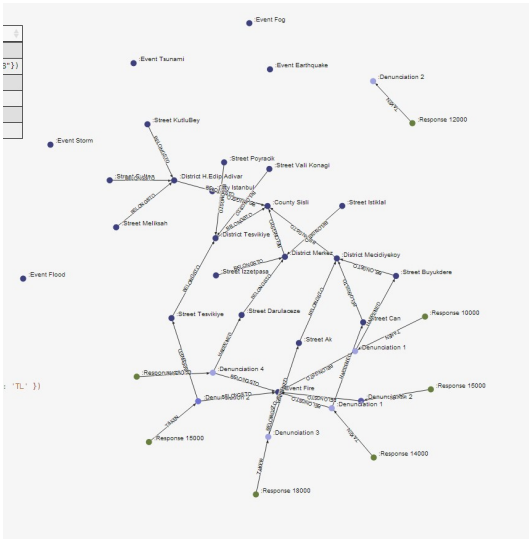


Figure 7 Denunciations happened in Sultan Street

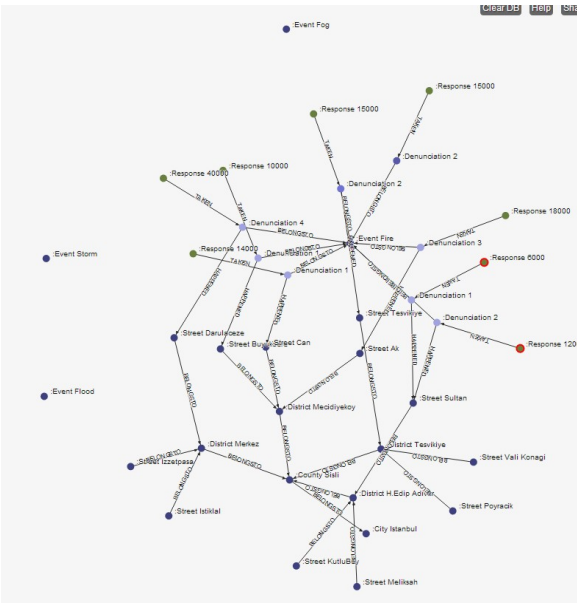


Figure 8 Sultan Street and taken responses model

STEP 3: Denunciations happened in Sultan Street and taken responses

```
START n=node(*)
MATCH m-[:TAKEN]->r-[rel:HAPPENED]->n
WHERE n.name='Sultan'
RETURN m,n
```

Figure 9 Sultan Street and taken responses model

Future possibilities

- Fire events can be count, a density map can be made from the evaluation and response plans might be developed for a specific city, county, district or street.
- System might find similar response to incoming denunciation
- A lot of scalable data can be stored
- System can learn from scalable data and generate responses
- Search and response action will be fast
- A visual map can be developed for a better understanding

Attributes can be specified and then a ordered weight function can be constructed to evaluate utility values of different solutions. The solution with the best utility will be selected. Each agent keeps a driver's decision hierarchy as multi-attribute preferences. The people who have some problems that related about the

1. System have to determine the problem class
2. After classification, system has to find solution about the problem in a short time and short distance to someone who needs to emergency assist.
3. Emergency assist agent system has to making decision with the all of the components.

IV. PERFORMANCE ANALYSIS

The suggested system considers the different parameters for the evaluation of the information for selection of the best choice. Neo4j is an open-source database, which implemented in Java. It is able to realizes 2^35 (~ 34 billion) nodes, 2^35 (~ 34 billion) relationships, 2^36 (~ 68 billion) properties, 2^15 (~ 32 000) relationship types. 10 Comparison of the neo4j and mysql performance with the vertice returned based on the time is given in Figure 10 and Table 11.

Size-wise Neo4j handles graphs of sizes of several billion nodes, relationships and properties out of the box. It is normal to reach read-performance of 2000 relationship traversals per millisecond (about 1-2 million traversal steps per second) fully transactional, with warm caches, per thread. With shortest-path-calculations, Neo4j is even on small graphs of a couple of 1000 of nodes 1000 times faster than MySQL, the difference increasing as the size of the graph increases.

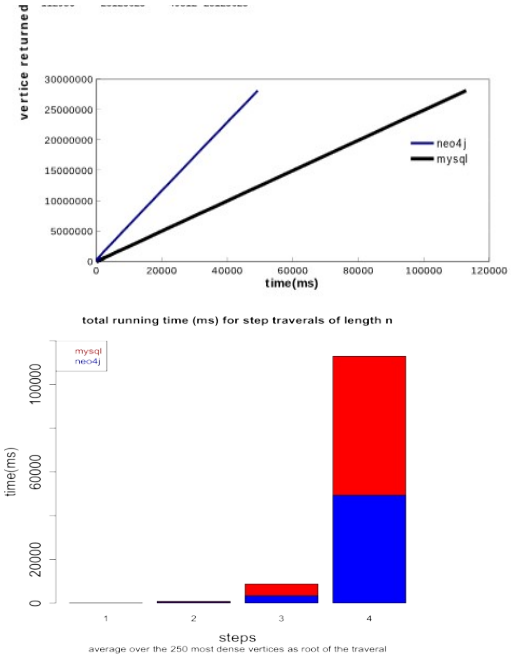


Figure 10 Comparison of the neo4j and mysql performance based on the vertice returned based on the time

mysql		neo4j	
time(ms)	vertice retu	time(ms)	vertice returned
124	11360	27	11360
922	162640	474	162640
8851	2206437	3366	2206437
112930	28125623	49312	28125623

Table 5 Neo4j and mysql performance value

V. Conclusion

This paper discusses agent based emergency response model with an active multi agent database system which incorporates active rules in a multi computing environment. This system helps people to reach emergency remedy resources easily. Finally, due to frequent changes in the positions and status of objects in an active mobile database environment, the issue of temporality should be considered by adapting the research results of temporal database systems area into active mobile databases.

