SESSION

CLOUD COMPUTING AND INFRASTRUCTURES

Chair(s)

TBA

ISBN: 1-60132-436-7, CSREA Press ©

ISBN: 1-60132-436-7, CSREA Press ©

Data Caching for Website in Cloud

Gaurav Aryal, Wei Hao
Department of Computer Science, Northern Kentucky University, Highland Heights, Kentucky, United States
aryalg1@nku.edu, haow1@nku.edu

Abstract – *This research was done to evidently prove that the response time for a cloud-hosted website is faster when the data is cached to a proxy server physically located closer than the cloud to clients. A K-means-based database partitioning algorithm was developed to group data with similar access patterns together for caching. A bookstore web application was implemented in the Amazon Cloud to perform experiment. The results of the experiment show that data caching can significantly reduce response time for a cloud-hosted website.*

Keywords: *data-caching; cloud-computing; K-mean data-partitioning; cloud-hosted website.*

1. Introduction

In simple words, Cloud Computing is the computing based on the internet. In a typical environment, programmers build some dynamic applications and deploy it in a physical computer or a server located near the business. But, with the advancement on Cloud Computing, working with dynamic applications has taken a new dimension. Cloud Computing allows people to access the same dynamic web application through the means of the Internet [16]. According to the Amazon Web Services (AWS), "Cloud Computing, by definition, refers to the on-demand delivery of IT resources and applications via the Internet with pay-as-you-go pricing." Whether we are running applications that share photos to millions of mobile users or we are supporting the critical operations of our business, the "cloud" provides rapid access to flexible and low cost IT resources. By using the Cloud Computing technology, we have a simple way to access servers, storage, databases and a broad set of application services over the Internet. In today's world, we see that more and more websites are moving towards the cloud. There are many advantage of using Cloud platform over website hosting in a server. Instead of having to invest heavily in data centers and servers before we know how we are going to use them, we can only pay when we consume computing resources, and only pay for how much we consume. Another benefit is that by using the Cloud Computing technology, we can achieve a lower variable cost than we can get on our own. Because usage from hundreds of thousands of customers are aggregated in the cloud, providers such as Amazon Web Services can achieve higher economies of scale which translates into lower pay as you go prices. In a cloud computing environment, new IT resources are only ever a click away, which means we reduce the time it takes to make those resources available to our developers from weeks to just minutes. This results in a dramatic increase in agility for the organization, since the cost and time it takes to experiment and develop is significantly lower [16].

Despite all these benefits and its flexibility, there is a major drawback for many businesses in using the cloud platform. Cloud is deployed in data-centers. It requires a lot of investment for businesses to get data centers up and running. There are a very few data centers in many locations around the world. This is because of the initial and the maintaining cost of data centers for many businesses. So, for businesses that are physically located far from the data center, the response time for the dynamic web application is usually more than those located near to the data centers. To improve the response time for the cloud hosted websites, there is a need of some cloud caching mechanism.

Web caching is a technology aimed at reducing the transmission of redundant network traffic. A lot of other related works have been done in this field of research. A user would opt for web caching to increase the bandwidth availability by curbing the transmission of redundant data [1]. Solutions including optimization of the Apache web server, introduction to caching technologies and widespread implementation of AJAX code are purposed as optimization techniques to improve performance [2]. In proxy based caching, two broad approaches exist in using proxies to cache dynamic pages, namely page-level caching and dynamic page assembly [4]. In these approaches, the dynamic contents are cached outside the site's infrastructure. In page-level caching, the proxy caches full page outputs of dynamic sites. Whereas some solutions are deployed in forward proxy mode, in distributed caching architecture located at numerous points around the Internet [5, 6]. Page-level caching can improve website performance by reducing delays associated with page generation, as well as reducing bandwidth consumption. The server side caching solutions are based on the idea of caching dynamic content within the site architecture at various levels, to accelerate dynamically generated content [3]. Various types of database caching have been suggested, including caching the results of database queries [7, 8], caching Web Views [9], caching database tables [10] and caching database tables in main memory [11].

All these techniques used work very well for web caching technologies. However, as we mentioned that more and more websites are moving towards the cloud platform, one key issue remains in hand. There is a need for caching of the cloud-hosted website.

2. System Model

In the model shown in Figure 1, the AWS Cloud Platform consists of the bookstore web application and the database. The bookstore application is a java based web application that serves as a web service for the clients to interact with the database. The web application is wrapped by a web service which acts as a standard used for exchanging data between the application and the client. In our application, the client sends a standard HTTP protocol involving the HTTP request methods like GET, POST, PUT, DELETE, and PATCH etc. to access, retrieve, modify and delete information from the database. The web service in the cloud accepts the client request, processes it with the web service, handles the request, and responds it back to the client with the HTTP response object. The web service has specific methods to handle the client requests. For example, there exists a web service method named getAllBooks() which serves as a service layer between the client and the database. When the getAllBooks() web service call is made by the client, the web service will then communicate with the database and retrieve all the books. The information thus received from the database will then be sent to the client in a JSON object format. The database hosted in the AWS cloud platform is a relational database using the MYSQL database engine.

The database consists of a table named Book. The table consists of the id of the book, the title, the name of the author, the language of the book, the online based user rating, the date that the book was published and the subject or the field of study of the book. The table consists of 5000 unique books. So, altogether the cloud platform consists of the web service along with the database with all the books available to the customer. When client requests for resources on the cloud, the response time is recorded. Now, on the other hand, a proxy web server is created which is located physically closer to the web client. This proxy server consists of the web application which is again a wrapped web service that lets a user request for resources. The database however consists of data retrieved after running the K-Means Data Clustering Algorithm on the main dataset. Since the proxy web server is just a proxy server and not the main server, only a fraction of the main dataset is stored in it. In order to retrieve only a fraction of the main data set, we choose the retrieve the dataset that are mostly accessed by a client of that location. For example, there is a client who uses the bookstore application living in Shanghai, China and the web application is hosted in Washington, United States. For this client who is accessing the application from China, he or she is most likely to search for Chinese Books in the bookstore. This does not mean that the client cannot access English books or French books, but it is highly likely that he or she just accesses the Chinese books for the most part. So, keeping this scenario in mind, the K-Mean Algorithm is run on the main dataset and clustered based on the query location of the client. Then, the fraction of the main database is stored in the proxy server located closer to the client. In this example, this fraction of data will most likely consist of many Chinese books. These books will then be stored in the proxy server. Now, when a client who is located in China makes a request to the web application, the client is served by the proxy server first. When the proxy server is able to handle the request of the client, the response time is recorded. When the proxy server is not able to handle the request, the original request of the client is redirected to the main server. The response time is recorded in this case as well. Based on this model, a various number of experimentations are performed to eventually prove that data caching can significantly reduce response time for a cloud hosted website.

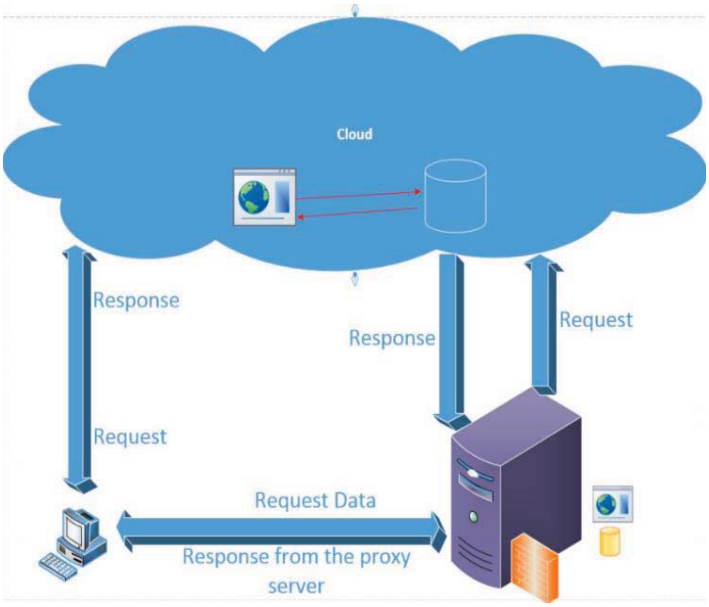


Figure 1: Model diagram

3. K-Mean Data Partitioning Algorithm

K-means is a clustering method that aims to find the positions of the clusters that minimize the square of the distance from the data points to the cluster. The problem statement of this algorithm is that we are given a set of data points and we need to group these set of data points into a cluster so that points within each cluster are similar to each other and points from different clusters are dissimilar. Usually, the data points are in a high-dimensional space, and similarity is defined using a distance measure like Euclidean, Cosine, Jaccard, Edit Distance, etc. K-Mean data partitioning algorithm is commonly known as the K-means clustering algorithm. It is also referred to as Lloyd's Algorithm. It is the most common algorithm which is as an iterative refinement technique. Clustering, in general, is the process of partitioning a group of data points into a small number of clusters. The goal is to assign a cluster to each data point. The K-mean Algorithm works as follows: First, initialize the center of the clusters. Then, attribute the closest cluster to each data point. Next, set

the position of each cluster to the mean of all data points belonging to that cluster and finally repeat steps until there is a full convergence.

The algorithm eventually converges to a point, although it is not necessarily the minimum of the sum of squares. That is because the problem is non-convex and the algorithm is based on a heuristic, converging it to a local minimum. The number of clusters should match the data. An incorrect choice of the number of clusters will invalidate the whole process.
Thus, for the K-means clustering method, if k is given, the algorithm can be executed in the following steps:

- Partition of objects into k non-empty subsets
- Identifying the cluster centroids (mean point) of the current partition
- Assigning each point to a specific cluster
- Compute the distances from each point and allot points to the cluster where the distance from the centroid is minimum
- After re-allocating the points, find the centroid of the new cluster formed

Mathematically, minimizing the within-cluster sum of squares is defined as

$$(\sum_{j=1}^{k} \sum_{i=1}^{n}) \|x_i^j - c_j\|^2$$

where, n number of objects or data items have been partitioned into k non-empty subsets S_i, i = 1,2,…,k and the term $\|x_i^j - c_j\|^2$ provides the distance between a data point and the centroid of the cluster.

K-means clustering has a lot of applications in day-to-day life. So examples include clustering customers based on their purchase histories, clustering products based on the sets of customers who purchased them, clustering documents based on similar words or shingles, clustering DNA sequence based on edit distance.

In case of our bookstore web application, The K-Mean Data Partitioning Algorithm was used to cluster the books into k = 3 groups or subsets. The clusters where chosen based on the languages of the books accessed by the clients. This information was retrieved from the apache access log. The access matrix was created of dimension n*m where 'n' were the number of data rows and 'm' were the number of data columns. To run the K means Algorithm, the Lloyd method was chosen which categorized a data point into a cluster based on the Euclidean distance from the point to the centroid of the cluster. There were a total of 5000 data points and each data point consisted of 10000 dimensions. So, instead of the two dimensional Euclidean distance, the 'n'th Euclidean Distance is calculated based on the 'n'th Euclidean distance formula. Mathematically, if 'p' and 'q' were two data points, then, in n-dimensional space, the distance is calculated as

$$d(p,q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + (p_3 - q_3)^2 + \ldots + (p_n - q_n)^2}.$$

After all the necessary data was collected, a Cluster Analysis programming language named 'R' was used. R Language is a programming language and environment for statistical computing and graphics. It is an integrated suite of software facilities for data manipulation, calculation and graphical display. Before we perform the clustering, we need the numeric matrix of data. This is the usage matrix of a tuple of database rows with the queries extracted from the apache log file.

Usage matrix is nothing more than just a multidimensional matrix. For example, in a two dimensional numerical matrix, if there are 'n' number of data rows and 'm' number of data columns, then the usage matrix can be represented by a numerical matrix of data, or an object that can be coerced to such a matrix (such as a numeric vector or a data frame with all numeric columns). For example, in a usage matrix of one thousand rows by one thousand columns, the item at row five and the column ten represents the data value for that specific row and column.

In case of our bookstore application, let us take a subset of the usage matrix of the tuple of database rows with the queries extracted from the apache long file.

Tuples/Queries	Q_1	Q_2	Q_3	Q_4	…	…	Q_n
T_1	0	0	1	1	1	0	0
T_2	1	0	0	1	1	1	0
T_3	0	0	0	0	1	1	1
T_4	0	1	0	1	0	1	0
…	1	0	1	0	1	0	1
…	0	1	0	0	0	0	0
T_n	1	1	1	0	0	0	0

Figure 2: Tuples-Queries Matrix

In figure 2, the tuples (T_1, T_2, …, T_n) represent the rows in the database whereas the queries (Q_1, Q_2, …, Q_n) represent the queries extracted from the apache access log file. Each data in the matrix represents if that query (column value) accesses the given tuple (database row). Inside the matrix, 1 represents the access hit while 0 represents the access miss of the database row using the query.

Now, to perform the k-means clustering on a data matrix, the following command was used in the R language. *kmeans (x,centers, item.max = 100000, nstart = 1, algorithm = c("Lloyd"))*
where,
'x' is the numeric matrix of data. This is the usage matrix of a tuple of database rows with the queries from the apache log file. 'centers' is the number of clusters (say k). 'iter.max' is the maximum number of iterations allowed. This was chosen to be 100000 so that algorithm does not run for a very long period of time. Algorithm is chosen to be 'Lloyd Algorithm' to calculate the cluster based on the Euclidean

distance from the data point to the chosen centroids. This Algorithm is the most common algorithm used for an iterative refinement technique. This algorithm if often presented as assigning objects to the nearest cluster by distance. This distance is the Euclidean distance between the cluster and the data point. When running the K-Means Algorithm, the centroids of the clusters are chosen at random selecting K points from the dataset. In our case, we choose 3 centroids at random from the dataset to serve as centroids. But, the centroids are adjusted in the following iterations in the algorithm.

Figure 3 represents a simple diagram of how the clustering looks after the K-Mean Data Partitioning Algorithm is run. As we can see, the data are clustered into 3 distinct and different clusters named 1, 2 and 3.

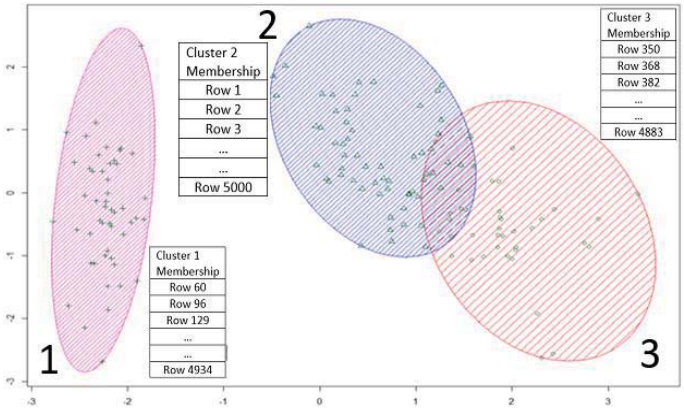


Figure 3. K-Means Data Clustering and the Cluster Membership

After the K-Mean Data Clustering Algorithm is run on the usage matrix, we retrieve the cluster membership. In the figure 3, we can see that there are 3 distinct clusters. Each data point defines its own cluster membership. For example, the first data-point belongs to Cluster 1, the second data-point belongs to Cluster 2, the third data-point belongs to Cluster 3, the forth data-point belongs to the Cluster 1, the fifth data-point belongs to cluster 3 and so on. This finally let all the data-points belong to one of the three clusters.

4. Experimental Study

Using the model described in the System Model, experimental study was done to prove that data caching can significantly reduce response time for a cloud-hosted website. In order to effectively manage a web server, it is necessary to get feedback about the activity and performance of the server as well as any problems that may be occurring. The Apache HTTP Server provides very comprehensive and flexible logging capabilities. As the client makes HTTP calls to the web service in the cloud, there are call traces created. The call traces are nothing but the apache access logs. Apache provides the access log which is the server access log records of all the requests processed by the server. Then, the entire access log was analyzed in detail. A subset of the data access log was taken into the experimentation. The subset consisted of ten thousand lines of access log data. The format for the access log is exactly the same as the Common Log Format, with the addition of two more fields. Each of the additional fields uses the percent-directive %{header}i, where header can be any HTTP request header. The access log under this format will look like:

127.0.0.1 - frank [10/Oct/2000:13:55:36 -0700] "GET /apache_pb.gif HTTP/1.0" 200 2326 "http://www.example.com/start.html" "Mozilla/4.08 [en] (Win98; I ;Nav)"

This server IP address, the GET request were then taken into account for the experimentation process. This was used to create the usage matrix. The usage matrix was nothing but an information matrix of the access log to the database row item as described in Section 2. The access hit or miss is determined from the access log.

Figure 2 represents a subset of the main usage matrix. In figure 2, the rows represent the database rows. As there are 5000 rows in the database hosted on the Amazon RDS instance, there are 5000 number of rows represented by T_1, T_2, T_3,…, T_{5000}. The columns represent the access hit or miss of each apache access log information. For instance, let us take the row number 10 from the database. As we analyze the tenth row with the column number 2, we see a value of '1'. We can interpret that as row number 10 is one of the data items in the database that is accessed/hit by the query in column number 2. Similarly, if we take row number 1 and column number 1, we see a value of '0'. We can interpret that as first database row is not accessed (or missed) by the query in column number 2 and so on.

This usage matrix is then fed into the K-Mean Data Clustering Algorithm. The K-Mean Algorithm then computes the distances from each point and allot points to the cluster where the distance from the centroid is minimum. For convenience, we set the number of clusters (k) to 3. After the K-Mean Algorithm is run, we have data nicely grouped into 3 clusters. We then cache one of the clusters to a proxy server physically located closer than the web server to clients. Then, we conduct the final set of experimentation. We created a pool of 25, 50, 75, 100, 125, 150, 175 and 200 concurrent users to test this scenario. On one hand, we record the response time for the clients who access the cloud application and on the other hand, we record the response time for the clients who access the application from the cached server.

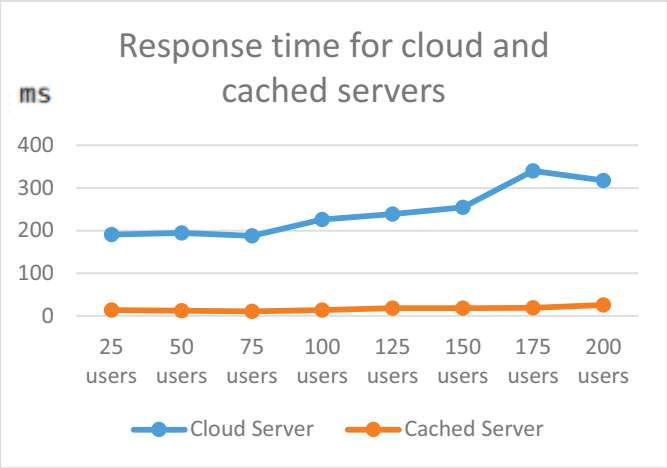


Figure 4. Response time for cloud and cached servers

The bookstore application used in the experimental study was created using the Spring Framework. The bookstore application was nothing but a web service with an HTML view engine for the client. When a client makes a request to the web application, an underlying service in the application is called to do a specific task. This service is nothing but a web service call. When the web application gets the request from the client, a specific web service method is called and the response is returned back to the client. This response is returned in the JSON format. Since this is not an ideal or a pretty form of viewing the resources returned from the server back to the user, the JSON objects are parsed and presented in a nicely formatted table or output to the user [12]. HTML view engine helps present the response of the web service is a user friendly view back to the client. The Spring Framework used to create the bookstore application uses the Spring Boot technique. Spring Boot makes it easy to create stand-alone, production-grade Spring based Applications that you can just "run". Most Spring Boot applications need very little Spring configuration. Spring Boot applications feature of stand-alone Spring applications; embedded tomcat (no need to deploy WAR files); provides opinionated 'starter' Gradle file to simplify the Gradle projects; provides production-ready features such as health checks and externalized configuration and automatically configures Spring whenever it is possible. This dynamic web application as well as the database is hosted in the Amazon Cloud platform. The web application is hosted in the AWS Elastic Beanstalk platform. AWS Elastic Beanstalk is an easy-to-use service for deploying and scaling web applications and services developed with Java, .NET, PHP, Node.js, Python, Go, Ruby, and Docker on familiar servers such as Apache, Nginx, Passenger, and IIS [14]. The bookstore web application and services was developed in Java with the Apache server. The web server environment was created as a single instance, auto-scaling Apache Tomcat 8 environment. The WAR file generated from the build of the web application was used as the source for the web server. The processing power of the web server in the environment used was of instance type t1.micro. With all these configurations, the environment was launched. Similarly, the Database was hosted in the Amazon Relational Database Service (Amazon RDS). Amazon RDS is an Amazon Cloud Platform application which makes it easy to set up, operate, and scale a relational database in the Cloud. MySQL was used as the database engine. It provides cost-efficient and resizable capacity while managing time-consuming database administration tasks, freeing us up to focus on our applications and business needs. The database engine hosted is easy to administer and is highly scalable. It is also highly available and durable, fast, secure and inexpensive [13]. The database engine used was My-SQL. The instance class was used as db.t2.micro with the storage capacity of 5 GB. The default endpoint is created which connects to the dynamic web application with the means of My-SQL driver.