This paper gets the agent-based simulation and determines an optimal plan to emergency response model in the shortest time possible with neo4j simulation model. Agent simulation for emergency response model improves upon other simulation models that are concerned with numerical analyses of inputs or amounts of people and structures. The agent-based system for emergency response model is grounded on empirical data taken from real-world experiments. If the agent sees an exit, it will proceed towards it and if it receives any types of direction to leave, that will be carried out without failure. Further study includes the improvement of the text mining techniques with new respect. Also agent based emergency response model provides to evaluate uncertain and vagueness information.

References

Adams M., N.M., Field, E. Gelenbe, D.J. Hand, N.R. Jennings, D.S. Leslie, D. Nicholson, S.D. Ramchurn, S.J. Roberts, A. Rogers, "The ALADDIN Project: Intelligent Agents for Emergency response Management".

Adams, M. Field, E. Gelenbe, D. J. Hand, "The Aladdin Project: Intelligent Agents for Emergency response Management - IARP/ EURON Workshop on Robotics for Risky Interventions and Environmental Surveillance", 2008.

Angles, R., Gutierrez, C., Survey of graph database models, Compt. Surv., 40(1) (2008), p. 1

Balasubramanian , Massaguer, Mehrotra , Venkatasubramanian, "DrillSim: A Simulation Framework for Emergency Response Drills"; Proc. of ISCRAM, 2006.

Basak, S., Modanwal, N., Mazumdar, B.D., Multi-Agent Based Disaster Management System: A Review, IJCST Vol. 2, Iss ue 2, June 2011.

Berry, D., Usmani, "A. FireGrid: Integrated Emergency Response and Fire Safety Engineering for the Future Built Environment"; UK e-Science Programme All Hands Meeting, Nottingham, UK, Sept. 19-22, 2005.

Chou, S.Y., Lin, S.W., Li, C.C., Dynamic parking negotiation and guidance using an agent-based platform, Expert Systems with Applications, Vol.35, Is. 3, 2008, pp 805-817.

FIPA, ''Specification Part 2:Agent Communication Language,'' The text refers to the specification dated 23 October 1997.

Java 8: support for more aggressive type-inference, http://mail.openjdk.java.net/pipermail/lambda-dev/2012-August/005357.html.

Jennings, N., Gopal Ramchurn, Mair Allen-Williams, Raj Dash, Partha Dutta, Alex Rogers, Ioannis Vetsikas; "The ALADDIN Project: Agent Technology to the Rescue".

Nick Jennings, Gopal Ramchurn, Mair Allen-Williams, Raj Dash, Partha Dutta, Alex Rogers, Ioannis Vetsikas; "The ALADDIN Project: Agent Technology to the Rescue".

Khaled M. Khalil, M. Abdel-Aziz, Taymour T. Nazmy, Abdel-Badeeh M. Salem, "Multi-Agent Crisis Response systems – Design Requirements and Analysis of Current Systems".

Kleiner, B. Steder, C. Dornhege, D. Höfer, "RoboCupRescue - Robot League Team RescueRobots Freiburg (Germany)"; RoboCup (Osaka, Japan), 2005.

Marecki, N. S., Tambe, M., "Agent-based Simulations for Emergency response Rescue Using the DEFACTO Coordination System"; Emergent Information Technologies and Enabling Policies for Counter Terrorism, 2005.

Mens, T., Van Eetvelde, N., Demeyer, S., Janssens, D., Formalizing refactorings with graph transformations, J. Softw. Maint. Evol.: Res. Pract. 17 (4) (2005) 247–276.

Mens, T., Taentzer, G., Runge, O., Analysing refactoring dependencies using graph transformation, Softw. Syst. Model. 6 (3) (2007) 269–285.

Neo4j graph database, http://www.neo4j.org/.

RoboCup project, [Online] Available: http://www.robocup. org/

Vicknair, C., Macias, M., Zhao, Z., Nan, X., Chen, Y., Wilkins, D., A comparison of a graph database and a relational database: a data provenance perspective, Proceedings of the 48th Annual Southeast Regional Conference, ACM (2010), p.42.