The java based web application, along with one of the clusters obtained from the K-Means Data Partitioning Algorithm is cached to a proxy server physically located closer than the web server to clients. Only one of the clusters is picked and cached to the proxy server. The web service also needs to be cached meaning that the web service is wrapped in the proxy server so serve the same functionality as the main server. The web service hosted in the cloud is modified in the proxy server so that when a request comes in from the client, the web service always looks for the data in the proxy server first. So, for instance, a client sends a request to the bookstore web application. This request is first sent to the proxy web server that is physically located closer to the client. Since this data is a cached version of the main web service, the client's request may or may not be handled by the proxy server. If the proxy server is able to handle the request, then the response is returned back to the client. But, if the proxy server cannot handle the request, then the original request from the client is sent to the main server so that the request is handled appropriately. Figure 4 represents the graph of plotting the response time for 25, 50, 75, 100, 125, 150, 175, and 200 concurrent users for both the main cloud server and the cached server. For the cached server, the response time for clients are significantly less than that of a server hosted in the cloud. The response time for the cloud server is 191 milliseconds for 25 concurrent users, 195 milliseconds for 50 concurrent users, 188 milliseconds for 75 concurrent users, 226 milliseconds for 100 concurrent users, 239 milliseconds for 125 concurrent users, 255 milliseconds for 150 concurrent users, 340 milliseconds for 175 concurrent users and 318 milliseconds for 200 concurrent users. Similarly, for the cached server which is located physically closer to the client, the response time is 14 milliseconds for concurrent 25 users, 12 milliseconds for 50 concurrent users, 11 milliseconds for 75 concurrent users, 14 milliseconds for 100 concurrent users, 14 milliseconds for concurrent 125 users, 18 milliseconds for 150 concurrent users, 19 milliseconds for 175 concurrent users and 26 milliseconds for 200 concurrent users.

After the experimentation, we can see that the response time for the cached server is significantly lower than the main server. Thus, the experiment's results show that data caching can significantly reduce response time for a cloud-hosted website.

5. Conclusion and Future Work

Thus, with the means of various applications and different software platforms, we were successfully able to carry out the research of the data caching for a cloud hosted website. After the experimentation was carried out, we were evidently able to conclude that the data caching for a cloud-hosted website can significantly reduce the response time for the clients.

In this research, all the HTTP requests by the clients are GET requests. In other words, both the cached server and the main server only handle the reads from the database. As part of an on-going research, the future work remains. The future work will consist of experimentation of the research along with the write operations. After a client buys a book, the number of books in the bookstore application should reduce from the total number of available books to be purchased. While this seems like a feasible task, we need to be careful on how we handle the synchronization issues. If a client buys a book from the main server, then all of the cached servers with the total number of books have to be updated. So, to make it practical, all the cached servers and the main cloud server have to be in sync. Determining and enforcing per-application resource quotas in the resulting cache hierarchy, on the fly, poses a complex resource allocation problem spanning the database server and the storage server tiers. Modern enterprise systems consist of multiple software layers including web/application server front-ends, database servers running on top of the operating system, and storage servers at the lowest level. Thus, there is a high need for better management of shared multi-tier caches. Thus, much of the work on this part is needed.

References:

[1] S.V. Nagaraj, Web Caching and Its Applications, 2004. ISBN: 1-4020-8049-2

[2] Parker, Anne. BMC Bioinformatics Volume. 2000-02-02 – present. Using caching and optimization techniques to improve performance of Ensembl website.

[3] K. Candan, W.-S. Li, Q. Luo, W.-P. Hsiung and D. Agrawal. Enabling dynamic content caching for database-driven web sites. In Proceedings of ACM SIGMOD International Conference on Management of Data (SIGMOD), May 2001.

[4] Suresha, Dr. of Philosophy in the Faculty of Engineering. June 2007. Caching Techniques for Dynamic Web Servers.

[5] S. Gadde, M. Rabinovich and J. Chase. Reduce, Reuse, Recycle: An Approach to Building Large Internet Caches. In Proceedings of Workshop on Hot Topics in Operating Systems, May 1997.

[6] M. Rabinovich and A. Aggarwal. Radar: A scalable architecture for a global web hosting service. In Proceedings of 8th International World Wide Web Conference (WWW), May 1999.

[7] S. Choi, S. Huh, S. M. Kim, J. Song and Y. Lee. A Scalable Update Management Mechanism for Query Result Caching Systems at Database-driven Web sites. In Proceedings of 8th Asia Pacific Web Conference (APWEB), January 2006

[8] Q. Luo and J. Naughton. Form-based proxy caching for database-backed web sites. In Proceedings of 27th International Conference on Very Large Data Bases (VLDB), September 2001

[9] A. Labrinidis and N. Roussopoulos. Balancing Performance and Data Freshness in Web Database Servers. In Proceedings of 29th International Conference on Very Large Data Bases (VLDB), September 2003

[10] M. Altinel, C. Bornhoevd, S. Krishnamurthy, C. Mohan, H. Pirahesh and B. Rein-wald. Cache Tables: Paving the Way for an Adaptive Database Cache. In Proceedings of 29th International Conference on Very Large Data Bases (VLDB), Berlin, September 2003

[11] Oracle Corp. Oracle 9ias database cache. http://www.oracle.com/ip/deploy/ias/db_cache_fov.html.

[12] Oracle, "What are web Services? – The Java EE 6 Tutorial", http://docs.oracle.com/javaee/6/tutorial/doc/gijvh.html

[13] Amazon Web Services, Inc., Amazon Relational Database Service (RDS) Documentation, http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/rds-ug.pdf

[14] Amazon Web Services, Inc., AWS Elastic Beanstalk Developer Guide, API Version 2010-12-01, https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/awseb-dg.pdf

[15] JSON Tutorial, http://www.w3schools.com/json

[16] Amazon Web Services, Inc., What is Cloud Computing? https://aws.amazon.com/what-is-cloud-computing

Improving Web Site Navigational Design and Performance from Web Log Data

Esther Amo-Nyarko, Wei Hao
Department of Computer Science, Northern Kentucky University, Highland Heights, Kentucky, USA

Abstract – *As data and usage of World Wide Web (WWW) grows, user demand for better user experience also increases. Web users desire to find important links and information quickly on websites. A poorly designed site thus can lead to low productivity and revenue. Designing website with better user experience is of critical importance. A faster response time and high server throughput play important roles in providing better user experience. This paper proposes a method based on association rule mining and proxy server caching to improve user response time and server throughput. The web log data a website is collected and sorted by on countries by utilizing GeoIP to deduce users' geolocations. Association rule mining is then used to analyze the geolocation-based log data and discover important user navigational patterns and redesign the site. The discovered rules are cached on proxy server closer to the users' location. An experimental study is conducted on this approach versus the regular approach where user activity is not considered in the system design. Results from the study show a high response time and throughput using the above described approach.*

Keywords: Web Log Mining, Web Caching, Association Rule, Cloud Computing, GeoIP

1 Introduction

"Websites must provide a "rapid retailing" experience if they want to expand their consumer sales, according to research" [1]. A poll conducted by Riverbed Technology, showed that 67% of online shoppers will leave a website due to slow performance. They discovered, almost 70% of online shoppers will return to buy and items from an online store if they had previous positive experience even if they could get it cheaper elsewhere [9]. Another study reveals that a response time of 0.1 and 1 second create a pleasant user experience however users lose interest for a 10 seconds response [10].

These results clearly depict the significance of the performance rate of any web site. Various researches have been conducted in this area to provide better user experience using methods such as prefetching and web catching, web mining [11, 12 and 13] and all of which have been adopted to provide recommender based systems [7] and as well improved web site navigational patterns [3]. Web mining is the usage of information processing techniques to the WWW in order to discover useful patterns from data collected from the web [2]. Since cache storage is limited, web mining provides us with the most relevant objects to cache. Data used in these analyses are collected from user activity from log files on application servers. The Prefetching technique predicts a future web resources request by a client and caches the objects before they are explicitly requested by the user. Web caching on technique on the other hand considers highly requested objects and store them close to the user machine on the client machine or proxy server [14]. Prefetching becomes disadvantageous when the objects initially loaded into the proxy server cache is not requested by the user which implies huge server traffic and load. Also to detect which object to put or remove (when the cache is full) from the cache, the following factors are used; size of the object, number of times the object is requested and the time the object was added to the cache. The relationship between the objects are not considered which flaws the criteria used in discovering the objects to cache [14].

This study combines web mining and web caching techniques (Proxy Server Caching). Proxy Server serves as an intermediary between web server and the user request. When a user sends a request, the proxy server checks its cache to see if the requested resource is available. If the resource is available (cache hit), it serves the client with a response otherwise (in the case of cache miss), it forwards the request to the original server. For the user, this gives a faster response time. On the server side, this reduces network bandwidth usage and traffic on the original server hence reducing the original server load [14]. The web mining approach discovers the user navigational and access pattern from log files using the Association rule Mining Technique. Data from obtained from the server logs is cleaned to remove irrelevant information like media files, error pages etc. The fields important to the web mining process are selected. Association rule is then used to discover the relevant user patterns. The objects obtained from the rules are then cached on the proxy server for performance enhancement.

2 Related Work

The plain web caching approach uses cache replacement policy. When a user request a resource which is not found on the proxy server, the request is forwarded to the original web server which returns the resource to the proxy server. The proxy server caches the object and serves the object to the web user. The decision on which object to replace or which object to keep as the cache gets full depends on the size of the

object, how recently it was accessed, and the number of request to that object and the cost of retrieving the object from the original server [14]. Some studies have used the Least-Recently-Used (LRU) algorithm in their cache management. In this approach the object that is least recently accessed is removed until there is sufficient space for new objects. This performs poorly in web caching as the size of object to be cached is not taken into consideration. Others have used the Least-Frequently-Used approach which replaces the objects that least referenced in the cache. However objects which have been highly referenced but no longer referenced are still kept in cache location [14]. Study [11] uses the LRU strategy with extra criteria. They combine Recent Access and Frequency strategies are based on Podlipngi and Bsz Bszremnyi classification. The algorithm performs better than the basic LRU only on a low cache storage environment.
Other studies have used the cost function based prefetching approach i.e. web objects are pre-fetched into the cache depending on their popularity and life time. Another approach is Markov Model Based where the objects are pre-fetched based on combining the user's current access with the user's web access sequence history. In [13], prefetching scheme is used to discover user access pattern and a cluster created. When a user requests an object, all other objects related to the cluster are fetched from the original server to the proxy cache. This is done in expectation that next set of objects requested by the user will be those that have been loaded into the cache. The aim is to boast server performance in hit ration and byte hit ratio as opposed to the plain web caching approach which has low hit ratio.

3 System Model

Cloud hosting based on cloud computing technology has become the preferred means of web hosting for web applications. Cloud is made up of a group of computing devices working together to provide a service. Cloud computing is where these cloud resources are used to provide service over the internet. Cloud hosting uses the server resources of the cloud to host websites as compared to other hosting options where one machine is used. With cloud hosting, security and high reliability are guaranteed. Cloud hosting is also cost effective and scalable among others and removes the burden of purchasing servers and other resources to run your own data center [16]. Cloud hosting providers include Amazon, Bluehost, InMotion, Westhost, SiteGround etc. Our system is hosted on Amazon cloud. The Amazon Web Service (AWS) provides cloud hosting for web applications. The Amazon Elastic Compute Cloud (EC2) provides the capacity to run application on the Amazon cloud. It gives the ability to manage storage capacity, create virtual servers and configure security [15]. However, Amazon data center is not available in every country like Ghana, hence access to resource from those countries take a longer time. A proxy server helps reduce this latency. The proxy server serves as intermediary between the client and the original server. Some resources are cached on the proxy server and it would only forward request to the original server when a particular requested resource is not found in its cache. The most important question here is what do we cache on the proxy server? Our approach uses Association Rule Mining. The research paper details this method.

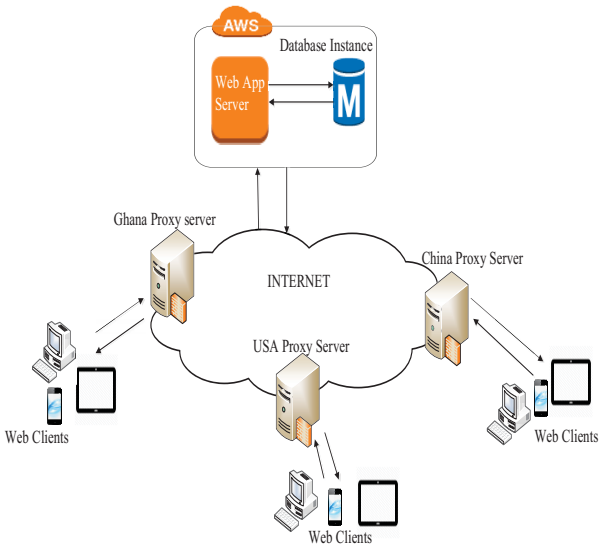


Fig 1. Architectural Diagram of our System Model

4 Our Approach

Association Rules are the relationships that are discovered from the probability of items continuously occurring from a collection using the association data mining function [4]. For example, let's denote "news.jsp" by N and "donations.jsp" by "D", the rule {N} => {D} implies if a user views the news page he is likely to visit the donations page. With large amount of data that is extracted from server logs, the following criteria; support and confidence are used to extract the most relevant rules.

Support: The support defines frequency of occurrence of an item in a collection. *Let's denote, Transactions by T.* the support of the rule {N} => {D} is evaluated as;

$$P(N \cap D) = \frac{\text{No of T in which both N and D occur}}{\text{Total number of user T}} \quad (1)$$

Support is symmetric in that the support of {N} => {D}

= {D} => {N} [5].

Confidence: Confidence shows how reliable the rule is. It defines the percentage of occurrence of a transaction containing an item N and also D [5]. The confidence of the rule {N} => {D} is estimated as:

$$P(D|N) = \frac{P(N \cap D)}{P(N)} = \frac{\text{No of T in which both N and D occur}}{\text{Number of T in which N occur}} \quad (2)$$

Confidence however is not symmetric, i.e. P (D|N) ≠ P (N|D) [5]. Another useful criteria is Lift. For the derived rule,

{N} => {D} to be significant, the conditional Probability of P (D|N) must be greater than the probability, P (N). This is known as lift [4] and estimated as [5];

$$\frac{P(D \mid N)}{P(D)} = \frac{P(N \cap D)}{P(D)*P(N)} = \frac{\text{No of T in which both N and D occurs}}{\text{No of T in which D occurs} * \text{No of T in which N occurs}} \quad (3)$$

For the purpose of this research, a support of 5% and confidence of 80% is used.

4.1 Processing Log files

Server logs contain information about the request -client IP address, time of request, the requested page, HTTP response code, and bytes of data sent, user agent, referrer etc. The collected data is cleaned and irrelevant data removed. A user session is identified using IP and type of browser used. The user's geolocation is also identified from the IP using GeoIP and the data obtained from all these processes grouped into logs based on the user location. GeoIP is a method of finding the geographical location of a terminal using the terminal'S IP address [6]. The purpose of the GeoIP in this study is to narrow the rules to be country specific. We consider log files from Ghana, USA, China users. After the logs are cleaned up the following data is retrieved – IP Address, Requested Page, Referrer, Duration on the Page and Location. Sample Cleaned data is shown in Figure 3. The data is then analyzed using *arules* and *arulesViz* packages in R software environment. Arules packages is used to create the mapping rules and the arulesViz used for plotting and creating graphical representation of the rules

4.2 Deriving the Rule in R

To derive the rules, first make sure the packages arules and arulesViz are installed in R and enable them. Also set the working directory to the appropriate directory so that the imported file can be identified. The R pseudocode is provided below

```
read.table(file location, separator, header) #Load data in R and assign to a variable
discretize (column name, number of categories, method) #categorize any continuous variable
as(log data, "transactions")#convert data to transactions
apriori(transaction, parameter =list(support=value, confidence=value)#generate the rules
inspect (rules)#view generated rule
plot(rule name, method) #plot a graph of the rules
```

```
192.168.198.92 - - [22/Dec/2015:23:08:37 -0400] "GET / HTTP/1.1" 200 6394 www.ccsafetynet.org "-" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1...)" "-"
192.168.198.92 - - [22/Dec/2015:23:08:38 -0400] "GET /images/logo.gif HTTP/1.1" 200 807 www.ccsafetynet.org "http://www.ccsafetynet.org/userProfiles.jsp" "Mozilla/4.0 (compatible; MSIE 6...)" "-"
192.168.72.177 - - [22/Dec/2015:23:32:14 -0400] "GET /pages/news.jsp HTTP/1.1" 200 3500 www.ccsafetynet.org "http://www.ccsafetynet.org/contactUs.jsp" "Mozilla/4.0 (compatible; MSIE ...)" "-"
```

Fig 2. Sample apache server web application log file

IP_ADDRESS	REQUESTED_PAGE	REFERRER	DURATION	COUNTRY
41.57.792.20	/pages/aboutUs.jsp	/index.jsp	392	GHANA
41.57.792.20	/pages/successStory.jsp	/pages/news.jsp	425	GHANA
41.57.792.20	/pages/successStory.jsp	/pages/donations.jsp	407	GHANA
41.57.792.20	/pages/contactUs.jsp	/pages/aboutUs.jsp	598	GHANA
41.57.792.20	/pages/aboutUs.jsp	/index.jsp	364	GHANA

Fig 3. Sample cleaned data to be used in association rule mining

4.3 Results

The rules derived from Ghana logs are shown below. A graph plot of the rules is shown in Figure 6.

```
  lhs                                      rhs                                          support     confidence     lift
{REQUESTED_PAGE=/pages/contactUs.jsp}  =>  {REFERRER=/pages/aboutUs.jsp}                0.10944911          1     9.136667
{REQUESTED_PAGE=/pages/aboutUs.jsp}    =>  {REFERRER=/index.jsp}                       0.10944911          1     7.028205
{REFERRER=/pages/donations.jsp}        =>  {REQUESTED_PAGE=/pages/successStory.jsp}    0.06092667          1     3.555123
{REFERRER=/pages/news.jsp}             =>  {REQUESTED_PAGE=/pages/successStory.jsp}    0.06895294          1     3.555123
```

Fig 4. Association rules discovered from Ghana log files. *Note: lhs – left hand side, rhs – right hand side*

A graph plot of the Ghana rules in shown in Fig.6. The difference in sizes in Figure 6 indicates the varying support value for all the rules from biggest to smallest ranging from (0.609 – 0.109) while the difference in color indicates the differences in the lift ranging from (3.555 – 9.136). The rules are sorted by the lift in order of highest level of significance. The arrows show the pages that have association.

Interpretation of Rules – Taking the rules discovered from Ghana (Table shown in Figure 4 and graphical representation shown in Figure 6), from the derived rules for Ghana, for every user who accesses the about us page, the user is likely to visit contact us page. Likewise after the user lands on the index page he is most likely to visit the about us page. Another important pattern seen, is for every user that visits the donation page or news page, that user is most likely to visit the success story page. The rules recommend creating a link to the corresponding pages if links between them do not already exist. Similar interpretation applies for rules discovered for China and USA as shown in figures 4 and 7 respectively.