Wang, D.Y., Pan, L.W., Lu, L., Zhu, J.P., Liao, G. X., Emergency Management Business Process Reengineering and Integrated Emergency Response System Structure Design for a City in China, Procedia Engineering 52 (2013) 371 – 376

Blur Detection For Video Streams In The Compressed Domain

Zhenyu Wu[1], **Daiying Zhou**[1], **and Hong Hu**[2]
1 University of Electronic Science and Technology of China, Chengdu, Sichuan, China
[2] Huawei Technologies Co. Ltd, China

Abstract - *In ubiquitous multimedia area, the number of digital videos increases dramatically with various qualities in video frames. Artifacts such as blur may commonly exist in video treams which will disturb compression and retrieval applications. This paper proposes an algorithm that detects video frames with global blur and partial blur. The proposed algorithm extracts the features which represent blurs according to analyses of DCT coefficients characters and motion vectors information firstly. And then detects blurs by suitable model efficiently. The method presented in this paper is low in complexity while high in performance. Experimental results demonstrate the high-accuracy global and partial blurs detection of the proposed scheme*.

Keywords: blur detection; DCT; video stream; compression;

1 Introduction

Digital video data is increasing dramatically with ubiquitous smart phones and personal cameras. However, due to the lack of expertise and performance of cameras, some of video frames are in poor quality, especially in sports streams. One of the most common afflictions is blur, and more specifically, motion blur. Automatic blur detection is highly desirable. It is one of video streams' pre-processing operations, which can benefit for video editing, video retrieval and video compression. Blur detection shall judge whether or not a given video frame is blurred and determine to what extent the video frame is blurred. It may help editors to restore those blur frames or simply discard them. Meanwhile, in massive video retrieval field, blur detection can improve the performances of key frame extraction and important features abstraction. On the other hand, in video compression filed, intelligent encoding according to video content has been studied more and more deeply. Blur detection can note those blur parts as the unimportant frames or blocks. Under above direction, the intelligent encoder can avoid to encode those blurred frames as intra frames or forward-predicted frames, and try to compress those blurred blocks with larger quantization steps in the case of limited bandwidth. So the compressing performance will increase greatly and the delivery of video streams in wireless network among ubiquitous digital devices shall become more efficient.

Similar with images, blur detection methods of images can be borrowed by video frames. Although the topics of image blur analysis have attracted much attention in recent years, most work focuses on solving the deblurring problem. As far as we know, few research efforts have been made to detect general blur up to now and they are still far from practical.

One kind of methods is based on model estimation [1]-[3], which takes blind deconvolution to form blur phenomenon. It is too complicated to implement blind deconvolution into blur detection, since general blurs are caused by quite different reasons and difficult to model.

Another kind of methods is to detect blur by comparing image features' differences [4] such as edge sharpness, local saturation, power spectrum slope, DCT coefficients distribution and gradient magnitude distribution. However these methods usually need cleared images as references to determine whether or not the blurring occurs.

Moreover, video frames are different from single images in some aspects that no cleared frames to refer. So the methods based image features' differences comparison cannot be adopted directly in video frames' blur detection, since video frame's features varies greatly with frame contents changing. For example, Fig. 1 demonstrates the gradients magnitude distribution of different video frames. Studying it, we can see that the blurred and clear frames with different contents may have similar gradient magnitude heavy-tail distributions.

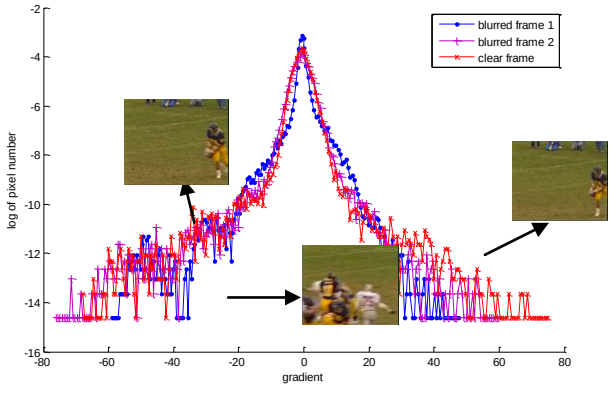


Fig. 1 comparisons of blur and clear frames' gradient magnitude distributions (blue and purple figures are blurred frames, red one is clear frame)

In this paper we propose a new blur detection scheme using DCT and motion vectors which does not require clear frame to refer or need to estimate blur models by complicated blind deconvolution operation. It seamlessly combines discrete cosine transform and motion estimation operations to detect blur of the whole frame or every macro-block adaptively. The proposed scheme takes advantage of DCT coefficients in direct features representation, edge sharpness etc. detailed information description, nonzero DCT coefficients' distribution and power ratio of low DCT components with high components cleverly. It is effective for global blur detection. According to motion vectors, the proposed method can track moving objects coarsely. So it can detect partial blur efficiently meanwhile. The whole detection processing is completed in the DCT domain with ready DCT coefficients and motion vectors in video streams. So our method is also with high computational efficiency.

The rest of the paper is organized as follows: Section II gives a brief describes of related work in image and video blur detection. In section III, we present our scheme in detail. Experimental results are given in section IV. Finally, we conclude the whole paper in section V.

2 Related Work

Image blur detection was a by-product of deblurring at first time. The degradation of images is mathematically modeled as:

$$g(x,y) = f(x,y) * h(x,y) + n(x,y) \quad (1)$$

where g(x,y), f(x,y), h(x,y) and n(x,y) represent the degraded image, original image, blur point spread function (PSF) and noise. Assuming that noise is negligible, the degradation is simplified into a pure convolution process. Most previous work focuses on the estimation of the blur kernel and the corresponding image de-convolution procedure for a particular type of blur. Gordana[3] proposed a formulation for maximum likelihood (ML) blur identification based on parametric modeling of the blur in the continuous spatial coordinates.