The rules derived from China log files are shown below in Fig 5.

lhs		rhs	support	confidence	lift
{REFERRER=/admin/serviceEdit.jsp}	=>	{REQUESTED_PAGE=/admin/services.jsp}	0.09881423	1.0000000	9.889251
{REFERRER=/admin/agencyProfiles.jsp}	=>	{REQUESTED_PAGE=/admin/memberProfiles.jsp}	0.09914361	0.9836601	9.664700
{REQUESTED_PAGE=/admin/login.jsp}	=>	{REFERRER=/index.jsp}	0.09881423	1.0000000	7.784615
{REQUESTED_PAGE=/admin/agencyProfiles.jsp}	=>	{REFERRER=/admin/login.jsp}	0.09881423	0.9803922	4.960784
{REQUESTED_PAGE=/admin/news.jsp}	=>	{REFERRER=/admin/login.jsp}	0.09881423	0.9740260	4.928571

Fig 5. Association mining rules discovered from China log files

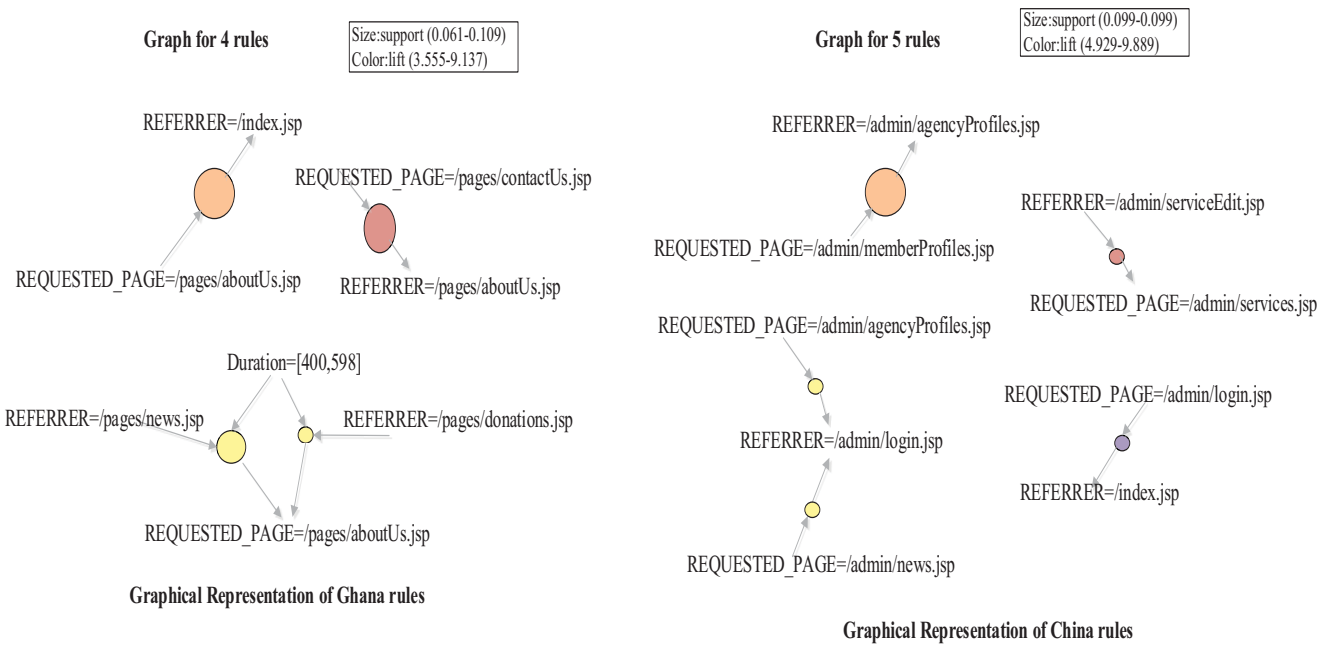


Fig 6. Graphical representation of association rules derived from Ghana and China log files respectively

The rules derived from USA log files are also shown below

lhs		rhs	support	confidence	lift
{REQUESTED_PAGE=/admin/reviewDonations.jsp}	=>	{REFERRER=/admin/reviewStory.jsp}	0.110011	1.0000000	8.712460
{REQUESTED_PAGE=/admin/login.jsp}	=>	{REFERRER=/index.jsp}	0.110011	1.0000000	6.992308
REQUESTED_PAGE=/admin/news.jsp}	=>	{REFERRER=/admin/login.jsp}	0.110011	0.9646302	4.384244
{REQUESTED_PAGE=/admin/reviewStory.jsp}	=>	{REFERRER=/admin/login.jsp}	0.110011	0.9554140	4.342357

Fig 7. Association rules discovered from USA log files

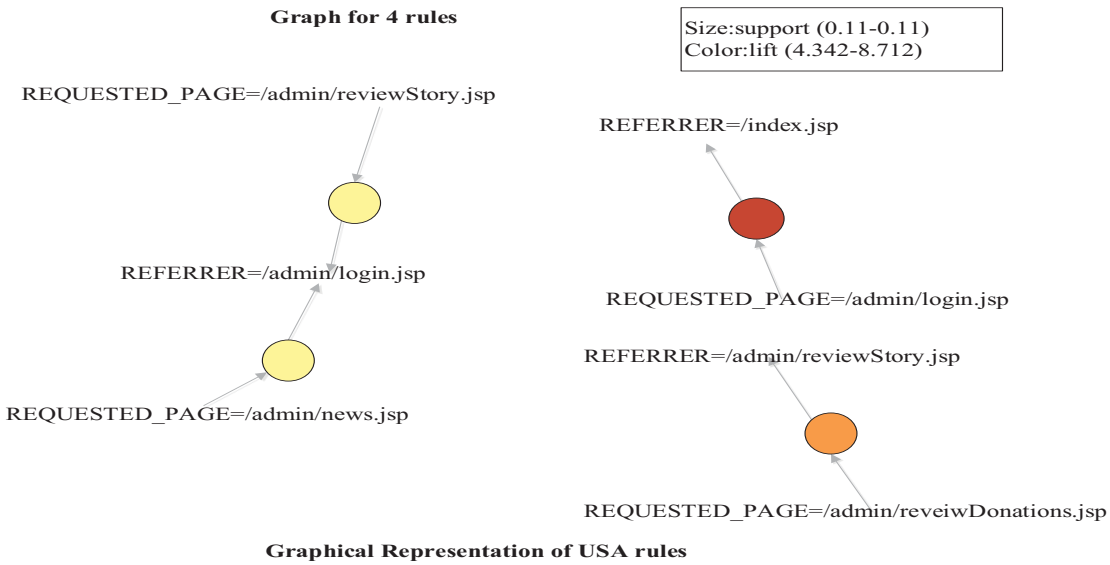


Fig 8. Graphical Representation of rules derived from USA Log files

5 Experimental Study

Based on our association rules and patterns derived, the website is redesigned based on country specific rules with the aim of improving performance. An experimental study is performed to prove the relevance of the research. The experimental study focuses on the rules derived from Ghana only and caches the pattern on a proxy server. A performance test based on two scenario is then conducted:
Case 1 - For every user request, the request is initially sent to the proxy server. If there is a cache miss, request is then forwarded to the original web server.
Case 2 - All requests are sent directly to the original web server in the Amazon Cloud.
In both scenarios, we lookout for Average Response Time and Throughput.
Response Time – This is a measure of how long it takes for response to be received from an application for a sent request [7]. The duration is measured in milliseconds (ms). Low response time value implies high performance and vice versa.
Throughput – This is a measure of the number of transactions processed over time. It depicts the capacity of a website [7]. We measure throughput per second (per sec). A higher throughput value implies the higher performance i.e. the system is able to process a lot of request per second.

5.1 Tools Used

The test is performed using JMeter 2.13 on Ubuntu 14.04 platform. The application runs on Apache Tomcat 7.0.41, AWS (Amazon Web Service) CLI, an AWS EC2 instance with the instance type of t1.micro.

5.2 Test Case

For each case, we consider 1, 10, 50, 100 and 200 users. We assume all users access the following pattern:
index.jsp -> aboutUs.jsp -> contactUs.jsp -> login.jsp -> member.jsp -> alerts.jsp ->logout-> news.jsp ->successStory.jsp.
Per the rules for Ghana, the following patterns will be cached on the proxy server;
index.jsp->aboutUs.jsp->contactUs.jsp, news.jsp-> successStory.jsp and the following pattern also accessed directly from the webserver*; login.jsp -> member.jsp -> alerts.jsp ->logout*. Our test plan is created for every case using apache JMeter. The average response time (ms) and average throughput (per sec) are recorded.

5.3 Findings on Response Time (ms)

A tabular and graphical Representation of result is shown in Table 1 and Figure 9 respectively. A plot of average response time indicates a lower response time value in Case 1 as compared to Case 2 which is as expected.

5.4 Finding for Throughput (request per sec)

A tabular and graphical Representation of result in Table 2 and Figure 10 below. A plot of average throughput indicates higher performance in Case 1 as compared to Case 2 which is also as expected.

Table 1. Tabular representation of results for response Time

Number of Users	CASE 1	CASE 2
1	81	71
10	431	522
50	725	1296
100	825	1249
200	1873	2581

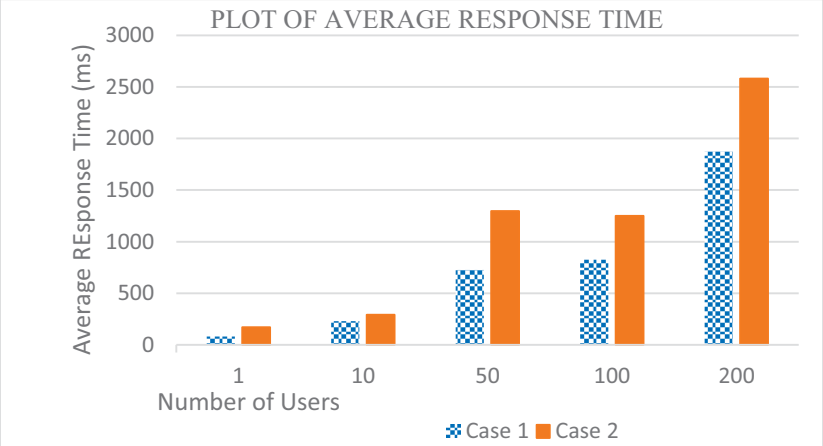


Fig 9. Graphical Representation of Results on Average Response Time

Table 2. Tabular representation of results for response time

Number of Users	CASE 1	CASE 2
1	11.9	5.8
10	14.5	8.6
50	34.3	17.6
100	39.8	25.8
200	52.7	41.4

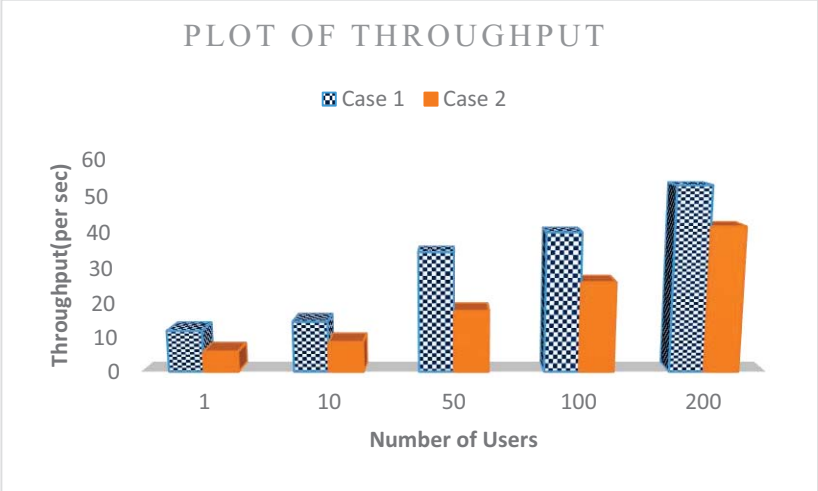


Fig 10. Graphical Representation of Results on Throughput

6 Summary and Future Work

The study showed web site navigational pattern discovered from association rule mining and cached on proxy improves performance. This method is therefore recommended to be used in improving efficiency of web application

This paper only looked at caching user navigational patterns on the proxy server. The database however still resides on the cloud server therefore for any call to the proxy server that requires database query, the proxy server communicates with the database server on the AWS. To further improve performance, future works would look at analyzing database log file for the web application. This would be done to discover the relationship between frequently accessed data in the database and then group them into one table for easy and faster access. With that the system architecture would be modified and the items that are frequently accessed together cached also on the proxy server. This would be done using K-mean clustering technique. K-mean data mining algorithm categorizes data into cluster in that items belong to clusters with the nearest mean [8].

7 References

1. 2012. Online shoppers demand faster experience. ComputerWorldUK The voice of Business Technology. http://www.computerworlduk.com/news/infrastructure/online-shoppers-demand-faster-experience-3350109/
2. J. LU, D. RUAN. 2007. E-Service Intelligence: Methodologies, Technologies and Applications, G. ZHANG, Ed. Springer-Verlag, Berlin, Germany, pp. 536-539.
3. M. KIRUTHIKA, R. JADHAV, J.RASHMI, D. DIXIT, A. NEHETE, T. KHODKAR. 2011. Pattern Discovery Using Association Rules. International Journal of Advanced Computer Science and Applications, 2, 6, 70.
4. Data Mining Concepts: Association https://docs.oracle.com/cd/B28359_01/datamine.111/b28129/market_basket.htm
5. Y.S. KIM, B. YUM. 2011. Recommender system based on click stream data using association rule mining. Expert Systems with Applications, 38, 10, 13321.
6. What is GeoIP? Available at https://docs.nexcess.net/article/what-is-geoip.html
7. J.COLANTONIO. 2011. Performance Testing Basics. http://www.joecolantonio.com/2011/08/17/performance-testing-basic%E2%80%99s-%E2%80%93-what-is-response-time/ and http://www.joecolantonio.com/2011/07/05/performance-testing-what-is-throughput/
8. K-mean Clustering. http://www.saedsayad.com/clustering_kmeans.htm
9. 2013. Riverbed - New Harris Poll Reveals 67 Percent Of Online Shopper Will Leave a Website Due to Slow Performance. http://www.riverbed.com/press-releases/New-Harris-Poll-Reveals-67-Percent-of-Online-Shoppers-Will-Leave-a-Website-Due-to-Slow-Performance.html
10. NIELSON, J. 2010. Website Response Times. Web Usability. https://www.nngroup.com/articles/website-response-times/
11. JARUKASEMRATANA, S. AND MURATA, T. 2013. Web Caching Replacement Algorithm Based on Web Usage Data. New Generation Computing, 31,311-329
12. SHENOY, A. 2011. Improving the Performance of a Proxy Server using Web log mining. San Jose State University. http://scholarworks.sjsu.edu/cgi/viewcontent.cgi?article=1170&context=etd_projects
13. SATHIYAMOORTHI, V. AND RAMYA, P. Enhancing Proxy Based Web Caching System Using Clustering Based Pre-Fetching With Machine Learning Technique. 2014. International Journal of Research in Engineering and Technology, 3, 7, 463-469.
14. ALI. W., SHAMSUDDIN, M.S. AND ISMAIL. A.S. 2011. A Survey of Web Caching and Prefetching. International Journal of Advances in Soft Computing and its Applications, 3, 1.
15. Amazon Web Services. http://docs.aws.amazon.com/AWSEC2/latest/WindowsGuide/concepts.html
16. Interoute from the ground to the cloud. http://www.interoute.com/cloud-article/what-cloud-computing

Cloud computing and data protections: state of the art and possible evolutions

Polzonetti Alberto [1], **Sagratella Matteo**[2], **Tapanelli Pietro**[1]
[1]School of Science, University of Camerino Italy
[2]School of Law , University of Camerino Italy

***Abstract* -** Cloud computing services are ubiquitous : for small and large companies the phenomenon of cloud, computing is nowadays a standard business practice. This paper would compile an analysis about the contractual issues related to the different services provided by all IT services provider. There are faced not only legal topics strictly connected to the personal data protection because, as the thesis will show, there are several legal acts provided by European Union, Italy (and from all EU's member states) and data protection authorities (both nation and international) that discipline the entire data protection topic in the cloud computing era. This paper analyze the state of the art and provide a possible solution in order to find the legal discipline of cloud computing contracts, especially using the theory of "connected contracts".

Keywords: Data Protection, Cloud Computing Contract, Cloud Computing Legislation

1 Introduction to the cloud computing phenomenon

The term Cloud Computing surrounds many different topics and this fashionable word is often treated as a general and generic concept. Of course cloud computing can be analyzed by a technical, economical or legal point of view, but the main aim of this work is to stress the idea that this term does mean nothing, to understand how laws apply to cloud computing and how cloud contracts operate. More precisely, it is necessary to differentiate between the main, and the most common, types of cloud computing services to find the proper legal discipline.

The cloud-computing phenomenon represents a group of services provided typically via Internet scalable up and down according to user requirements [10]. Therefore, users usually will rent IT resources from some specific service providers instead of purchasing their own. The scalability is central because users will be able to satisfy their own IT needs even if they fluctuate very quickly and over a large range. Moreover, cloud phenomenon will encourage the economical rise up from EU's crisis because of its big economic value1. In fact «the diffusion of cloud computing is expected to generate substantial direct and indirect impacts on economic and employment growth in the EU, thanks to the migration to a new IT paradigm enabling greater innovation and productivity. According to the model developed by IDC, the "No Intervention" scenario" of cloud adoption could generate up to €88 billion of contribution to the EU GDP in 2020. The "Policy driven scenario", instead, could generate up to €250 billion GDP in 2020, corresponding to an increase of €162 billion over the first scenario. Cumulative impacts would of course be even stronger. IDC estimates a cumulative impact for the period 2015-2020 of some €940 billion in the "Policy-driven" scenario, compared to €357 billion in the "No Intervention" one. » IDC continues predicting that the "Policy-driven" scenario could affect, in the job market, with 3.8 million of new jobs against 1.3 million without an intervention.

Many definitions are possible, but cloud services are usually listed into the following three categories [7]:

1. Software as a Service (SaaS).

2. Platform as a Service (PaaS).

3. Infrastructure as a Service (IaaS): these is the classical end user applications such as email applications or document remote backup (for example, Google Drive, SugraSync or SkyDrive).

Essentially the key characteristics are the following:

1. On-demand self-service - a consumer can access computing capabilities without requiring human resources working whit each single service provider.

2. Broad network access - capabilities are available over the network (usually Internet) and the consumer could reach their own data and information using any device.

3. Resource pooling - providers apply a multi-tenant model to serve a large number of consumers. Usually consumers do not know exactly the location of their data (e.g., country, town, or datacenter).

4. Rapid elasticity - capabilities could be elastically provisioned and released to scale rapidly adapting to the incoming demand of hardware/software resources.

5. Measured service - pay per use or pay as you go services. Resource usage can be monitored, controlled, and reported, providing transparency for both the provider and consumer of the used service.

The deployment models are, instead, three or four. It depends if we consider the Community cloud as a single type with its peculiarities. It easier to consider only the most important models and they are Private cloud (where cloud infrastructure is provided for exclusive use of enterprises, or natural persons, and it could be owned and managed by the same organization or by a third party, or by a combination of them), Public cloud (where the structure could be similar to the Private one, but it is provided for open use by the general public) and Hybrid cloud that is a mix of the other ones. For instance, an organization that has its own (private) cloud could use a public cloud to deal with sudden work peak that cannot be managed from their own cloud resources [5]

2 The outsourcing and cloud computing contracts

The term outsourcing identifies the practice of finding resources outside the company, in the sense that (before that it was done internally with own resources) a strategic asset has been moved to a third party [8,9]. This solution - that find a large practical application - is used to move outside to a third party an entire business process and not only because of the simply necessity of some kind of goods or services. The company, in this way, transfers the execution of some internal services (in our interest they are IT services) to an external supplier, which shall implement this kind of requested service with his staff and resources. The usual procedure, which has been developing by all size of enterprises, to delegate and to move outside the internal structure some kind of activities and some specific services, will involve both essential and accessorial business processes. More precisely, this kind of choice, in fact, is implemented not only to contain costs, but also to be more engaged into core business activities. In this way, the company should manage with more benefits all its internal (core) business resources.