General blur detection and recognition is relatively less explored. By examining the absence of alternating component (AC) coefficients which indicate the edge sharpness, Marichal[5] proposed a approach to characterize blur extent qualitatively according to DCT information. However, it fixed weighting grid for the blur measurement applied on the DCT coefficient by highlighting diagonal direction, which is lack of accuracy since direction features of DCT coefficients are quite different. Described in [6], Zhu etc. analyzes the distribution of gradient magnitudes, and then proposes a fast and effective blur detection based on heavy-tailed distribution. Since the heavy-tailed distribution of gradient is completely dependent on image contents, a gradient magnitudes distribution of clear background image is needed as a reference to do judgment. It is not suitable for blur detection without clear reference image or video frame, such as non-surveillance video case. Tong[7] takes Harr wavelet transform into blur detection to analyze edge type and sharpness of whole images. It can determine whether an image is blurred or not and to what extent an image is blurred. However, although it can extract the overall directions such as horizontal, vertical or diagonal, it misses the local features. So, although it can detect images' global blur, but it is powerless in partial blur case. Furthermore, without considering the temporal information, this method is not efficient in video blur detection. Karl[8] improves Tong's work by adding temporal information into single image blur metric. However, since the temporal information added is just the whole frame's DWT feature differences, Karl's method cannot detect partial blur either. So it is not suitable for blur detection in sports video streams, which have both global and partial blurs.

3 Proposed Blur Detection scheme

In general, clear video frames always have sharpness edges and rich details. When blur occurs, no matter what kind of blur is, the edges will disappear or lose their sharpness and details will lose their richness. Since DCT coefficients are correspondent with video frames' pixel values, the characters of DCT coefficients will change when blur occurring. The basic idea of our scheme is to detect blurs by analyzing the DCT coefficients' characters changing.

This section studies the relationship of DCT coefficients with direction features and edges firstly in subsection 3.1. After that we describe how to design adaptive DCT weighting matrixes to detect blur according to the importance levels of every DCT coefficients in subsection 3.2. In the following step, we add temporal information extracted by motion vectors to track every macro-block in neighbor frames to determine blur-threshold in subsection 3.3. Finally, we design an efficient global and partial blur detection scheme by the elements obtained in section3.1-3.3.

3.1 DCT coefficients' characters analysis and blur effect

Before analysis, we briefly describe the DCT model of compression. The original video frame is transformed using DCT-II to frequency domain on a block by block base. The standard block size can be 4x4, 8x8 and 16x16. After DCT, quantization is performed on transform coefficients to discard visually unimportant ones. This is a lossy process and some compression is achieved in this step. Discrete cosine transform (DCT) is the heart of this compression scheme. It is defined as follows (here takes 8x8 DCT in general):

$$F(u,v) = \frac{C_u C_v}{4} \sum_{i=0}^{7} \sum_{j=0}^{7} \cos\frac{(2i+1)u\pi}{16} \cos\frac{(2j+1)v\pi}{16} f(i,j) \quad (2)$$

Where

$$C_u, C_v = \begin{cases} \frac{1}{\sqrt{2}} & \text{for } u,v = 0 \\ 1 & \text{otherwise} \end{cases}$$

One simple observation is that each DCT coefficient $F(u,v)$ is a linear combination of all pixel values within the block. Every DCT coefficient has certain relationship with the pixels' values in a block. For example, the coefficient in the upper left corner of a DCT encoded block is the DC coefficient and it represents the average luminance of the block. The remaining coefficients are all called AC coefficients and each of them reflects variations in gray level values in certain direction at certain rate. To study this relationship, let us consider the coefficient $F(1,0)$ and $F(0,1)$ for instant.

$$\begin{aligned} F(1,0) &= \frac{C_1 C_0}{4} \sum_{i=0}^{7} \sum_{j=0}^{7} \cos\frac{(2i+1)\pi}{16} f(i,j) \\ &= \frac{C_1 C_0}{4} \sum_{i=0}^{7} \cos\frac{(2i+1)\pi}{16} \sum_{j=0}^{7} f(i,j) \end{aligned} \quad (3)$$

Since $\cos(\pi - \theta) = -\cos\theta$, (3) can be expanded as

$$\begin{aligned} F(1,0) &= \frac{C_1 C_0}{4} \left[\cos\frac{\pi}{16} \left(\sum_{j=0}^{7} (f(0,j) - f(7,j)) \right) \right. \\ &+ \cos\frac{3\pi}{16} \left(\sum_{j=0}^{7} (f(1,j) - f(6,j)) \right) + \cos\frac{5\pi}{16} \left(\sum_{j=0}^{7} (f(2,j) - f(5,j)) \right) \\ &\left. + \cos\frac{7\pi}{16} \left(\sum_{j=0}^{7} (f(3,j) - f(4,j)) \right) \right] \end{aligned} \quad (4)$$

Study (4), we can see clearly that the value of F(1,0) essentially depends upon intensity difference in the vertical direction between the upper and lower parts of the input block. Similarly, F(0,1) can be got by (5) and its value essentially depends upon intensity difference in the horizontal direction between the left and right parts of the input block.

$$\begin{aligned} F(0,1) &= \frac{C_0 C_1}{4} \sum_{i=0}^{7} \sum_{j=0}^{7} \cos\frac{(2j+1)\pi}{16} f(i,j) \\ &= \frac{C_0 C_1}{4} \sum_{j=0}^{7} \cos\frac{(2j+1)\pi}{16} \sum_{i=0}^{7} f(i,j) \\ &= \frac{C_0 C_1}{4} \left[\cos\frac{\pi}{16} \left(\sum_{i=0}^{7} (f(i,0) - f(i,7)) \right) + \cos\frac{3\pi}{16} \left(\sum_{i=0}^{7} (f(i,1) - f(i,6)) \right) \right. \\ &\left. + \cos\frac{5\pi}{16} \left(\sum_{i=0}^{7} (f(i,2) - f(i,5)) \right) + \cos\frac{7\pi}{16} \left(\sum_{i=0}^{7} (f(i,3) - f(i,4)) \right) \right] \end{aligned} \quad (5)$$

Based on above analysis, the physical meanings of 6 most important DCT coefficients are shown in Fig. 2.

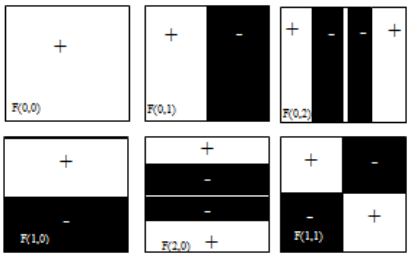


Fig. 2 Physical meanings of most important DCT coefficients

Further studying all DCT AC coefficients, we can conclude that high values of $F(0,j)$ $(j = 1,...,7)$ represent the vertical dominant edges ($|F(0,1)|$ should be larger than $|F(1,0)|$); high values of $F(i,0)$ $(i = 1,...,7)$ represents the horizontal dominant edges($|F(1,0)|$ should be larger than $|F(0,1)|$); and $F(i,0)$, $F(0,i)$, $F(i,i)$ $(i = 1,...,7)$ jointly determine the diagonal dominant edges ($|F(1,0)|$ is usually equal to $|F(0,1)|$).

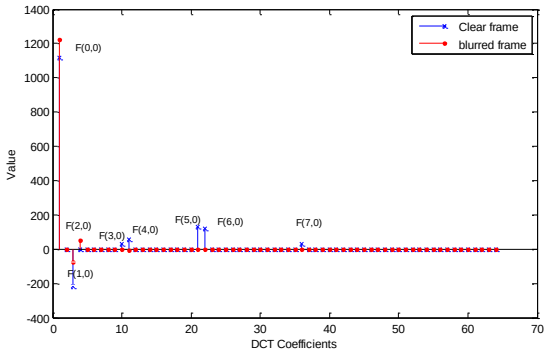
Fig. 3 physical meanings of low, middle and high frequency components

Fig. 3 demonstrates the physical meanings of low, middle and high frequency components. From it we can see that low frequencies gives overview of images while middle and high frequencies are representing the details and edges.

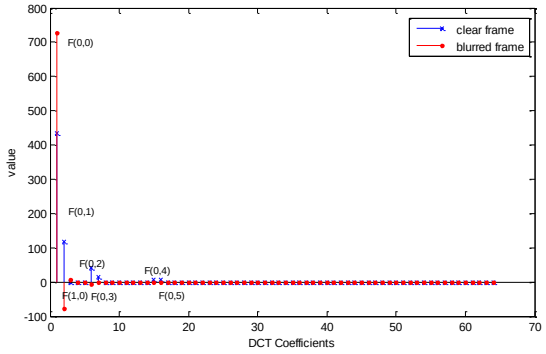
When blur occurring, the middle frequency AC coefficients, high frequency AC coefficients and corresponding edges directional AC coefficients' values will drop greatly. Fig. 4 compared the corresponding DCT coefficients between clear and blur frames with dominant direction edges. It has verified the conclusion we drew above greatly.

Fig. 5 gives a comparison of non-zero DCT coefficients numbers between clear and blurred frame without dominant direction after quantization. It is also obviously that, the non-zero numbers of middle and high AC frequencies are dropping heavily when blur occurs.

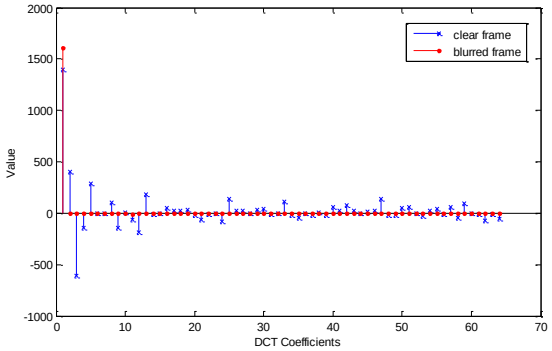
Fig. 6 compares the value of $|F_{01}/F_{10}|$ for clear and blurred frames block by block. Most blocks' ratio values are dropping deeply. A few blocks' ratio values increase a little. It is caused by fake vertical dominant edges, introduced by camera's motion movement. Greatly dropping in $|F_{01}/F_{10}|$ values indicates that vertical dominant edges are vanishing when blur occurs. It goes in line with our analysis above.