More precisely the outsourcing is a business practice that regards all sectors of business activities and not only the specific area of IT services. In this context, it expresses a general organizational model, which is raised from 70s and originated from the necessity to reduce costs, to increase the internal flexibility and the competitiveness of the company, focusing energies and resources over the most important activities of the enterprise [1]. In the area of information technology, the outsourcing phenomenon can hardly be considered a recent invention: it is a matter of fact that outsourcing has gone through a first phase of development, which has substantially coincided with the raise of IT services and data processing in all business sectors and industry branches. In these cases some specialized companies used to provide to their clients the so-called "data service centers" and they were responsible for managing specific procedures (salaries, inventory, invoicing, etc..) for each single client, which generally was not equipped with any IT recourses or it had not any specific competence about IT resources or data processing.

In a subsequent phase, the outsourcing phenomenon was radically changed and, in a certain way, has followed an opposite way of developing respect the first phase above-mentioned. This phase was in fact characterized by the outsourcing practice where the client used to transfer, entirely or in part, its EDP (Electronic Data Processing) asset to an external provider. The distinctive features was that the EDP already existed and was built by a complete IT system with employees with IT expertise.

Nowadays we can affirm, at least in the field of IT outsourcing, that we are playing an original phase. In fact today we are facing, once again, with multinational and international players who are providing to its customers, day by day, more and more specific services. However we are not in an era where both services and, in some case, technologies are almost unknown to the customer. On the contrary, today the customer is aware that it is not appropriate to immobilize human and economic resources for internal, expensive and secondary, activities. For this consciousness, he decides to find them externally. In a certain way, we are moving in a step back to analyze the methodological approach that seems to wink eyes to the past; but on the contrary, we are running toward the future by using new technologies. Today, for many companies, the outsourcing of secondary business processes, especially IT business process, becomes an essential choice, and through certain types of services (yesterday outsourcing contract, today cloud ones), regulated by specific contractual schemes, it is possible to realize, to develop and to manage all kind of IT systems [3]. Starting from the IT system marketplace and looking towards the cloud phenomenon, a mixed framework of contracts is emerging. It is a matter of fact that with this background, we will have to move over different contracts both typical and atypical. It is sufficient, in this regard, to mention the design activity of the entire system, the purchase of hardware equipment, the strictly connected software licenses, the physical realization of the IT system and the complementary technical assistance provided. In all these cases, we are able to analyze more than one single contractual scheme. More precisely, there is an articulated and global economic operation because the (contractual) object is not composed of a single performance, but it is instead made up and framed by multiple, and different, performances connected, with one another, using different contractual schemes, which must all be considered as a unit (Sanmarco 2006).

The global economic operation that is defined by the general, and generic, category of IT contract can be realized only through the procedure of connected contracts and not by

the other mixed contracts procedure. The connected contracts category is the combination of different contractual typical models, where each one has peculiar and qualifying elements, which together connected are able to reach and to realize the overall and original aim wanted, from the beginning, by contractual parties. Otherwise, regarding to the mixed contracts category, characterized by a single contractual model, there are different parties of different typical contracts. For this reason, the legal doctrine use the definition "mixed contract", because this kind of contract is, de facto, a puzzle of specific pieces (contracts element) that are able to build one single (in contrast with the connected contract procedure where there are several contracts), and new, contract.

Anyway, as already noted, the options will be only two: the first one is the presence of a global economic operation that will be analyzed and, later, regulated by the connected contracts theory: the second one it will be represented by the presence of the atypical contract.

3 The supply contract

As we have already seen, each single contracts, with their particular legal discipline, will then be considered within the entire economic operation. In this way, we will have the synthesis of all the interests and the considerations of each contract using one procedure, in order to give greater guarantees for the application of law. However, it does not seem to share the Italian legal doctrine that wants to apply the discipline of supply contract to the IT system supply, on the contrary for the remaining parts (study and design of the IT system, technical and maintenance support, costumer care, etc.) the above mentioned doctrine wants to apply the discipline typical of the work contract.

Is a matter of fact that the realization of an IT system means building of a material good (that is the main characteristic of work contract), but the economical operation cannot be atomized up to this level, both when there is a single contract (when the operation is simpler than the complex one where there is the connected contract) and when there is a connected one [6].

In summary, the manufacturing of a specific good is the element that characterizes the work contract compared with the supply contract, where the main characteristic is to provide services. It should also be noted that if the manufacturing activity is just secondary and necessary to achieve another purpose, which is the main one, we would have a supply contract. Moreover, if the supplier does not perform the work, for example, it is the result of natural forces, it will be an atypical contract but not a supply or work contract where the human activity is necessary.

This entire dissertation is not mere theoretical. In fact, the inclusion of IT service contracts in the category of supply or work contracts is not without consequences. More precisely, the supply contract discipline is different from the other one, referred to the work contract. The qualification of a certain contract, using one of the above-mentioned contractual models, is fundamental to find the proper legal regulation. In fact, not all legal rules laid down by the Italian Civil Code for work contracts are also applicable to IT service contracts. In order to determine which rule is more specific than others to regulate one specific contract (work or supply one), there will not be any kind of help using the literal classification provided by the Italian Civil Code.

In fact, it is clear that some of the provisions - article 1655 (Definition), article 1671 (Unilateral retirement), article 1674 (Death of contractor) - are explicitly referred to both types of contracts. However, it is as much evident that others rules - article 1672 (Impossibility to carry out the work) and article 1675 (Rights and obligations of the contractor's heirs) - are literally referred to work contracts but, in practice, they find application to both contractual types. Therefore, it happens that the jurist cannot use, without critical approach, what it is established by the Italian Civil Code. In fact, he has to find the real contractual consideration, of each single contract, in order to use the proper legal regulation for the specific case.

Applying the above-mentioned criteria, there were found incompatible with supply contracts the following rules: art. 1658 about the supply of the raw material; article 1663 about complaints by the contractor of the raw material defects; article 1669 about defects for real estate built for a long term period; article 1673 about the good deterioration before the consignment; article 1228 related to work contract. In addition, there are not applicable, obviously, the various provisions regulating the work consignment and the others regarding the retroactive effect following the retirement of the contract for non-fulfillment of contract performance (article 1458).

On the contrary, when a specific (supply) contract is stipulated to provide an IT service, there is not any reason to avoid the application of article 1667, paragraphs 1 and 2, of Italian Civil Code, which states the deadline for the notification of defective goods and for suing the contractor. Furthermore, It was also recognized that even in supply contracts, such as in works ones, the client has the right to check the work execution step by step (article 1662).

Therefore, it cannot be sustained the approach that uses, and defines a priori, different types (models) of contracts for the same case. In fact, either we are in presence of an hardware purchase (even a work contract), and then the customer will be a mere purchaser of goods with the result that the problems related to the identification of the discipline will be reduced to the fact that the legal operator must apply the sale contract regulation, or we will face a more complex operation. In this case, that is our cloud computing scenario, even if there is a single (supply) contract of IT services or a connected IT

service contract, the key point will be the requested service to the entrepreneur/supplier (cloud provider).

4 Cloud computing and data protections

The intense flow of data, that usually is generated between the provider and client as a contractual parties of an IT service, concerns information that, in many cases, have the nature of "personal data", because these data are related to third parties directly or indirectly identifiable (employees, suppliers, business partners, customers, etc.). If there is no doubt that these operations involve the management and processing of personal data, the classification of the role played by provider and client, who have an active role (both of them) and, consequently, the definition of their specific responsibilities and obligations may have several legal issues [2].

In general, terms it must be underlined that the EU's legislation in the field of personal data has a data-centric structure, whereby the role assumed by the contractual parties is determined based on their effective activity. Each single contractor will have a different approach and a different role with all data processed in the execution of an IT service.

If there is a wide freedom of decision making power for the definition of the essential characteristics of (personal) data management, by each single contractual party, there will be two separate data controllers, with the result that the information flow, that is established between them, will be necessarily qualified in terms of transmission data between autonomous data controllers. Otherwise, if the data management flow is directed from the user to the provider, there will be a relationship between controller and processor. The main consequence, of this case, is that the (personal) data exchange can be more easily qualified as an "internal" process (between controller and processor).

The implications related to the adoption of one or other organizational model are able to produce consequences on both the operational structure and hierarchical responsibility for data processing. In fact if it is closed the independence of the contractual subjects, the user will benefit from a reduction of costs in terms of regulatory compliance and he cannot be held responsible for violations occurred by the provider (processor). Moreover, we have to consider the issue of possible indirect damages resulting from the unlawful treatment (reputational damages and loss of potential future customers, etc.) resulting from the provider activity. Beyond the possibility of a claim for punitive damages against the provider, the user may more effectively regulate this aspect in advance through the adoption of appropriate contractual clauses that take into account these possibilities. So, he can integrate the contractual covenant with provisions useful to quantify, in advance, the possible damages and he can also establish and define specific cases for the discharge of the contract.

The provider (as a processor) will not have to put in place specific formalities (except if there is a delegation by the controller), in terms of giving information and acquisition of specific consent but on the other hand he will have, from these higher powers, the obligations to improve and monitor security measures required by law. The user (as a controller) will have the duty to choose a processor with adequate technical skills provided by the article 17, paragraph 2, of Directive 95/46/EC.

From data processing point of view, there are not differences between the dissimilar models of cloud computing services [4].

Anyway, some further legal issues can arise when these (personal) data are moving towards "third countries". In accordance with the provisions of the article 4 of Privacy Directive, it is stated that «each Member State shall apply the national provisions it adopts pursuant to this Directive to the processing of personal data where: (a) the processing is carried out in the context of the activities of an establishment of the controller on the territory of the Member State; when the same controller is established on the territory of several Member States, he must take the necessary measures to ensure that each of these establishments complies with the obligations laid down by the national law applicable; (b) the controller is not established on the Member State's territory, but in a place where its national law applies by virtue of international public law; (c) the controller is not established on Community territory and, for purposes of processing personal data makes use of equipment, automated or otherwise, situated on the territory of the said Member State, unless such equipment is used only for purposes of transit through the territory of the Community.» For example, a cloud customer based in Rome, using a Russian cloud provider with servers in France to process personal data, must comply with the Italian Privacy Code (not Russian or French laws).

Many of the IT processes, for reasons mainly attributable to cost containment, currently are located in third countries. These location are obviously different from the cloud provider headquarter whit the consequence that we are testimony of a globalization of data process flows. This phenomenon, in legal terms, not only results in the introduction of international elements but often it implies a different level of protection afforded to the rights of natural and legal persons. Moreover, maybe the most important aspect, these third countries have a different degree of autonomy and a different level of democracy of so called western countries. It is sufficient to think about a data center of one big cloud-computing provider located in a country politically instable. What will happen if a specific government fall down under a coup d'état and all the IT facilities, over there located, are confiscated or destroyed? This profile is not marginal and must be considered very carefully.

5 Conclusions

The two key points that need to be addressed in the cloud-computing era are:

- Data protection;
- Legislation about cloud computing contracts.

The first point is probably the thorniest but, paradoxically, it is the most addressed. In fact, the European Commission is producing a new European regulation (actually a proposal, as said above) with a clear focus over cloud phenomenon. For sure, it is not easy to convert theory into practice but the starting point is well defined. In order to improve the above theory we can use other precious allies like ISO/IEC FDIS 27001 (there is also a recently published ISO/IEC 27013:2012 standard that merges ISO 27001 and ISO 20000-1): it is a landmark for improving information security management system. The basic idea is to understand the real ambient where we are operating, so it is central to adopt the PDCA process model (Plan-Do-Check-Act). Of course ISO 27001 does not make data 100% secure, but it is a useful guidance to approach with common sense the entire lifecycle of our data (and information) and, as a consequence, our cloud service.

Regarding the second key point above-mentioned, we have to understand that it introduces a new idea of contract as a legal instrument: it is nowadays more and more oriented to the entire economic operation concept. It is certain that IT contracts (included the cloud service ones) usually express the parties' need to negotiate contracts not provided by typical legal schemes, even if worthy of protection, because of their socially relevant interests. In this way, to find the most suitable regulation for cloud contracts, a possible solution could be obtained from the Italian legal concept of "connected contracts".

This work want to demonstrate that the cloud-computing contract, as a legal category, does not exist. This is a strong statement but it is evident how not every kind of IT phenomenon produces direct effects over our legal order. The whole Italian legal system has the correct legal instruments to manage every ICT innovation and the constantly necessity of definition of every new contracts that seem to born, represent nothing over than the inactivity of the jurist in front of the new IT challenges. Even in this millennium the role of lawyers, law professor and judges, is day by day more important because all these subjects have to adapt the already existing and provided legal instrument to, in this case, the new cloud computing contracts.

6 References

[1] Bradshaw, S., Millard, C., & Walden, I. (2011). Contracts for clouds: comparison and analysis of the Terms and Conditions of cloud computing services. International Journal of Law and Information Technology, 19(3), 187-223.

[2] Chen, D., & Zhao, H. (2012, March). Data security and privacy protection issues in cloud computing. In Computer Science and Electronics Engineering (ICCSEE), 2012 International Conference on (Vol. 1, pp. 647-651). IEEE.

[3] Delfino, R. (2014). European Community legislation and Actions. European Review of Contract Law, 10(3), 422-426.

[4] Dixit, K., Suman, M. A., & Mishra, M. S. K. (2014). A Survey on Data Protection in Cloud Computing. International Journal of Emerging Trends in Science and Technology, 1(10).

[5] LaPointe, M., Walker, L., Nelson, M., Shananaquet, J., & Wang, X. (2014, October). Comparing public and private iaas cloud models. In Proceedings of the 3rd annual conference on Research in information technology (pp. 69-70). ACM.

[6] Millard, C. J. (Ed.). (2013). Cloud computing law. Oxford University Press.

[7] NIST Definition of Cloud Computing , Recommendations of the National Institute of Standards and Technology. It is available online at http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf (accessed on 27 February 2015) .

[8] Pierazzi E.M., L'outsourcing, in Contratto e impresa, No.6/2009, pp. 1349 - 1364.

[9] Pittalis M., Outsourcing, in Contratto e impresa, 2010, pp. 1006 - 1023;

[10] SMART 2011/0045. D4 – Final Report Quantitative Estimates of the Demand for Cloud. Computing in Europe and the Likely Barriers to Up-take.. The report is available online at http://ec.europa.eu/ information_society /activities/ cloudcomputing/docs/ quantitative_estimates.pdf (accessed on 27 February 2015) .

MTC: Multi-Tiered Cloud Architecture

Wei Xie[1,2], **Chongwu Dong**[1], **Wushao Wen**[1,2,*], **Zhe Xuanyuan**[1], **Yin Jia**[1]
[1]School of Data and Computer Science, Sun Yat-sen University, Guangzhou, Guangdong, China
[2]Joint Institute of Engineering at Carnegie Mellon University, Guangzhou, Guangdong, China

Abstract—*A cloud has both finite bandwidth resources and service capability. Specifically speaking, one mega data center belonging to a cloud is only able to provide services for a certain number of connections that do not exceed a performance threshold. When saturated by bandwidth-consuming connections from a huge number of users, the cloud could not sustain such load, resulting in a low level of service quality, such as high latency and limited bandwidth. In order to solve this problem, we present a new cloud computing architecture named Multi-Tiered Cloud (MTC) architecture. MTC has high scalability and customizability. It supports hybrid pull-push model especially focusing on pushing multimedia resources with high concurrency. We abstract a concept, User Groups capable of Autonomous Programming (UGoAP), to help explain how our architecture is both inward and outward customizable. A case study is also shown to better illustrate the architecture.*

Keywords: Cloud computing, Multi-tiered architecture, Inward and outward customizability

1. Introduction

Cloud computing is a specialized distributed computing paradigm where computing power and resources are provided as service to be accessed remotely via the Internet [7]. It has successfully enhanced the efficiency of resource management, supporting pay-as-you-go fashion, and lowers the up-front investment and operating cost of enterprises. Companies and developers no longer need to throw large capital outlays in hardware to deploy new services, thus also save vast expenses related to maintaining hardware [1]. By purchasing the services with elasticity, startups can even handle huge transient data flow. It seems like that cloud computing provides the tenants with an illusion of infinite computing resources available on demand [6].

However, in most cases cloud providers adopt a centralized architecture that has inherent problems such as unreliable latency and bandwidth bottleneck [8]. There are four categories of typical scenarios in cloud computing: 1) big data processing. (e.g., with MapReduce or Spark model) 2) video streaming. 3) cloud storage. 4) interactive services. (e.g., video chat, cloud gaming). In big data processing scenarios, a centralized architecture is appropriate, benefiting from discount over standard power rates [4]. However, in video streaming scenarios, which are bandwidth-consuming, a centralized architecture will cause the mega data center to be under a heavy bandwidth load. In cloud storage scenarios, the interplay between the server side and user side is usually pull-based (i.e., end users are always the initiator of a connection). Since end users lack the global view of the whole system, it is difficult to achieve the overall optimization. In interactive services scenarios, latency is the essential part that influences the user experiences. Unfortunately, to pursue lower energy and construction costs, cloud service providers prefer to locate mega data centers in a desolate area far away from end users, leading to a higher latency [2].

In general, the current centralized cloud computing architecture is not efficient and scalable in bandwidth consumption and could not meet the stringent latency requirements of evolving applications. Moreover, as applications become more data-intensive and the need for interactiveness continue to grow [6][11], new cloud computing architectures are necessary to bring both bandwidth, storage and computing power closer to the users.

We are motivated by the fact that though content delivery network (CDN) has been the most widely used solution in the industry, it only provides the data and content delivery. CDN does not provide computing service delivery. Therefore, we are inspired to design an architecture that transforms the delivery model from simple 'delivery of data' to 'delivery of services'. Besides, we leverage the feature of locality of CDN to achieve high bandwidth utilization, high end-to-end throughput and low service delay.

In this article, we present a multi-tiered cloud (MTC) architecture. A novel feature of MTC is to decouple the cloud into control plane, data plane, and service plane. Based on such decoupling, the bottom tier of our architecture can be built not only by cloud providers, but also a group of users working in the same company, living in the same residential area, or learning in the same campus. The goals of this architecture are highlighted as follows:

- High scalability: It is convenient to add new nodes to an existed MTC architecture.
- High bandwidth efficiency: MTC has an obviously better bandwidth utilization than a centralized architecture.
- High customizability and encouraging innovations: MTC is both inward and outward customizable and exposes open interface for various content providers

*Corresponding author

(CPs) or ISPs to deploy their services.
- Hybrid Pull-Push Model: MTC provides an efficient mechanism and interfaces to support both pull and push services.
- Co-operative construct: Cloud providers and users can construct MTC in co-operative efforts. Users can build their own local cloud subordinated to the MTC model and interfaces. Then the local cloud can access into MTC core framework.

The remainder of the article is organized as follows. We first present the multi-tiered cloud architecture. Then we make a comparison between MTC and traditional solutions such as CDN, hybrid cloud. After that, we provide a case study of MTC followed by conclusion.

2. Related Work

There are several works in the literature that address the bottleneck of the current cloud models. For example, CDN focuses on the network and content placement optimization problems for deliverying data contents based on locality and caching techniques. Hybrid cloud and cloud federation, which are created by interconnecting multiple independent clouds either among public clouds and private clouds, or among public clouds built by different vendors [9][12], also try to solve scalability and security issues of a centralized cloud.

2.1 Solutions based on locality

CDN is a distributed system that provides fast web content and streaming media delivery based on the geographic closeness to the users. As a distributed approach lacks a global view of the whole system, some research introduced a central control plane designed around centralized optimization to provide great benefit [10]. However, CDN cannot be customized by end users.

2.2 Hybrid cloud

A Hybrid cloud is a cloud composed of two or more cloud types, such as private cloud and public cloud. It hosts resources with a shared resource pool for all the members in the cloud and a private resource pool solely for the private members in the cloud [14]. Fog computing was proposed to address latency problem by providing elastic resources and services to end users at the edge of the network [13]. However, it is expensive to build many micro data centers at the edge or use high-end devices such as IOx [5].

2.3 Cloud federation

In this research track, there are many researches on Intercloud. The concept 'Intercloud' was proposed to support scaling of application across multiple vendor clouds [3]. If a cloud cannot meet the demand of users, other clouds can serve them immediately. Intercloud requires different vendors obeying to a common standard and protocol, and most importantly sometimes vendors do not provide the same cloud service functionality. Such fact causes a big obstacle if Intercloud will apply in the real IT industry.

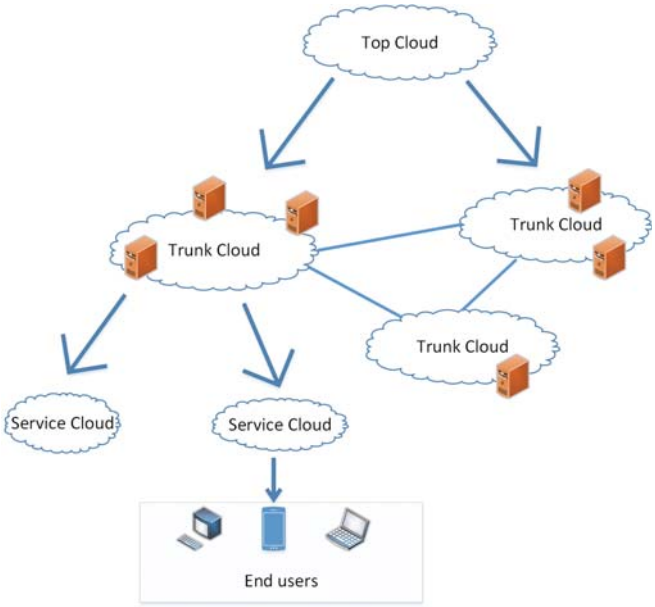


Fig. 1: Multi-tiered cloud architecture.

3. Multi-tiered Cloud Computing Architecture

The MTC architecture consists three tiers as depicted in Figure 1. We refer to the top tier as the Top Cloud, the second layer as the Trunk Cloud, the bottom layer as the Service Cloud. This architecture has one Top Cloud, several Trunk Clouds and many Service Clouds, each belonging to its corresponding Trunk Cloud. Service Clouds are located in end users vicinity, ensuring QoS to end users. A Trunk Cloud can be designed as a distributed data center, which provides basic cloud services like storage capacity and computation capacity. Different Trunk Clouds connect to each other over Internet. Data can be transferred from a Trunk Cloud to another. The Top Cloud serves as a control center which owns a global view of the whole architecture.

3.1 Decouple the control, data and service plane

In MTC architecture, we decouple the control plane, data plane and service plane as depicted in Figure 2. According to the geographical distance to end users and the difference of storage capacity, we divide the data plane into three tiers. According to the range of known information, we divide the control plane into three tiers to establish a top-down control model. By defining interfaces between each tier and each plane, we hide the implementation detail of them and expose

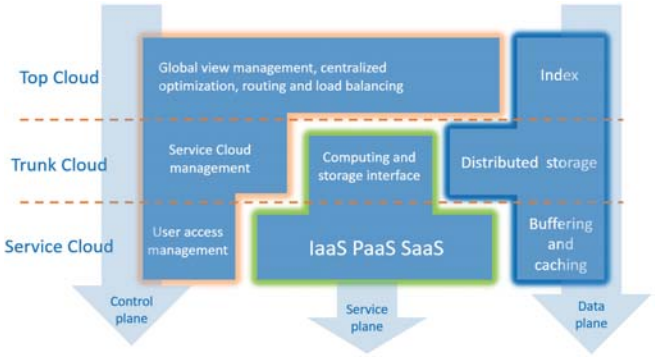


Fig. 2: Decouple the control, data and service plane.

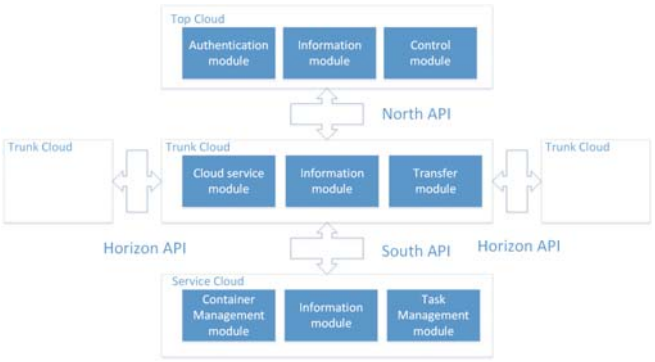


Fig. 3: Design of MTC architecture.

the abstraction of their functionality, making this architecture more agile and scalable.

The control plane is responsible for supervising and directing the whole system. In this top-down control setting, the Service Cloud responsible for managing end users (e.g., user access management) and Trunk Clouds are responsible for managing Service Clouds belonging to it. The Trunk Cloud hands over the control information (e.g., link state, bandwidth load) to the Top Cloud for a global view and optimization (e.g., centralized optimization and load balancing).

The data plane is mainly constituted by Trunk Clouds. A Trunk Cloud can be implemented by a distributed data center, which can provide storage and computation capacity. The Service Cloud has the mechanism to buffer data that is frequently being used by end users, relieving the bandwidth load between the Trunk Cloud and the Service Cloud. Data can be transferred between Trunk Clouds. The Top Cloud stores the data index to identify which Trunk Cloud contains the users' desired data and controls the data transfer from a Trunk Cloud to another.

The service plane eventually provides services to end users. It takes advantage of the geographic closeness to end users, ensuring a good QoS to them. The Service Cloud can run various applications in separated application containers, thus supporting multi-tenants and enabling a PaaS service with very low latency. Compared with the Trunk Cloud, the Service Cloud has a limited storage capacity and computation capacity. So encountering a heavy data or a heavy computation tasks, the Service Cloud delivers these tasks and data to the Trunk Cloud. The Service Cloud continually synchronizes user data with the Trunk Cloud periodically and just maintain the frequently used data for users.

3.2 Design of MTC architecture

The design of MTC architecture is depicted as Figure 3. The Top Cloud consists several servers located in different areas, responsible for controlling the whole system. As the Top Cloud server and the Trunk Cloud are scalable, this architecture supports convenient appending of new server and new Trunk Cloud, which is controlled by the authentication module. The Top Cloud, Trunk Cloud and Service Clouds have the information module, and the information, containing performance states like storage load and available bandwidth, is shared between adjacent clouds. The Top Cloud leverage the control module to process data and task flows. The cloud service module is used to provide storage and computation services while the transfer module is used to transfer data from a Trunk Cloud to a Service Cloud or between Trunk Clouds. The container management module manages the application containers to run applications and provide PaaS services. User tasks are scheduled by the task management module.

The Top Cloud interact with the Trunk Cloud though the north API. The Top Cloud can direct the transfer of data between Trunk Clouds. For example, when data is stored in unbalance between Trunk Clouds, the Top Cloud can recognize this unbalance by checking its information module. To adjust this unbalance, the Top Cloud can give a command to transfer data from a high-load Trunk Cloud to a lower one when idle bandwidth is sufficient. In some cases, when a Trunk Cloud needs some data located in other Trunk Cloud, such data transfer will also happen. Once this transfer occurs, a Trunk Cloud transfers its data to another through the horizon API. The north API also support the Top Cloud in pushing data to end users directly.

The horizon API is the interface between different Trunk Clouds. Each Trunk Cloud has several storage nodes. A file with considerable size will split into a series of slices. These slices and their duplicated slices are arranged to be stored on different nodes. When receiving the command from the Top Cloud to transfer files from a Trunk Cloud to another, the transferring initiator invokes the horizon API to manipulate the nodes to transmit data to nodes of other Trunk Cloud. Such multi-nodes to multi-nodes data transfer can reach high throughput as depicted in Figure 4.

Service Clouds can provide an environment for applications to run in the application containers. Applications in these containers can use storage and computation capacity

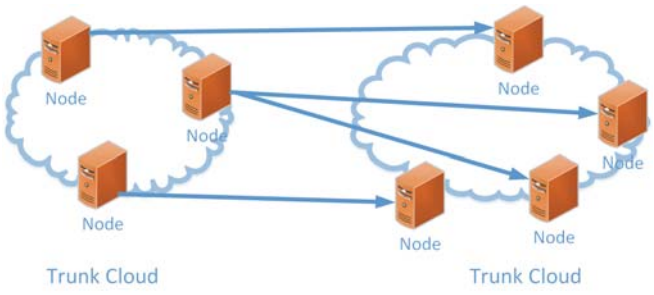


Fig. 4: Horizon API for data transfer.

provided by the Trunk Cloud. Service Clouds invoke the south API to leverage these cloud services. Service Clouds intrinsically provide services to end users. Service Clouds have a standard interface for new services to be deployed, so that Trunk Clouds can deploy new services on Service Clouds easily and a group of people can construct or customize their own Service Cloud conveniently.

3.3 UGoAP based customization and hybrid pull-push model

We abstract a concept, user groups capable of autonomous programming (UGoAP), to help in explaining the customization feature of MTC architecture. UGoAP refers to a group of users that have the ability to program based on a standard API. For example:

- A user group in a campus which consists of students and professors.
- A user group in a company with IT department.
- A user group in a government organization which has outsourced their IT function.
- A user group in an intelligent residential area of which the property management company is capable of programming.

These groups have the ability to achieve the fine granularity customization. Once one architecture provides the standard API (e.g., REST or HTTP API), they can deploy all kinds of services for their specific demands, thus bringing the service clouds high customizability.

The Service Cloud provides both inward and outward customizability compared with a traditional CDN solution or a hybrid cloud. Regarding CDN, CDN is usually deployed by cloud providers or a third-party CDN company. So the end users do not have the privilege to operate the detail of the CDN server. As for hybrid cloud, Companies usually put their sensitive and private data in a private cloud and put their large size files and computing data on a public cloud. The private cloud can connect to the public cloud, but it does not expose API to outside. The outside usually do not know the presence of the private cloud of a company. The content providers have little chance of distributing their data and application on the private cloud. The Service Cloud can provide inward customizability. The services cloud can provide IaaS, PaaS, and SaaS to the inner users. Users can deploy their application on services cloud, which we referred as inward customizability. Moreover, the Service Cloud also exposes open API to outside, which supports different ISPs to deploy their services and different content providers to deliver their data.

In a pull-based model, the users usually serve as the initiator. Different initiator does not know the state, information of connection and status of links. In our architecture, the Top Cloud has the global view of the whole system, which can:

- For the large-scale foreseeable video streaming distribution (e.g., The World Cup), it can provide predistribution when bandwidth is idle.
- Achieve better load balancing based on the transferring content, status of link and the idle states of Service Clouds.
- Support more business form (e.g., auto upgrade of applications and information notification)

By introducing the hybrid pull-push model, our architecture can support the variety of demands of application and increase the bandwidth utilization.

3.4 Comparison with existing architecture and model

As shown in Table 1, under these four typical scenarios, we contrast three traditional approaches to our architecture. In the big data processing scenario, mega data center which adopts centralized power supply and cooling is fit, beneficial from discount over standard power rates. So it is cost-efficient, while other approaches do not have such advantages. In the video streaming scenario, a centralized architecture has low bandwidth utilization. The mega data center needs to establish different connections to different users. Many approaches use the locality designs to provide fast content distribution. The locality designs can be implemented by CDN and hybrid cloud. In the cloud storage scenario, traditional architectures and models are usually pull based and are activated by end users. Our architecture supports hybrid pull-push based model. In the interactive scenario, our architecture provides both inward and outward customizability. Content providers can deploy their data, information and even application on the services cloud which is close to the end users. That can help to achieve a low-latency QoS.

MTC can be beneficial to all the roles related to cloud computing, such as end users, content providers, cloud providers and ISPs. From the perspective of end users, they can experience a higher QoS (e.g., more fluently video streaming and lower latency). From the perspective of content providers, they can reduce some cost of building the CDN infrastructure. From the perspective of cloud providers, the expense also can be cut since MTC can be constructed

Table 1: Comparison with existing architecture and model.

		Big Data Processing	Video Streaming	Cloud Storage	Interactive services
Sample applications or models		MapReduce and Spark model	YouTube; Twitch	Dropbox; OneDrive	Cloud gaming; Video chat
Challenges		The scale of computing cluster	High bandwidth ensurance	Content security and privacy ensurance	Latency and response time
Features	Mega DC	Cost efficient. Beneficial from discount over standard power rates	Bandwidth-consuming applications cause bottleneck	Pull based	Relatively high latency
	CDN	None	Use locality to provide fast streaming delivery	Pull based	Do not support customization from users
	Hybrid Cloud	Depend on public cloud	Use locality to provide fast streaming delivery	Pull based	Do not support customization from ISP and content provider
	MTC	Depend on the Trunk Cloud	Use locality to provide fast streaming delivery; Centralized optimization	Hybrid pull push based	Support high customization

in co-operative efforts. From the perspective of ISPs, the bandwidth efficiency of MTC can release the congestion on the core network.

4. Case Study

In this section, we elaborate an example case in a video surveillance system. The case is that the government is constructing a video surveillance system as shown in Figure 5. The government instructs a hypermarket to connect into the system. During the process, technical operations are outsourced to a video processing company. We show how MTC is superior in this case. The reality requirements are: 1) the hypermarket does not allow all videos to be uploaded, since machine rooms and storage rooms involve privacy or confidentiality. Therefore, the hypermarket requires configuring their privacy policies as a filter. 2) The government only requires a subset of the videos. For example, the government just wants videos from 6:00 AM to 11:00 PM (i.e., duration policies as a filter). 3) The video processing company may want to pre-process (such as video compression and video content indexing) videos in the hypermarket locally before uploading, to reduce the bandwidth consuming. In MTC, the Service Cloud provides policy customization and the environment for applications running. For inward customizability, the hypermarket as the end user can provide privacy polices as a filter. For outward customizability, the government can define other filters and the company can deploy their proprietary applications. After passing through these filters and applications, the videos become smaller in size and volume.

In the above paragraph, we observe the case from the perspective of the Service Cloud. Now we observe it from the perspective of a global view as shown in Figure 6. The company have deployed a Deployment Management System (DMS) in the Trunk Cloud to achieve the automatic deployment from the Trunk Cloud to the Service Cloud. The DMS makes the deployment convenient from a Trunk Cloud to multiple Service Clouds by a top-down control, a key feature of MTC. When videos transfer back to the Trunk Cloud after filtering and processing, the company transfers videos to the government's storage system in other Trunk Cloud by the horizon API. The government uses their storage

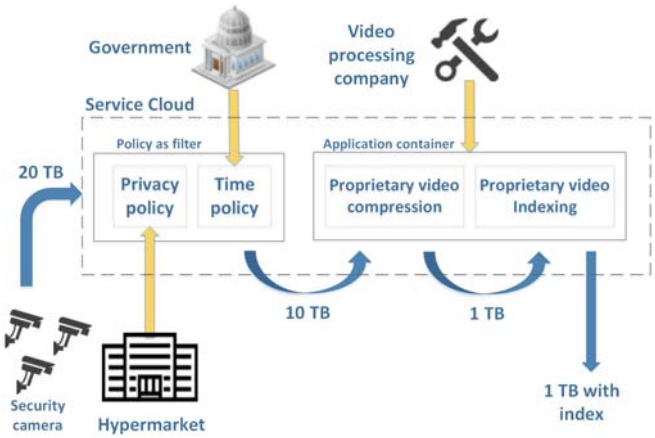


Fig. 5: Video surveillance system in the Service Cloud. The input is 20-TB surveillance videos. After the process of the Service Cloud, the output is reduced to 1-TB videos that need to be transfered. We have saved the broadband resources.

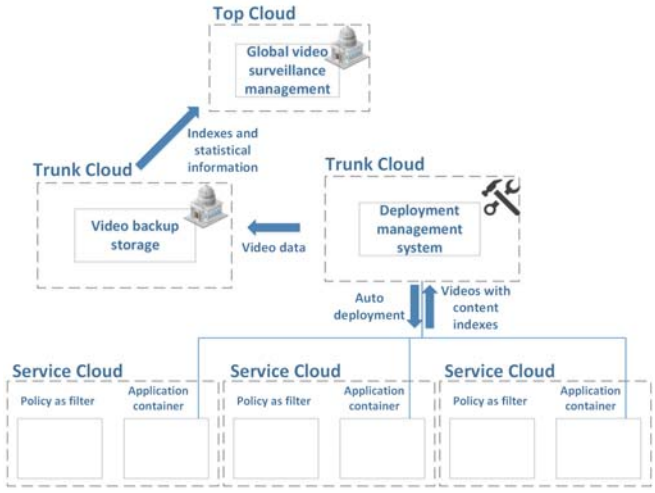


Fig. 6: Video surveillance system overview in MTC. Every Service Cloud can set their own filters and these filters can be auto deployed by a management system running on the Trunk Cloud. All the video data can flow across Trunk Clouds.

system to back up these videos, which can be retrieved for taking evidence. After all above procedure, the government gathers the statistical information and indexes and transfer them to the Top Cloud for universal control and scheduling.

Traditional CDN solutions and hybrid cloud cannot achieve this since the case requires the local components to have both inward and outward customizability. First, the case requires videos to be processed close to end users (i.e., the locality feature). CDN can support the feature but it is transparent to the users. The users cannot customize CDN (CDN has no inward customizability). Second, hybrid cloud is not suitable since the case requires the policy configuration from the government and the application deployment from the company (hybrid cloud has no outward customizability).

MTC shows three superiorities in this case that makes it outperform CDN solutions and hybrid cloud. 1) From the perspective of resource utilization, MTC makes a trade-off between the computation capacity and the bandwidth consuming. Since the videos pass through the filter and the pre-processing, we leverage the computation capacity of the Service Cloud to reduce the bandwidth consuming between the Service Cloud and the Trunk Cloud. 2) From the perspective of service demand, MTC can flexibly fulfill various requirements such as privacy requirements and duration requirements. 3) From the perspective of the whole architecture, the Top Cloud has a global view and it has a high control over the whole system. When the number of the Service Clouds increases or massive data flows occur, the Top Cloud can achieve some centralized optimizations such as link optimization and load balancing in MTC.

5. Conclusions

In this article, we have proposed a novel cloud computing architecture. The key part of this architecture to decouple the control plane, data plane and service plane. In addition, MTC architecture supports hybrid pull-push services and focusing on pushing multimedia resources with high concurrency. It is a high scalable architecture and it is easy to be customized by users. MTC supports both inward and outward customizability. We believe a highly customizable multi-tiered architecture will be prevalent in the near future as applications continue to become more data-intensive and require more stringent latency.

References

[1] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica *et al.*, "A view of cloud computing," *Communications of the ACM*, vol. 53, no. 4, pp. 50–58, 2010.

[2] M. F. Bari, R. Boutaba, R. Esteves, L. Z. Granville, M. Podlesny, M. G. Rabbani, Q. Zhang, and M. F. Zhani, "Data center network virtualization: A survey," *Communications Surveys & Tutorials, IEEE*, vol. 15, no. 2, pp. 909–928, 2013.

[3] R. Buyya, R. Ranjan, and R. N. Calheiros, "Intercloud: Utility-oriented federation of cloud computing environments for scaling of application services," *Algorithms and architectures for parallel processing*, pp. 13–31, 2010.

[4] K. Church, A. G. Greenberg, and J. R. Hamilton, "On delivering embarrassingly distributed cloud services." in *HotNets*, 2008, pp. 55–60.

[5] A. V. Dastjerdi, H. Gupta, R. N. Calheiros, S. K. Ghosh, and R. Buyya, "Fog computing: Principals, architectures, and applications," *arXiv preprint arXiv:1601.02752*, 2016.