(a) Horizontal dominant edges



(b) Vertical dominant edges



(c) Diagonal dominant edges

Fig. 4 DCT coefficients' value comparison between clear and blurred frames (The coefficients are ordered by zig-zag scanning[9])

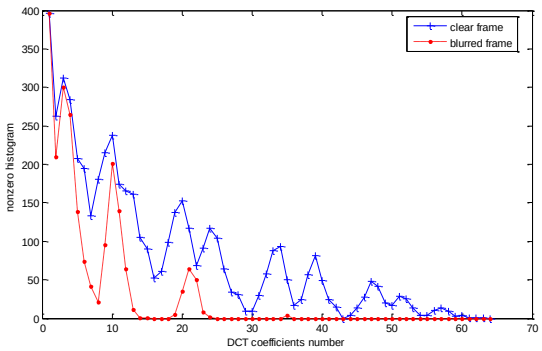


Fig. 5 DCT coefficients' nonzero histogram (the coefficients are ordered by zig-zag scanning[9])

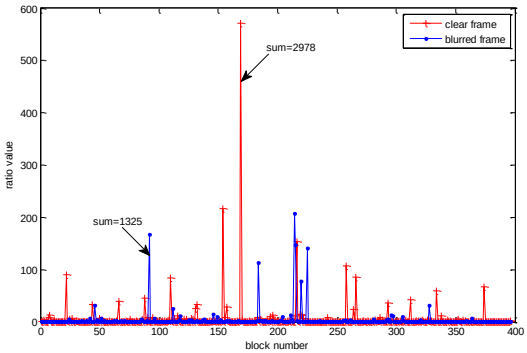


Fig. 6 ratio comparison of $|F_{01}/F_{10}|$ for every block between motion blurred and clear frame with vertical dominant edges

Since continued video frames are shot by one camera in the same condition. We can boldly assume that the total powers of continued frames within one shot are similar which shown in (6). Powers of low, middle and high frequencies will decrease when corresponding AC coefficients dropping. So the power of low frequencies in blurred frames shall be much higher than that of clear frames with the same contents, vice versa.

$$P_{total} = P_{low\text{-}freq} + P_{middle\text{-}freq} + P_{high\text{-}freq} \quad (6)$$

Fig. 7 displays the low frequencies and high frequencies' power ratio of a standard sport video stream. It is obviously that the ratios of clear frames are much lower while blurred frames with similar contents are much higher. However, this rule may not meet if frames are with different contents (see No. 78 and No. 79 frame for example. No.79 is more blurred but has lower power ratio.) So similar with other DCT coefficients' characters, we should guarantee that the compared frames are with similar contents.

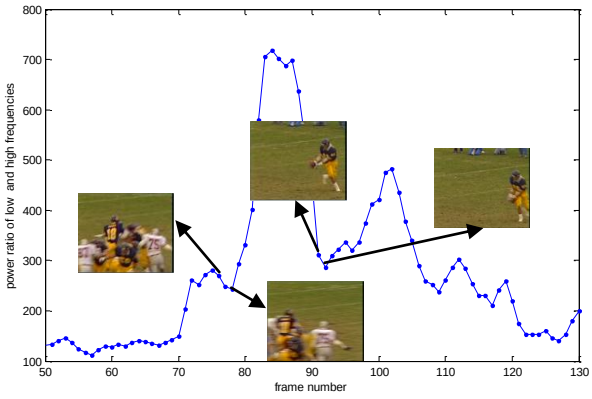


Fig. 7 low and high frequencies power ratio

3.2 Blur metric

As analysis in subsection 31., blur is the opposite of edge sharpness and details richness. DCT coefficients render these blur features via characters' changing in DCT coefficients. Our proposed blur metric method therefore looks for such changing, which is considered to judge whether an image is blurred or not.

In the implementation, we treat those DCT coefficients whose value is inferior to a threshold MinValue as zeros, to neglect small values which may result from noise generally. Typically, the threshold is set to 10. In order to be as independent of the image contents as possible, coefficients should not be considered directly since their values are too sensitive to the image they depict and noises. In our proposed scheme, we take DCT nonzero histogram to measure.

The final quality measurement is obtained via the weights matrix. Its default values are shown in (7), designed according to the general importance of coefficients.

$$\text{Weight}_{\text{default}} = \begin{bmatrix} 14 & 13 & 12 & 11 & 10 & 9 & 8 & 7 \\ 13 & 12 & 11 & 10 & 9 & 8 & 7 & 6 \\ 12 & 11 & 10 & 9 & 8 & 7 & 6 & 5 \\ 11 & 10 & 9 & 8 & 7 & 6 & 5 & 4 \\ 10 & 9 & 8 & 7 & 6 & 5 & 4 & 3 \\ 9 & 8 & 7 & 6 & 5 & 4 & 3 & 2 \\ 8 & 7 & 6 & 5 & 4 & 3 & 2 & 1 \\ 7 & 6 & 5 & 4 & 3 & 2 & 1 & 0 \end{bmatrix} \quad (7)$$