[6] A. Fox, R. Griffith, A. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, and I. Stoica, "Above the clouds: A berkeley view of cloud computing," *Dept. Electrical Eng. and Comput. Sciences, University of California, Berkeley, Rep. UCB/EECS*, vol. 28, no. 13, p. 2009, 2009.

[7] Y. Jadeja and K. Modi, "Cloud computing-concepts, architecture and challenges," in *Computing, Electronics and Electrical Technologies (ICCEET), 2012 International Conference on*. IEEE, 2012, pp. 877–880.
[8] J. Lloret, M. Garcia, J. Tomas, and J. J. Rodrigues, "Architecture and protocol for intercloud communication," *Information Sciences*, vol. 258, pp. 434–451, 2014.
[9] P. Mell and T. Grance, "The nist definition of cloud computing," 2011.
[10] M. K. Mukerjee, D. Naylor, J. Jiang, D. Han, S. Seshan, and H. Zhang, "Practical, real-time centralized control for cdn-based live video delivery," in *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication*. ACM, 2015, pp. 311–324.
[11] T. V. Seenivasan and M. Claypool, "Cstream: neighborhood bandwidth aggregation for better video streaming," *Multimedia Tools and Applications*, vol. 70, no. 1, pp. 379–408, 2014.
[12] D. Villegas, N. Bobroff, I. Rodero, J. Delgado, Y. Liu, A. Devarakonda, L. Fong, S. M. Sadjadi, and M. Parashar, "Cloud federation in a layered service model," *Journal of Computer and System Sciences*, vol. 78, no. 5, pp. 1330–1344, 2012.
[13] S. Yi, C. Li, and Q. Li, "A survey of fog computing: concepts, applications and issues," in *Proceedings of the 2015 Workshop on Mobile Big Data*. ACM, 2015, pp. 37–42.
[14] L. Zhao, F. Wang, and K. Fan, "A secure and fair resource allocation model under hybrid cloud environment," in *Systems and Informatics (ICSAI), 2014 2nd International Conference on*. IEEE, 2014, pp. 969–973.

Decision Models for Cloud Computing

Hong Wang[1, 2]
[1]School of Business and Economics, North Carolina A&T State University, Greensboro, NC, USA
[2]School of Information, Yunnan University of Finance and Economics, Kunming, China

Abstract - *With cloud computing being around for many years and most organizations wondering if they are ready to adopt this concept, a proper decision model that is used to help them make the right decision and its proper use in business environment becomes inevitably necessary. In this study, we analyze different business scenarios that warrant different solutions by using the decision models developed using some well-established decision theories.*

Keywords: Decision theory, decision model, cloud computing

1 Introduction

Cloud computing has been present for only about a decade in our current society; however, the concept of cloud computing along with its technologies and applications have been steadily gaining ground in today's business use of computers. In order to understand the history of cloud computing and the background of this study, let us take a look at the November 28, 2012 edition of the Information Week that presented an article by Charles Babcock (Babcock, 2012) that introduced 10 cloud computing pioneers (of which we only choose seven of to present here). Babcock claims in his article that his list is neither exhaustive nor all-inclusive. And, undoubtedly, there will be other lists, highlighting other quiet innovators whose names we're just beginning to hear, and whose accomplishments will be no doubt well-known in the coming years.

Werner Vogels, CTO and VP of Amazon Web Services, joined Amazon in 2004 as the director of systems research, coming from a computer science research post at Cornell University. He's had a vision of a new type of a distributed system, one that relied on inexpensive parts but could scale out infinitely, making the Amazon Compute Cloud elastic and not come to a halt if a piece of hardware failed underneath it. He was an advocate of Amazon getting into the business of distributing virtual server computing cycles over the Internet and charging on a basis of time. Before Werner Vogels got a cloud infrastructure to evangelize at Amazon, there was Chris Pinkham, designer of Amazon Enterprise Compute Cloud (EC2). Pinkham was the project's managing director while Amazon software architect Christopher Brown was lead developer. Together, Pinkham and Brown produced Amazon's first public cloud infrastructure. Pinkham had the knowledge of how things needed to scale in a Web service environment. Soon, both he and Brown then set about exploiting the possibilities of a fully virtualized data center to allow for virtual server launch, load balancing, storage activation and adding services such as database.

In the early days of cloud computing, NASA CTO Chris Kemp took several leading concepts of how to assemble a low cost, horizontally scalable data center and put them to work at the NASA Ames Research Center in Mountain View, Calif. One concept was placing banks of standard x86 server racks in a shipping container with a single power supply and network hookup. Kemp initially created the Nebula cloud project to collect big data from NASA research projects, such as the Mars mapping project. But Kemp also conceived of a mobile cloud data center that could be transported to different locations to provide onsite compute power. In addition, he initiated the OpenStack open source code project when NASA sought to team up with Rackspace to combine

cloud computing software assets. In March 2011, Chris Kemp resigned his post with NASA, an agency with which he had dreamed of working since he was a child, to become founder and CEO of Nebula.

Randy Bias, cofounder and CTO of CloudScaling, has been a specialist in IT infrastructure since 1990. He was a pioneer implementer of *infrastructure-as-a-service* as VP of technology strategy at GoGrid, a division of hosting provider ServePath. GoGrid launched a public beta of its Grid infrastructure in March 2008. He pioneered one of the first multi-platform, multi-cloud management systems at CloudScale Networks and went on to found CloudScaling, where he was a successful implementer of large-scale clouds based on a young and unproven open source code software stack, OpenStack.

Jonathan Bryce partnered with website designer and friend Todd Morey to host sites on their own rented servers in Rackspace. They left Rackspace in 2005 to branch out into their own website building and hosting business, Mosso Cloud. But Mosso still ran on servers in the Rackspace data center. Rackspace executives saw the relationship between its hosting-services business and emerging uses of cloud computing, so they asked Bryce to keep building out the Mosso Cloud. Bryce later rejoined the company as the head of Rackspace Cloud.

Marc Benioff, CEO of Salesforce.com, stands out as the pioneer and guerrilla marketer of *software-as-a-service*. He drew much attention to the concept at a time, when it was widely disregarded as an aberration of limited use, by brazenly advancing the concept of cloud services as the "death of software," in which he meant that on-premises software, the systems that have been keeping enterprise data centers running since 1964, were starting to go away, replaced, instead, by software running in a remote data center accessible over the Internet.

As the senior VP for technical infrastructure at Google, Urs Holzle led the design and build-out of the search engine's supporting infrastructure and ended up supplying a pattern for Amazon, Microsoft, GoGrid and others to follow. A Google data center is designed to use about half the power of a conventional enterprise data center.

Frank Frankovsky worked as Dell's director of Data Center Solutions during the crucial period of 2006-2009, building up the hardware maker's ability to sell rack-mount servers to search engine and Web service companies seeking to build new, more efficient data centers. In October 2009, Frankovsky became the director of hardware design and supply chain at Facebook during a crucial period in its expansion. In April 2011, Mark Zuckerberg and other Facebook officials announced the launch of the Open Compute project to set standards for efficient cloud servers.

2 Different business scenarios

We may classify the different business scenarios by using a variety of methods. One such method is to recognize the size of the business organizations. The rationale behind this method is that the size of organizations is closely related to the complexity of decision making processes (see Simon, 1991, Wang, 2004, Wang and Chu, 2004), i.e., as the size of organizations increases, the complexity of decision processes increases as well. When we study the business scenarios in which we make decisions, we have to consider some of the most influential factors, where size is one of them. Another way to classify business scenarios is to recognize the type of organizations, for example, businesses vs. nonprofit organizations.

2.1 Small and medium-sized enterprises (SMEs)

While different countries have their own standards of classifying the SMEs, European Union is one of the first that defines SMEs (European Union, 2003). Here, we only focus on the number of employees in SMEs, not the "Turnover" or the "Balance Sheet Total", because they are less relevant to our problem. The number of employees in small-sized enterprises is less than 50 while in medium-sized enterprises is less than 250. With such a small number of employees, and the fact they are normally privately owned, SMEs have become the "Innovation Center" of any industry in the business world. In addition, SMEs outnumber large organizations and hire many more people. These characteristics of SMEs determine their decision making process will be unique. Table 1 provides the decision process characteristics for small and medium-sized enterprises.

Table 1. The characteristics of decision making process for small and medium-sized enterprises (SMEs)

Decision Aspects	Requirements
Decision process	*Simple*
Decision time	*Short*
Decision accuracy	*Not important*
Decision constraints	*Limited*
Project/budget size	*Small*
Project period	*Short*

2.2 Large enterprises

Large enterprises in turn will have almost all the characteristics opposite to the ones for SMEs; therefore, their decision making process must be different from the one used in SMEs. In addition to simply the huge difference in sizes of the two types of enterprises, many of those large enterprises are owned publicly, meaning there are many "owners" of the organizations. The fact that owners are usually not managers or decision makers will separate the personal interests and the business interests and lead the decisions to be more "rational." Table 2 demonstrates the characteristics of decision making process for large enterprises.

Table 2. The characteristics of decision making process for large enterprises

Decision Aspects	Requirements
Decision process	Complex
Decision time	Long
Decision accuracy	Important
Decision constraints	Many
Project/budget size	Large
Project period	Long

3 Decision models

The decision making process is actually a problem solving process. First, we need to understand the problem to be solved, then develop a solution or find as many alternative solutions as possible before we choose the "right" one using predefined criteria. We discuss three different decision models in this section.

3.1 A general decision model (Herbert Simon's three-step model, Simon, 1960).

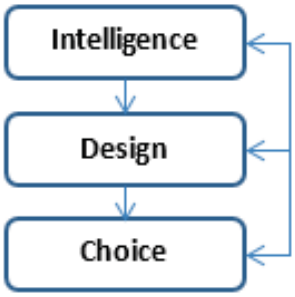


Figure 1. Herbert Simon model of decision making

Herbert Simon's three-step decision making model is widely adopted in solving general decision problems (see Figure 1). "Intelligence" includes data collection, problem identification, problem classification and problem statement; "Design" includes setting criteria for the choice, searching for alternatives and predicting for outcomes; while "Choice" includes sensitivity analysis, selection of the best/good alternative, plan for implementation/action and design for a control system. After a choice is made, if it is decided to be unsatisfactory, we may return to "Design" stage to explore other options, or even return to "Intelligence" stage to verify the problem. Simon pointed out that the outcome of the "Choice" stage may be satisfactory for small problems and "satisficing" for large problems. It complies with his "Bounded Rationality" theory (Simon, 1991).

3.2 For small-to-medium sized enterprises, an Intuition Model is developed.

Since decision problems for SMEs are small in accordance to the size of the enterprise, it is easy to observe the requirements, the available options of cloud computing services, and therefore the managers or the owners of the enterprises will initiate the decision process and make the decision with input from internal and external sources. Figure 2 depicts the decision model used by SMEs.

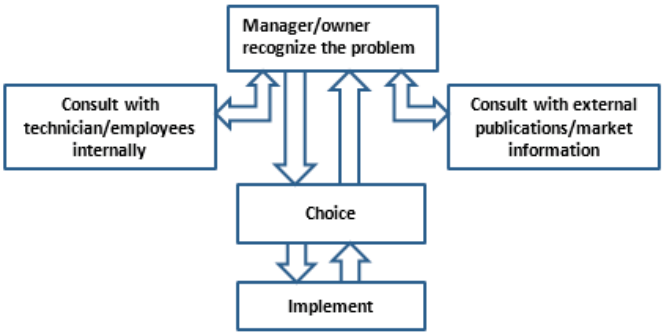


Figure 2. The Intuition Model for small and medium-sized enterprises (SMEs)

Once again, when the choice implemented is not satisfactory, the feedback goes to the manager/owner to further explore different options. After this takes place, an adjustment will be made accordingly.

3.3 For large organizations, a Scientific Model is developed (using Herbert Simon's Bounded Rationality theory, Simon, 1991)

Decision problems for large organizations are usually large and extremely time consuming. When decisions are to be made, there are many factors to be considered, including the impacts of the decisions, which are normally very significant. Since cloud computing will affect the IT infrastructure of an organization, its business processes and even its business models may be changed solely because of the adoption of cloud computing. A Scientific Model is developed for this decision making process. In this model we use the System Development Life Cycle (SDLC) technique, the Group Decision Support System (GDSS), and a variety of business analyses such as Cost/Benefit analysis, Sensitivity analysis, User Satisfaction analysis, Service Level Agreement (SLA), Security, Big Data and Internet of Things (IoT). A Pilot program is needed for testing and analyzing. After that, once again, the feedbacks from different stages will be returned to the previous level for adjustment if needed. Figure 3 shows the components of this model.

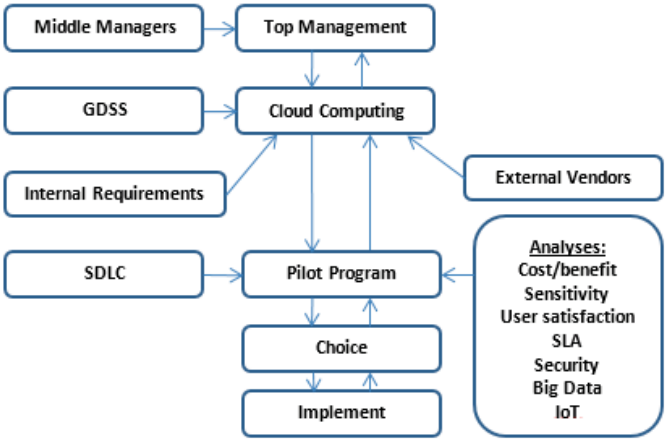


Figure 3. A Scientific Model for large enterprise decision making

4 Conclusions

We have developed two decision models in this study that are to be used for different types of enterprises or businesses, mostly based on the size of the enterprise or business in question. These models are developed using "Bounded Rationality," a well-known decision theory by Herbert Simon (Simon, 1991). We first classify the types of enterprises by using the European Union standard (European Union, 2003) and then analyze the characteristics of decision processes for them. We conclude that the models for decision making used by different types of enterprises should be different from those of the other types of enterprises. We believe the models developed in this study will help both small and large enterprises make a good decision for their cloud computing needs. These decision models may also be applied for the sub-problems of cloud computing such as adoption of infrastructure as a service (IAAS), platform as a service (PAAS) and software as a service (SAAS).

A further study in modeling may include decision models for special types of organizations such as government agencies and nonprofit organizations (NPOs). Nonprofit organizations are also called non-business organizations. The key difference between businesses and nonprofit organizations is the regulation requirements and the stakeholders. While many nonprofit organizations are not government agencies, their goal is also to provide public services or values to a specific group of people, but to not gain profit at the same time. And the stakeholders of nonprofit organizations are similar to those of

government agencies. The impact of their decisions and the decision models to be used for both nonprofit organizations and government agencies will therefore no doubt be drastically different from the ones used by business enterprises, large or small/medium.

5 References

[1] Charles Babcock. "10 cloud computing pioneers". Information Week, 2012.

[2] Herbert Simon. "Bounded Rationality and Organizational Learning". Organization Science, 2, 1, 1991.

[3] Hong Wang. "An empirical study on using decision support systems to solve very large choice decision problems.". International Journal of Management and Enterprise Development, 1, 4, 2004.

[4] Hong Wang and P.C. Chu. "The impact of problem size on decision processes: an experimental investigation on very large choice problems with support of decision support systems." Expert Systems, 21, 2, 2004.

[5] European Union. "What is an SME? - Small and medium sized enterprises (SME) - Enterprise and Industry". ec.europa.eu, 2003.

[6] Herbert Simon. "The new science of management decision." The Ford Distinguished Lectures, 3, 1960.

SESSION

CLUSTER AND DISTRIBUTED COMPUTING + SUPERCOMPUTING

Chair(s)

TBA

ISBN: 1-60132-436-7, CSREA Press ©

ISBN: 1-60132-436-7, CSREA Press ©

Distributed Collaborative Caching

Wardwell, Steven
Contact Author
Department of Computer Science,
Rochester Institute of Technology,
70-3005 102 Lomb Memorial Drive,
Rochester New York, 14623 United States
sjw1579@rit.edu

Bischof, Hans-Peter
Department of Computer Science,
Center for Computational Relativity and Gravitation,
Rochester Institute of Technology,
Rochester New York, 14623 United States
hpb@cs.rit.edu

Abstract— *In this paper we explore distributed collaborative caching. Caching in distributive networks can improve data access performance and reduce dense communication between the client and the server. Collaborative caching can improve caching by allowing clients to share and organize (coordinate) their cached data. This paper will describe our profile algorithm and its implementation and algorithms that presently exist in the literature and their implementation. The object is to establish a baseline of current research existing at the present time to allow for implementation of our proposed algorithm. We will evaluate the algorithms for efficiency, accuracy, disk or server access, hit and miss rates etc. Our presentation of our algorithm of profiling client cache data will consist of constructing a real time profile of cache data to predict a likely client(s) match for data requests. Simulation and graphing analysis will show the effectiveness of our scheme.*

Keywords: Profile, Distributed, Collaborative, Adaptive, Cache, Score

1. Introduction

In various networks, file sharing and data sharing are widely used. One such technique to improve the performance of wired, wireless or distributed network sharing is caching. The aim of this type of caching is to reduce traffic, congestion, data accessibility, and bandwidth management of resources, such as battery power [5]. Caching schemes typically do not facilitate data access based on knowledge of distributed data schemes [5]. Our work attempts to bridge this gap between data access and knowledge of distributed data. In simple caching, it is the requesting node that performs the caching. This cached copy is then utilized by the node to service subsequent requests, as they are needed. In most cases, the requester must retrieve data from the data center or in our case, the "Server/Disk" when missed cache requests occur. The caching of frequently requested data in distributed environments can potentially improve data access, performance and availability. Cooperative or Collaborative caching in distributed networks allows for the sharing and organization of cached data between and/or among various clients or devices and groups[2]. Things such as mobility and resource limitations, as well as limited device battery power, make some collaborative caching management schemes more and more attractive [2]. The ultimate goal is limited server and/or disk access.

In this paper, we propose a distributed collaborative caching scheme based on previous algorithm work, simulation and evaluation. We further suggest our own "Cache Profile" scheme which builds off and utilizes previous ideas and incorporates them into our framework and simulation. Within this paper, we describe the design of our basic framework or simulation architecture and its clients, Content Manager, thread communication layer, logger, various scripts and server. We will describe the basic idea of our cache profile mechanism and what constitutes a "good" profile among client caches. We will describe how each client will use a similar mechanism to the "Summary Cache" [3] to update the Content Manager with its profile at various times or percentages of cache data. We will also describe how the Content Manager will act as a pseudo intelligent proxy, much like how the "Adaptive Cache" [2] mechanisms work in our earlier researched algorithms that evaluate each clients broadcast profile and group/cluster like profiles. It is based upon various criteria or threshold and will mark each related client in the cluster from a strongest to weakest match. This clustering idea was based on previous work found in adaptive collaborative cache mechanisms [2]. We will also describe the mechanism by which the client will retrieve its matching cluster information. It will make a direct request on a fellow node or client for fast retrieval to make use of limited resources and battery power in real world applications.

We will also show, through graphical analysis from simulation studies, predicated actual results of our work. Our contribution will be a unique way of viewing profile data from client caches, making a semi-intelligent Content Manager while cutting down on the number of hops to clients and server requests. The data retrieval among clients will form a collaboration of their data which will be beneficial for limited resources in distributed networks and further the

progress for distributed collaborative caching.

2. Related Work

There has been a vast quantity of work performed in the area of collaborative cache and distributed collaborative cache to improve performance, such as "Summary Cache" [3] and Adaptive Caching algorithms [2]. Most of the adaptive work has been carried out on MANETS and mobile networks. We have applied that work to our static distributed simulation work. In the "Summary Cache" [3] algorithm, different proxies (every other) store a summary of client's cached data in something called a directory of data. The local proxy shall be required to check the stored summaries to determine if the requested document might be present in other proxies. If the document is not found, the request is then forwarded on to the web server [3]. The summaries are not required to be up-to-date or accurate. The updates may occur at regular specified intervals or at the time a certain percentage of the cached documents are not reflected in the summary (unchanged data) [3]. Our representation of this summary proxy is reflected in the content manager that performs the same work.