The dominant direction of block features is got by:

$$\theta = arctan\left(\frac{\sum_j F(0,j)}{\sum_i F(i,0)}\right) \quad (8)$$

where $F(i,j)$ is the DCT coefficients after de-quantizing. In this paper we restrict θ to be $0^o, 45^o, 90^o, 135^o, 180^o, 225^o, 270^o, 315^o$ by proximity principle. To improve detection accuracy, we need to estimate direction information more accurately in similar video contents. The probability of direction θ with similar contents is calculated by Bayesian fusion in (9). The weighs' actual values are adjusted by the characters of DCT coefficients via different dominant directions with max-value as 14 and min-value as 0. With the guiding idea of increasing weights of $F(i,0)\ (i = 1,...,7)$ and decreasing weights of $F(0,j)\ (j = 1,...,7)$ in horizontal dominant direction case, increasing weights of $F(0,j)\ (j = 1,...,7)$ and decreasing weights of $F(i,0)\ (i = 1,...,7)$ in vertical dominant direction case, while increasing weights of $F(i,i)\ (i = 1,...,7)$ in diagonal case, the final weights are got by (10)-(12).

$$P(\theta \mid content) = \frac{P(content \mid \theta) \cdot P(\theta)}{P(content)} \quad (9)$$

$$Weight = \mathcal{F}(P(\theta : \theta = 0^o, 180^o \mid content)) \quad (10)$$

$$Weight = \mathcal{F}(P(\theta : \theta = 90^o, 270^o \mid content)) \quad (11)$$

$$Weight = \mathcal{F}(P(\theta : \theta = 45^o, 135^o, 225^o, 315^o \mid content)) \quad (12)$$

where $\mathcal{F}(\cdot)$ is a weights adjuster.

3.3 Blur detection with temporal information

According to common sense, frames within a shot cut should have comparable edges with almost the same strength. So those unblurred frames within one shot cut shall have similar blur metrics.

In order to detect partial blurs and improve blur metric accuracy, we add motion vectors to track every 8x8 block's moving path. And classify blocks according to MV paths to design more accurate thresholds. It can weaken the influence of continued frames' differences efficiently, in order to improve the blur detection's accuracy, especially the partial blur detection.

3.4 Proposed blur detection scheme

The whole scheme is illustrated in Fig. 8. Firstly, we do shot cut detecting to segment video stream. And then use motion vectors (MVs) to classify video blocks into several kinds within one shot cut. Thirdly we analyze their DCT coefficients to calculate $P(\theta \mid content)$ in correspondent kinds of blocks located in continuous frames. After that, we adjust weight matrixes for every kind of blocks, calculate nonzero DCT histograms for different parts of frames segmented by different weight matrixes and measure the whole frame's blur together with every part's blur. Global blurred frames and partial blurred frames can be detected out more accurately by the blur metrics achieved above at final.

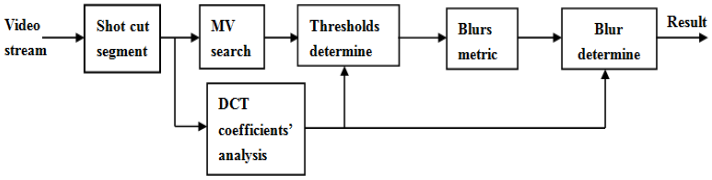


Fig. 8 Blur detection scheme

4 Experimental Results

Some experimental results are shown in this section. We tested the proposed scheme with three types of video streams. The first one is standard video testing sequences "soccer", "football", "mobile", "news", "ice", "highway", "container", "tennis", "coastguard". The second type is sports programs such as "CUBA" and "FIFA". The third one is homemade video streams. Fig. 9 displays some typical global blurred and partial blurred frames which can be extracted by our proposed

method successfully while hard to be detected by other methods described in section II. These video frames have no global dominant directions, some of them are with blurred foreground, others are with blurs in background, and the remainings are blurred in the whole frames. So they are difficult for those methods who determine global blur by Harr wavelet parameters, according to frame's DCT metric and gradient magnitude distributions.

(a) Global blur frames

(b) Partial blur frames

Fig. 9 examples of video frames global and partial blur detection results of the proposed method

A typical global blur detection result is shown in Fig. 9. The global blur is envalued by blur metric values (when united blur metric values are higher than 0.1, we give out blurred judgment), which are calculated by the proposed scheme. The testing result is quite fit with mannual decision. Fig. 11 displays the partial blur detection result of the proposed scheme. We track every block's blur metric value by motion vectors, and note it as blurred one if its value is much higher than correspondant blocks. Finally, we give out partial blur decision if the blurred blocks percentage is larger than 20%, by excluding noise effect. Talbe I compared three typical blur detection methods with the proposed one. In un-blurred and global blurred frames' detection four methods are all performing quite well. Marichal's method's accuracy is a little bit lower when the video frames with vertical or horizontal dominant directions, since it fixed the DCT coefficients' weighting matrix by emphasizing non-directional blur. Karl's method and the proposed method's accuracies are perfect, for they took temporal information into global blur's judgement. However, Marichal's method and Karl's method cannot detect partial blurs. Tong's method can pick out some partial blurs caused by motion blur according to edges' information analysis. But it is hard to extract smaller objects' moving blur and background blur. Compared with other three methods, the proposed one can give out more accurate detections in different blur cases both global and partial.