Other related work is known as "Adaptive Caching with Heterogeneous Devices" [2]. This work was performed in a mobile peer-to-peer network. Our work is a static distributed network, but we have adapted it to basic ideas from this paper. Many applications today, such as file sharing or multimedia streaming such as Netflix [6], are widely used in wireless networks. Different users may carry heterogeneous mobile devices with different transmission ranges, latency, and cache sizes[2]. Service is usually provided by Mobile Support Stations [2], much like the proxy mentioned previously or similar to our Content Manager. Mobile peer-to-peer (P2) networks can also provide services to each other. This P2P would require a cache mechanism that could handle heterogeneous devices and sharing among themselves [2]. This paper[2] proposes a cache scheme that is adaptive to the actual device condition and its neighbors (clients). In this scheme, a distinction is made between what they call "Strong" peers that retain popular data to do self service work. This protects what they refer to as "Weak" peers with their limited cache space. We have adapted their work to fit our simulator, using what we call "Client-to-Client" services.

The "Adaptive Caching with Heterogeneous Devices"[2] paper also builds from another called "A caching and Streaming Framework for Multimedia"[4]. It investigates the middle ground on caching and streaming technologies for multimedia[4]. The algorithm described in this paper is meant for broadband architectures. The foundation of that work is based on the internet standard "Real Time Streaming Protocol" or RTSP [7]. They also use a proxy manager or what they call an "Intelligent Agent" or "Broker", much like our Content Manger that has some "intelligence". Their scheme is much like an enhanced RTSP that maintains state information, similar to the "Summary Cache" [3]. The broker contains their cache algorithm's and runs them, our algorithms are run in the individual clients cache and/or the "Content Manager". The main goal of their work attempts to create the right model for RTSP to perform broker-based streaming/cache architecture.

Our work for "Distributed Collaborative Cache" builds upon the previous work mentioned by simulating the algorithms, environment and incorporating it into our "Cache Profile System". It utilizes the proxy/summary basic idea and intelligent idea while applying them into our individual clients and Content Manager.

3. Algorithm

The algorithm is our "Profile Algorithm". It is often called "SPro" for lack of a better term. This algorithm was built using some of the ideas from the Summary algorithm [3] mentioned above. The basic idea was derived from reading countless collaborative cache and adaptive cache papers. No one paper is responsible for the general idea, except for ideas from the Summary algorithm. Part of the idea was born from implementing and running cache algorithms while observing and studying their results. Also, it was helpful to study and analyze actual trace data while observing how the method runs and the results. The last contribution that led to the idea came about from studying the various replacement policies and then implementing some on a small scale. We found LRU was the most interesting to begin our algorithms investigation. We discovered how one could "stack" the trace data to eliminate the need to access disk or content manager request or trip. We found this useful in our understanding and design of our algorithm. From watching the updates to the summary algorithm (since we implemented this algorithm in our simulator and others to evaluate) and while reading trace data, there always seemed, on average, a certain "pattern" to the cache and its current trace data. This led to the idea that each client must have some type of data "signature" to its cache at any given time and over time. We started to investigate this idea, as well as reading mathematical publications [18] [19] [20] [21] [22] [23] to determine how to profile or recognize similar groups of numbers.

From there, we began with the idea that we could build on the summary algorithms timer update method to update information back to the content manager to perform a type of mathematical calculation. This would allow the client to build an up-to-date profile of its cache data at that given time. We also developed a scheme to build the profile as quickly as the data streams into the cache. We sought to build off the idea that we could use an idea similar to a heap to calculate the score in real time using the efficiency of a heap, comparable to a "score heap" (our term for it).

This idea was gleaned from [29] and from papers such as the FSR [2]. We noticed most distributed cache or networking systems deal with power, bandwidth, frequency

and hop issues. These examples tend to lead to distributed systems not hopping or going to each client in the network for data or trace requests. We built off this idea, as it was our desire to limit the number of hops and bandwidth while utilizing the profile algorithm in order to group or segment similar client cache profiles within an adjustable tolerance or threshold.

Also, borrowing from the summary algorithm idea of compact summaries idea or proxies [3], we used this to create a small compact table that uses a group id and uses the client id. Both are integers. In future work, it could be possible to make this more compact or even smaller using some sort of binary number or other ways. We left this design very flexible for that reason. We made it simple to enable us to prove our algorithms theory first.

We also expanded on the idea that over time, the data or traces seem to repeat. The intervals seem to go in cycles based on a cache life cycle. Using this data, we discovered the "accumulator" parameter which allows clients to belong to many groups or to limit the clients group membership. The theory being that if the clients were in a group at one time, the cycle will repeat for that group and become a match again, increasing the hit ratio or probability. The draw back to this idea, theoretically, in the worst case scenario, all accumulate in the same group, defeating the hop and bandwidth and power limitations. By using the "accumulator" limiter, we can force group re-evaluation and limit group membership to pairs or a small number. If the client is in a group, it forces a re-evaluation and asks, "does this client still belong in this group" and removes it if it does not meet the criteria.

Finally, for the general description of the profiler algorithm, we use all the mathematical calculations to arrive at a final score. We devised the simplest way to combine all the data in such a way to utilize it all in the final evaluation, but quickly. The score is merely a combination of all data; we basically add the total and use the absolute value. Based on the score, we compare clients against each others' score and use the threshold or tolerance to determine ultimate group membership.

Each client stores its own group data, but the content manager does all of the work by calculations and redirection based on which client is in the group. The request is made and then returns it to the requesting clients.

To build a profile, it is calculated by using six different measurements. These measurements can be improved. We used these as a starting point to prove our theory and study its results. The calculations are Euclidean distance raw and normalized. The obvious mean, median and mode. Finally, the standard deviation figure.

The Euclidean distance is used on the cache trace data within the same cache per client. We do toss out the "extra" data piece if the snap shot of the cache is odd, going on the assumption one piece will not give us inaccurate results. This warrants further study in the future. We also considered calculating the distance between two or more clients caches but felt this may overwhelm the system. It can be explored in the future. What would constitute a good sample to perform this cross distance, if not all? We treat every trace data as a point on an imaginary line to calculate our measurements. We use the same concept on the trace data for a normalized Euclidean distance. This idea emerged from [21] reading and studying this paper. We believed we could could achieve a type of second opinion on the result by using this idea.

Finally, in our profile calculations, we use the standard deviation across the cache trace numbers. We desired to use this measurement to produce an overall measure of the degree of desperation of a probability distribution. We compared how far the data or trace points are from the average of the cache traces. This calculation, we believed, was the winner or top contender that gave us an accurate profile of the data [28]. This idea, as well, could be performed across the different client's cashes. For the same reasons stated above, we did not explore this at this time.

Once the content manager calculates all of the above measurements, it then calculates the score for the clients' cache snap shot.

The content manager will then set or pass this score back to the client requiring the update. At this point, our algorithm will assign the group or groups from within the content manager. It will perform this across all clients. All of the clients are retained in a table with their respective scores that is updated as the profile is built. This data may be kept in any kind of allocated memory within the content manager, for our needs it is a passed in a thread table to the content manager. The score of the requesting client is qualified by using the threshold from the user and plus/minus as a test to determine if the score is within the range of the client at that point in the table. The content manager will also issue the next group number in increasing order. If there is a match, both clients are assigned to the same group; if not, the group number issued is removed or put back and saved for the next client request. If the accumulation is set, there is no re-evaluation if this client or clients already exist in a group assignment. If it is not set, and if the client has a group membership, it will get re-evaluated for that group and expelled if it is no loner eligible to be a member. This all occurs at the point the update timer fires per the timer setting.

Once the groups are set or the clients are clustered (grouped), the cache is able to make a request for data. It is able to do this without group membership, as there is a default. There is no penalty other than a wasted request, as there will be no match and the content manager will pass operations off to the server.

If there are legitimate group assignments, the content manager will sort through the table and find the matching client in the requesting clients' group and immediately stop.

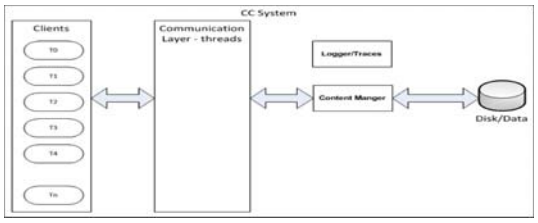


Fig. 1: Our basic CCDistSimm architecture.

It will then move to the client who possessed the match and request the data. If the data is not found, is it passed on to the server.

4. Implementation

Our implementation of our algorithm was built using java. The architecture was developed using Java SE-1.7, some on Windows 7 and other parts on Mac OS X Yosemite version 10.10.5.

The basic architecture in figure 1 called "CCDistSIMM" or Collaborative Cache Distribution System, consists of five parts, a client object; a communication layer (java threads) which is the client object; logger/tracer; content manager (brains) and disk/data server. Each client (thread) owns its own cache object. The logger/tracer content manager and disk/data server are singleton pattern objects. The logger/tracer is a singleton (built from using the java.util.logging [11]) to allow us to trace all data through one file while also doing statistical analysis on one file for convenience.

The basic architecture was set to accommodate using a shell script to run the simulator overnight without supervision to gather data. Once data or log files were collected, we used secondary scripts to parse the data. We used tools such as, grep [8] to collect statistics and run calculations to feed into another file to graph. We studied the results using Excel [9] and other programs, such as OpenOffice [10].

The overall run architecture derives from the SimmDriver object which contains the main entry point. When the main entry point commences, an array list is used to collect the command line arguments to send to our SimmIniReader object. This is responsible for handling and distributing the input parameters. Once all of the input or command line parameters are collected, the drive sets up the content manger. It is required for the clients once they start to instantiate. After the content manager is up, the server object is instantiated and its memory size is set. Its memory is filled from a pre-made file. This pre-made file may be any size. At times, this disk memory file took some time or what we call "warm-up" up time, as we had up to one million trace data, in some cases.

Once the server was set up, all clients would be instantiated (but not started, as this is important). This operation was performed using an array of clients and a basic for loop. The reason for utilizing an array was that some operations in the algorithm required knowledge or data from other clients in real time. The best way to relay all client data over to the Content Manager or "brains" of the operation was to collect all client references in an array which was then passed on to the Content Manager. We refer to this as the client table.

This array was then passed or sent to our previously set-up Content Manager in a LUT or Look Up Table method. Once all the various parts necessary for run time operations were set up, all clients were then started. This order is very crucial to prevent the clients from performing operations until all data and necessary basic architecture objects are in place.

The initial trace data was found in this area [12] [13] [14] [15] [16]. Some, if not all, was used. In some cases, this data was combined to diversify the data.

Our algorithm used the general synchronized java keyword to synchronize the thread operations. Great care was used to avoid dead lock.

The "tick" count shown in figure 2 mechanism is our way of tracing service area and a high level way of monitoring the performance of our algorithm and its implementation. This eliminates the need to have the evaluation tied to any processor speed, making future reads of our paper applicable to any processor. Tick counts are tracked in two ways in our CCDistSimm system. The first way utilizes a simple print string in the log called "TICK". This method can be problematic in that it slows down the run process/simulation as it prints the string to log file every tick. The other method utilized was an actual counter in the SimmTickCounts object. This count is incremented every tick count instead of printing a string to the log. Each client object owns its own SimmTickCounts object and presents its final result once the LRU execution is completed. This count is processed by a script to give final tick count numbers. For our convenience, there are times where we use a combination of both methods.

Our algorithm implementation or cache Profile algorithm was set up and executed within our simulator along with the Summary algorithm [3] and FSR [2]. We use an LRU, Least Recently Used replacement policy out of convenience and familiarity with the mechanism and its impressiveness. We found this mechanism able to feed itself with internal hits after the cache size (determined by user) gets larger and proportional to the size of what the data feed (from file) traces in the clients are filled or fed with. As an example, if the file feeding the client has 1000 traces, this number is usually 500 for cache size.

This algorithms main objects are SimmProfileEngine, SimmProfileGroupInfo. The different portions of code are executed based on SimmAlgorithm settings. The first one being SimmProfileEngine which contains the engine behind all the math utilized to build the calculations for the profile generation score. We set this up as a separate object for future modification, as we were ultimately unsure which mathematical operations would best build up our profile. In this way, we add or subtract methods for calculations.

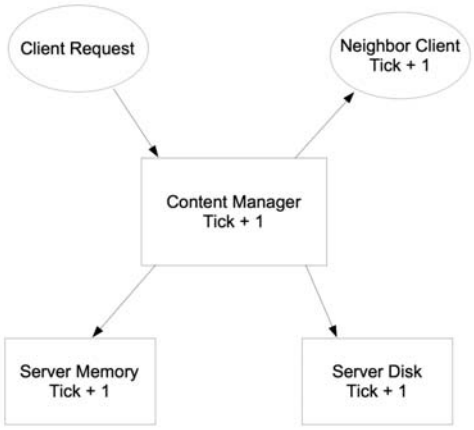


Fig. 2: The basic Tick counter method[2]

The methods in this object were determined by some research and some obvious measurements using this small sample of site [18] [19] [20] [21] [22] [23]. The methods used are rawEuclidean(), normailizedEuclidean(), cacheAverage(), findDeviation(), calculateMedian() and calculateMode(). The three main methods of significance being rawEuclidean(), normailizedEuclidean() and findDeviation(). We believe the remainder are obvious to the reader why they were picked, but we will briefly touch on it in subsequent sections.

These calculations were chosen as a starting point and can be improved. We drew upon our research and previous algorithm knowledge for distance between numbers.

The first was Euclidean distance [24] shown. We calculate this to be over every two points in our cache, the local cache. We attempted to calculate this over one or more caches between clients, but ran out of time. For odd numbered cache trace data, we discard the non-matched trace. We desired to baseline, therefore, we did not worry about the accuracy hit we would take by discarding one.

Next, we attempted to calculate the normalized Euclidean distance. Some of our references above mentioned the way it can be more accurate than the raw Euclidean distance. It helps scale the variance and quantifies the distance. The raw measure has no bound value for maximum distance. Raw works best under relative ordering for a fixed set of attributes. This also discards an odd valued cache. The same reasoning applies as before. We merely desired to bench mark and prove our theory before we were more precise with our handling of even and odd data.

The third calculation worth mentioning was to discover the deviation. This was used to measure the spread across the cache "snap shot" that was given to the content manger by the timer object. We think of it as taking it across the cache trace population to attempt to summarize the continuous cache data. We were not sure how this would react with skewed data or a number of outliers in the cache. Some preliminary data analysis showed this not to be the case, on average.

The SimmProfile object contains the results of the SimmProfileEngine() calculations. It also holds the groupId. Every Client object has a profile object for use by the Content Manager. The total calculations in this object are raw Euclidean,normalized Euclidean, average, deviation, median, myScore, mode and groupID. This object has defaults set for beginning threads which proceed first (setters and getters). It serves as a type of a holding place to hand off to other methods that require the numbers for calculation in the content manager, as well as for final group matching and group data look up.

The majority of the work is completed in the content manager; again, what we named the "brains" of this operation or simulator. The trigger point is the timer, where, as we mentioned before, used the SimmSummaryUpdateTimer() object. It runs the buildProfile() method in the Content Manager when the timer goes off, taking a "snap shot" of the cache, that point, it calculates all profile fields while also calculating the score and group matching. There also exists a parameter we call accumulation that allows clients to belong to one or more groups. If this is false, the group membership is evaluated each time the group membership is calculated.

The timer can be unpredictable, as it is a thread. It runs when there is neither enough clients nor any data traces in the client's cache. We therefore have built in some mechanisms to reject the profile build request by the timer object to "tell" it to comeback when the cache reaches a minimum size. This size is usually two. Not enough clients being up is not as important as the buffer size, as we find this in what we refer to as "warm up" in the data analysis.

The client object instantiates the buildProfile() object and the SimmProfileEngine() object every entry.Once all the calculations are made, the score is tabulated by simply adding the total, score = getRawEuclidean() + getNormailizedEuclidean() + getAverage() + getDeviation() + getMedian() + getMode(). We left this number in its raw double form.

The next method to execute was the assignGroup(), which keeps track of all the groups at the place it left off and removes it if re-evaluated, based on the accumulation parameter. The tolerances are checked at this point to determine group matching on the scores. The range is based upon the user tolerance input, and it is a plus/minus range. Attention must be directed to the raw initialized score (then move on) assigning it its own id when determining the group membership. If both match, a group id is not required. The next group id in line is assigned if one is needed. If one is assigned to a group, the other is added to its group. The group number is rescinded when there is no match, as it gets

Group	Group ID
1	Client-2
1	Client-1
1	Client-3
2	Client-77
2	Client-777
3	Client-888
3	Client-999
3	Client-7

Fig. 3: The basic Multiple Group Accumulation Assignment table (virtual) [24]

Group	Group ID
1	Client-2
1	Client-1
2	Client-3
2	Client-4
3	Client-777
3	Client-888
4	Client-999
5	Client-7

Fig. 4: The basic Single No Accumulation Group Assignment table (virtual) [24]

pre-incremented. If no match is found and this client is in a group, it will be reevaluated if there is no accumulation (set to false).

It should be noted, whenever any client requires a request from the content manager and it has to sweep the allclients[] array, it must always exclude itself to avoid mis-reading.

Once the cache is full in the client object, we simulate a request and in the profile algorithm the getDataFromClientsInMyGroup() method is called from the content manager. It confirms the requesting client is assigned to a group, and if so, looks to find a match among the clients. It also confirms the group does not include itself and is not the default group. It will proceed through all groups or the other clients in the group depending on settings and search for the requested data and return it, if found. Otherwise, it proceeds to the server for a memory search or disk.

The group assignment table is shown in figure 3 below with exaggerated id. In reality, they are integers, for space reasons. The table is virtual, in other words, it does not truly exist as a table in the code. It is merely a concept to illustrate the way the group to client relationship architecture is constructed. Within the code design, the table is just a for loop that performs group id comparisons to find matches.

The single group assignment table figure 4 is shown below, again the id, is an exaggeration.

The overall group algorithm request is shown in the figure 5 below.

The last section of our implementation is the server or

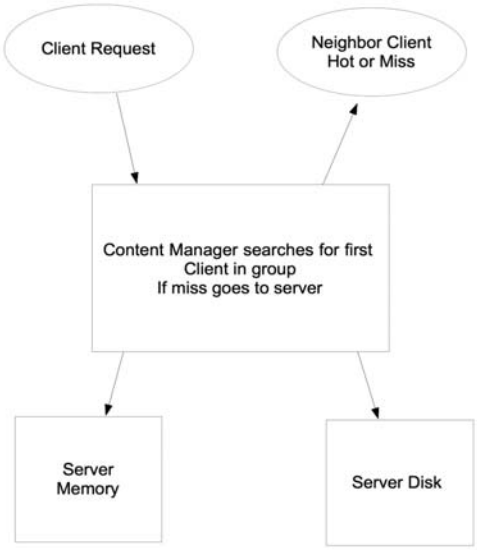


Fig. 5: The basic Group Request (virtual) [24]

SimmServer object. This is a singleton that simulates a real world server. Its file assignment is kept in the SimmConstants object. When the simulator starts up, its file is loaded into its memory store using its fillMemm() method called in the constructor. This has two searchMemm() methods; one is over loaded to handle a client object parameter in order to run the second tick count method. It searches its memory for a memory hit, or else merely logs a disk hit.

5. Performance/Results

Our approach to the performance and evaluation of our Profile algorithm developed in our distributed collaborative cache simulation was to predict the results that were predetermined during our research phase. It is summed up by using predetermined expected graphs.

Predictions were based upon our initial research of what we would expect to find under the conditions graphed. We researched approximately 20 papers in the literature on caching, adaptive caching and distributed collaborative caching to arrive at our predictions. The predictions were not necessarily correct, merely what we would expect to find in our distributed system.

To start, we gathered data with small hand-made data sets. Our reasoning was to obtain a "feel" or to discover any patterns on a smaller level, rather than large. We were able to run any number of clients within system resource confinements. We also used small hand made files to verify our theory for group assignments and group profiles. We used files with all the same traces (integers). We did every other trace as the same, or completely different, to ensure we would obtain the results we expected. If the results were something other than what we expected, would would then have to explain it. This enabled us to establish a baseline to determine if the larger data should run.