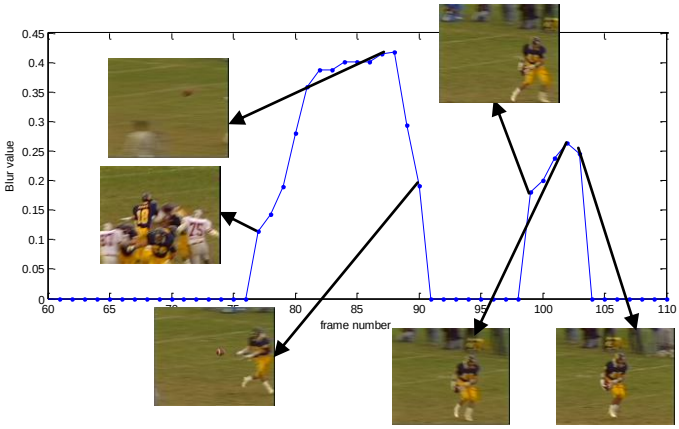


Fig. 10Global blur detection (*football_qcif.yuv*)

TABLE I BLUR DETECTION COMPARISON

Test video stream set	Accuracy (%)			
	Karl's method[8]	***Marichal's method[5]***	***Tong's method[7]***	***Proposed scheme***
Un-blurred	100	96.5	98.83	100
Global blurred	98.97	90.5	98.77	100
Partial blurred			87	95.6

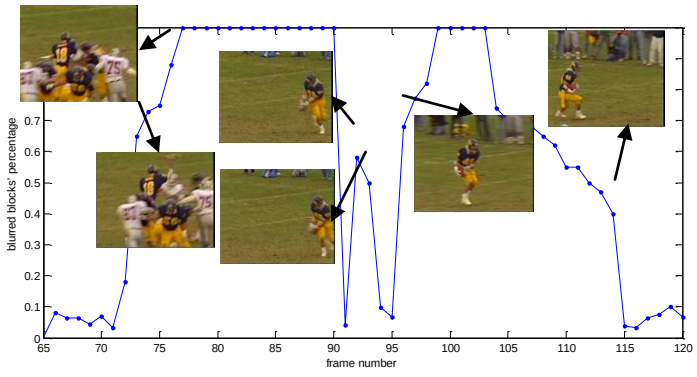


Fig. 11 Partial blur detection (*football_qcif.yuv*)

5 Conclusion

In this paper, we have proposed a new blur detection scheme for video frames in compression domain. This scheme is based on studying DCT coefficients' characters under blur affection. We have designed adaptive DCT coefficients' metric matrixes by coefficients characters and MVs' blocks classification. It can achieve more accurate global blur detection and better performance in partial blur checking. The experimental results show that the proposed scheme is effective and efficient. The computation of the proposed blur detection scheme can be further saved, since the DCT coefficients and MVs are already in compressed video streams. So the proposed detection procession can be embedded into real-time applications easily.

Furthermore, when implementing the proposed scheme into video retrieve and intelligent compression fields, they will also be benefitted a lot.

Acknowledgment

This work is supported by Chinese University Basic Funding at (ZYGX2012J024).

References

[1] R. L. Lagendijk, Basic Method for Image Restoration and identification, Academic Press, 2000.

[2] K. Sheppard, D. G. Marcellin, M. W. Marcellin, and B. R. Hunt, "Blur identification from vector quantizer encoder distortion," *IEEE Trans. On Image Proc.*, March 2001, pp. 465-470.

[3] G. Pavlovic, A. M. Tekalp, "Maximum likelihood parametric blur identification based on a continuous spatial domain model," *IEEE Trans. On Image Proc.*, Oct. 1992, pp. 496-470.

[4] W. Xu, J. Mulligan, D. Xu, and X. Chen, "Detection and classifying blurred image regions," Proc. of the *IEEE ICME*, 2013.

[5] X. Marichal, W. Y. Ma and H. J. zhang, "Blur Determination in the Compressed Domain Using DCT Information," Proc. of the *IEEE ICIP*, 1999, pp. 386-390.

[6] Y. Zhu, "Blur Detection for Surveillance Video Based on Heavy-tailed Distribution," Proc. of Microelectronics and Electronics (PrimeAsia) 2010 Asia Pacific Conf. on Postgraduate Research, Sept. 2010. Pp. 101-105.

[7] H. Tong, M. Li, H. Zhang and C. Zhang, "blur Detection for Digital Images Using Wavelet Transform," Proc. of the *IEEE ICME*, Jun. 2004, pp. 17-20.

[8] K. S. Ni, Z. Z. Sun and N. T. Bliss, "Real-time Global Motion Blur Detection," Proc. of the *IEEE ICIP*, Sept. 2012, pp. 3101-3104.

[9] Advanced Video Coding for Generic Audiovisual Services ITU-T and ISO/IEC JTC1, Rec. H.264-ISO/IEC 14496-10 AVC, 2003.