We also tested different range traces within the same session, as an example, 12345 and 1234567. As you will note, the range of these numbers is large. This large range made a difference when it came to the profile data. It would skew the group scores and at times, if the data skew was extreme, it did not find any matches. This caused the tolerance or threshold to be adjusted, accordingly. It also verified that caching works best when all devices are processing something similar, such as different devices viewing the same movie.

We also studied the results of utilizing homogeneous trace data verses data that was the same or similar.

The graph shown in figure 7, was one of our first evaluations involving the affect of the accumulation parameter on the algorithms performance. With the accumulation parameter turned off (disabled), group membership reevaluates each profile execution. As can be observed in the graph in figure 6 the line is "jagged" and increasing. The increase, which is the increase in hits, is caused as the number of clients is increased, as are the number of hits. This behavior

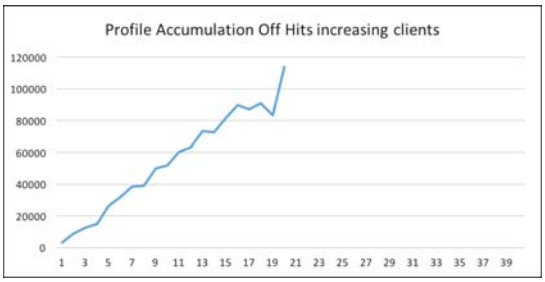


Fig. 6: Profiler algorithm Hit comparisons, increasing Clients with no accumulation

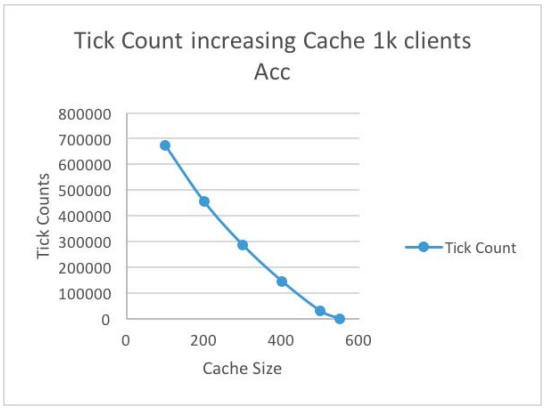


Fig. 7: Profiler algorithm Tick comparisons, increasing Clients Cache with accumulation. 1k Client count

is expected. Also, around 20 clients the affect of "warm up" can be observed with the sharp "dip" in the line. The "jagged" observation is a result of the group reevaluation. This "jagged" look is expected, but we must admit, we were a bit surprised until we examined the data in order to discover the reason for this behavior. The reason the "jaggedness" occurs is because as group membership comes and goes, so does its hit ratio. In other words, as the client enters a group, it has a certain hit count. As it is expelled and/or joins another group (in and out of groups), its hit ratio changes again, as seen in the jagged line. This "jagged" phenomena can be observed in other evaluations, as well, whenever the accumulation parameter is turned off.

The graph shown in figure 7 displays the tick counts for the Profiler algorithm. The settings for this run are 1k clients in the system, 1k trace data per client, with group accumulation enabled. The results of the graph show, as the cache size increases, the tick count decreases. Note, that since the group accumulation is enabled, there is no "jaggedness" to the graph trace. This result is as expected.

The graph in figure 8, evidences the miss rate for the profile algorithm. The settings for this run are group accumulation enabled, 1k clients in the system and 1k trace data. This run had a high threshold, as well. This high threshold would allow "easy" or a wide range for group

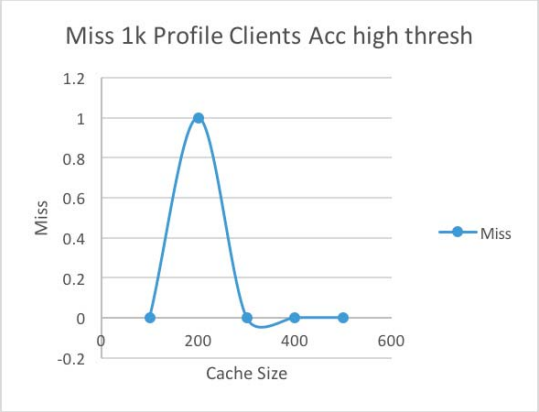


Fig. 8: Profiler algorithm Miss comparisons, Mid Sized Cache with accumulation and high threshold. 1k Client count

membership. The results in this graph show, as the cache size increases, there was one miss! We attribute this one miss to the "warm up". This was not expected behavior. We were quite surprised that the Profiler algorithm worked so well. We were concerned that this may have hit the worst case scenario where every client is a member of the same group, nullifying the efficiency of hop count, battery power etc. We surmise this not to be the case, as the testing finished in under 15 minutes. If every client was in the same group it would have taken hours to complete from previous observations. Without further study, we cannot say this is a fact. Note the small "dip" in the graph, it looks to be a negative number, but it is not. This is the same phenomena observed with excel as before [9].

The graph shown in figure 8 shows the percentage of hits for the Profile algorithm with increasing cache size. The run settings are accumulation disabled, 1k clients in the system and a wide threshold as above. This graph shows the "jagged" phenomena. This occurs due to the clients entering and leaving groups, changing the hit ratio as the group membership changes. This behavior is as expected.

The graphs shown in figures 10 and 11 show the Profiler algorithm with accumulation disabled, 200 clients in the system and increasing cache size for miss and hit rates. Both graphs show as cache size increases the miss and hit rate decrease. This behavior is as expected due to self service from the increased cache size. In graph 10 the "jaggedness" can be slightly observed. This run only had 200 clients. As can be observed, it is less accurate than the run with 1k clients. This is a result of possessing less clients with which to collaborate.

The next two graphs shown in figures 12 and 13, show the Profile algorithm for Hit and Miss rates. The run settings are 1k clients in the system, group accumulation enabled, average threshold and increasing cache size. Observe that the graph in figure 12, has two misses! This was not expected. The performance of the Profiler algorithm exceeded our

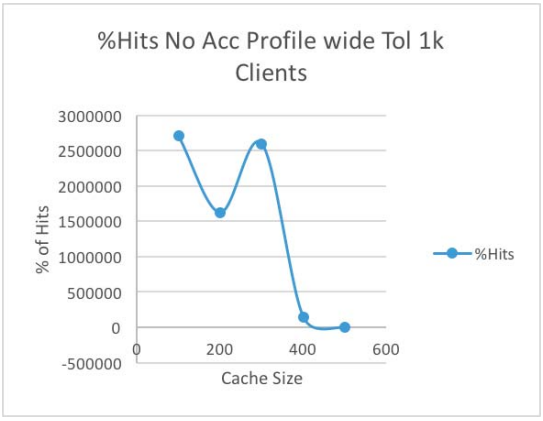


Fig. 9: Profiler algorithm %hits comparisons, increasing Clients Cache with no accumulation and wide threshold. 1k Client count

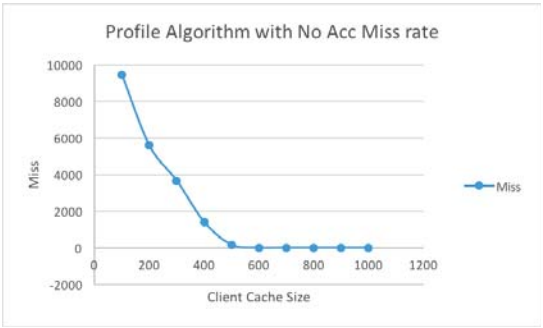


Fig. 10: Profiler algorithm Miss comparisons, increasing Clients Cache with no accumulation and ave threshold. 200 Client count

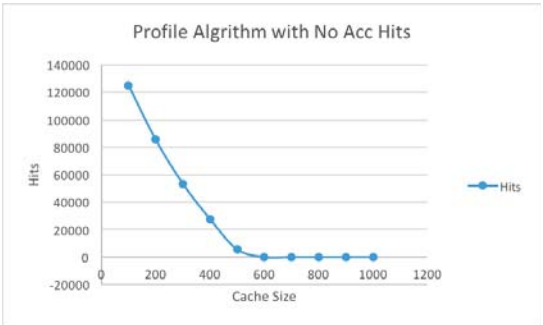


Fig. 11: Profiler algorithm Hits comparisons, increasing Clients Cache with no accumulation and ave threshold. 1k Client count

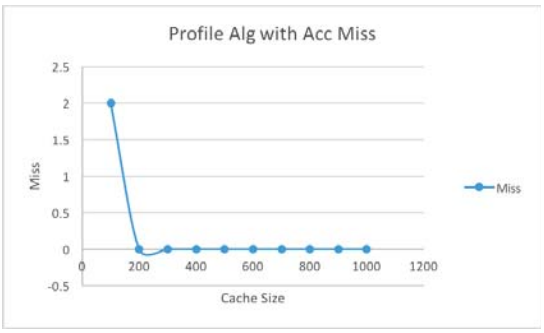


Fig. 12: Profiler algorithm Miss comparisons, increasing Clients Cache with accumulation and ave threshold. 1k Client count

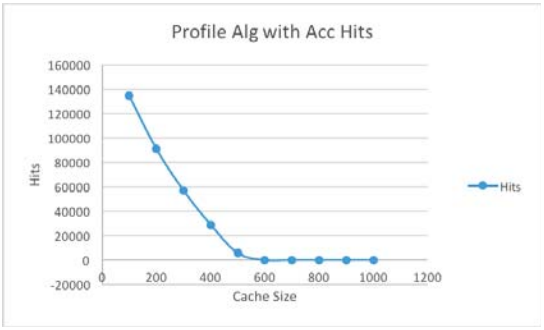


Fig. 13: Profiler algorithm Hits comparisons, increasing Clients Cache with accumulation and ave threshold. 1k Client count

expectations. Note that the performance improves greatly with 1k clients, as compared to 200 or smaller.

The graph in figure 12 is shown for Miss rates with increasing clients. This graph illustrates the "jaggedness" phenomena that is very exaggerated. This behavior is as expected.

It should be noted that once our performance study for the Profile algorithm reached the point of 1k clients, its performance was so outstanding that we could not accurately display the percentage numbers. We observed miss rates of just one with group accumulation and without group accumulation. We saw numbers for misses fall from 249 to 28 to 11 to 1 ultimately to 0 within the same run. This data was also averaged over a handful of runs (20). This justifies further study and could be possible publication in the future.

Figure 15 shows the tick counts for the Profile algorithm with accumulation disabled. It can be observed that as the cache size increases the tick count decreases. Once the cache is large enough, it can service itself. This behavior is as expected.

Fig. 14: Profiler algorithm %Miss comparisons, increasing Clients with no accumulation and ave threshold. 1k Client count

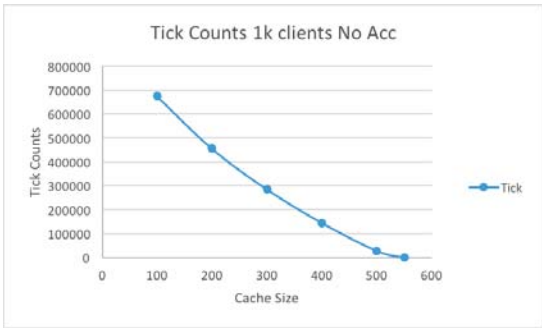


Fig. 15: Profiler algorithm Tick comparisons, increasing Cache with no accumulation and ave threshold. 1k Client count

6. CONCLUSIONS

Our conclusions are based upon our graph and observation data. We have proposed our Profile algorithm. The results of our CCDistSimm, while it needs some improvements, has been quite productive. With our profiler algorithm, the more groups that exist, the more accurate the system is on hit. There is also less server access.

The memory layout as set forth by a file determines disk access. If we randomly sample client cache files to create memory, we obtain real world results. We found we can skew it, or make it so that we never get any memory hits, just disk. We did observe there is a science to the data that should be placed in server memory, as opposed to what is retained on disk. This is something that is worth future research.

With our profile algorithm, we discovered theories that some optimization could be performed by not using a timer but by calculating how many fetches are made. In other words, there could be a "fetch" profile created to determine the update of the cache profile. This could be utilized in conjunction with a timer, and/or our idea mentioned in previous sections of a "score" heap.

In future work, the number of groups should be studied. To observe that if by having the accumulation feature on, would the worst case scenario ever happen where every client

belongs in each others group. We could research to find a better mathematical way to build the profile (if needed).

The LRU or Least Recently Used replacement policy was observed to possess a threshold for cache size and internal hits that is proportional to the size of the incoming data. One average was around 1/2 the data file size. We do understand that in reality, this incoming data is not a file and is to be considered infinite. We feel this observation is worth noting.

Our Profile algorithm shows a lot of promise. The results far exceeded our expectations for speed and accuracy. One such expectation was that of minimizing the cost to server memory and disk hits. Our Profile algorithms main ideas should be studied and improved in future work. These results show the Profile algorithm warrants a test bed and/or real world scenario testing and analysis.

References

[1] Simon Fear. (2005). Publication quality tables in LaTeX. [Online]. Available: http://www.ctan.org/tex-archive/macros/latex/contrib/booktabs/booktabs.pdf
[2] Fan Ye, Qing Li and EnHong, "Adaptive Caching with Heterogeneous Devices in Mobile Peer to Peer Network", Fortaleza, Ceara, Brazil, 2008.
[3] Li Fan, Pei Cao, Jussara Almedia and Andeida Z. Broder, "Summary Cache: A Scalable Wide-Area Web Cache Sharing Protocol,IEEE/ACM Transactions on Networking, Vol 8, No, 3, June 2000.
[4] Mohan Kankanhalli, Kalpathi Ramakrishnan, Shantanu Paknikar, "A Caching and Streaming Framework for Multimedia", Research Gate, January 2000.
[5] Atul Rao, Prashant Kumar and Naveen Chauhan, "Cooperative Caching Strategies for MANETs and IMANETs", Department of Computer Science and Engineering, National Instituite of Technology, Hamirpur, India.
[6] https://www.netflix.com.
[7] https://www.ietf.org/rfc/rfc2326.txt
[8] https://en.wikipedia.org/wiki/Grep
[9] https://office.live.com/start/Excel.aspx
[10] https://www.openoffice.org/download/index.html
[11] https://docs.oracle.com/javase/7/docs/api/java/util/logging/package-summary.html
[12] http://stackoverflow.com/questions/7920571/block-level-i-o-trace
[13] http://opera.ucsd.edu/Projects/BufferCache/trace.htm
[14] http://iotta.snia.org/tracetypes/3
[15] http://www.cs.utexas.edu/users/mckinley/352/homework/project.html
[16] http://users.ece.cmu.edu/ jhoe/distribution/2005/741/proj1.pdf
[17] https://en.wikipedia.org/wiki/Bloom_filter
[18] http://commons.apache.org/proper/commons-math/userguide/stat.html
[19] http://introcs.cs.princeton.edu/java/stdlib/StdStats.java.html
[20] http://www.cs.nyu.edu/courses/fall09/V22.0002-002/programs/programs17-0002/Deviation.html
[21] www.pbarrett.nettechpaperseculid.pdf
[22] http://pages.cs.wisc.edu/čs302-5/MeanMedianMode_Methods.java
[23] http://www.cs.nyu.edu/courses/fall09/V22.0002-002/programs/programs17-0002/Deviation.html
[24] http://cs.carleton.edu/cs_comps/0910/netflixprize/final_results/knn/index.html
[25] http://spreadsheetsolving.com/2013/06/15/sample-standard-deviation/
[26] https://en.wikipedia.org/wiki/Internet_Cache_Protocol
[27] http://www.codeproject.com/Articles/29224/Six-Sigma-Interview-Questions
[28] http://www.separatinghyperplanes.com/2014/04/why-do-statisticians-use-standard.html
[29] Algorithm Design, John Kleinberg and Eva Tardos
[30] S. Paknikar, M Kankanhalli, K.R. Ramakrishnan, S.H. Srinivasan, L.H. Ngoh, "A Caching and Streaming Framework for Multimedia", ACM Multimedia 2000, Los Angeles, USA.

Round Robin Data Bases for Performance Evaluation of High Performance Applications and Clusters

Fernando G. Tinetti[1], **Leopoldo J. Rios**[2]
[1]Fac. de Informática, UNLP, Comisión de Inv. Científicas Prov. Bs. As., Argentina
[2]Instituto IMIT, CONICET-UNNE, Corrientes, Argentina

Abstract – *We introduce a tool for automating (or aiding) performance evaluation of HPC (High Performance Computing) applications by combining both, Round Robin Databases (also referred to as RRD) and performance data collected at runtime. RRD monitoring is at the base of several well-known and popular tools for cluster monitoring. We use take advantage of already existing RRD tools for collecting, processing, and presenting runtime data for scientific processing users. Scientific applications (and even the hardware used by those scientific applications) are assumed to be performance optimized, but it is not always the case. Thus, collecting and analyzing runtime information will help scientific users to decide new optimizations and/or runtime strategies. The primary focus will be on parallel applications running in clusters, i.e. distributed hardware, since they are the most complex to optimize given their different and varying computing and communications patterns.*

Keywords: Performance Evaluation, Round Robin Databases, High Performance Computing.

1 Introduction

Performance optimization and tuning has been done since the beginning of computer science up to these days [1] [2] [3] [4] [5]. Classical tools such as the well-known profilers are used in the first steps of optimization, but they not always provide enough and/or accurate information beyond plain (and statistical) running time/s. Currently, there are multiple types of HPC (High Performance Computing) applications running on production clusters as well as there are multiple possibilities or sources of monitoring information. At the lowest level is found the most accurate information provided by hardware event counters [6] [7]. At the highest level is the (mostly statistical data) collected by operating system (OS) for accounting (such as that found at Linux or OS X /proc).

Beyond abstraction levels, there are multiple ways of collecting data for analysis as well as there is no single place/tool to have every single piece of data needed for system and/or application performance analysis. Some tools are focused are "system wide" in that summarize or provide information regardless specific users and applications, such as Ganglia [8] [9]. Tools such as profilers are specifically designed (at least initially) for single application performance optimization and post-mortem data analysis [10] [11]. Most profilers are not meant for real-time monitoring and are focused on applications, not HPC systems such as clusters (either in hardware or used in a cloud facility). At most, several profilers are single system-wide, and there it should be necessary to aggregate several data sources to have a complete view of a parallel processing application, for example.

Multiple abstraction levels, tools, sources, and types of data usually lead to a complex scenario for HPC application programmers as well as those in charge of medium/long term decisions for HPC systems. This work is a first attempt to provide a single tool for the scientific related to HPC systems, so that it can be adjusted/configured to collect and visualize different data in a uniform way. The huge amount of data involved in performance monitoring will be handled in RRD taking advantage of a priori storage limits as well as the large amount and usefulness of tools for handling those data [12]. Actually, is can be considered that parallel application resource usage/monitoring is more complex than whole cluster monitoring, because a parallel application does not necessarily uses all the resources at the same time.

2 Basic Information

Overcoming the basic problem of real-time profiling (as opposed to the post-mortem data provided by profilers is relatively simple combining three tools/tasks:

1. Find and extract information in real-time (at runtime) from /proc filesystem.
2. Store data in a RRD.
3. Visualize/generate graphic charts and/or define further analysis.

Even when /proc information is system-wide, it is not rare that nodes in the cluster are "reserved" or "exclusively used" for a single process (via some resource manager), thus in that case the /proc (mostly) contains data of a single process. Beyond sampling, ell data is handled by RRDtool. Furthermore, data stored in RRD can be used for more specific analysis beyond graphic charts, depending on users' requirements.

Data extraction from /proc filesystem is relatively easy using standard tools such as grep/awk. Periodic samplig is also simple using OS tools such as at/cron. The most complex data management is data storage and handling, which is made in this proposal by RRDtool. The key of the integration between sampling and RRDtool usage is matching sampling period in order to avoid noise generated by "extra" interpolation. It is not reasonable to sample with a 4 hours rate if the analysis is going to be made in terms of minutes. Given that users have the application knowledge, the final tool will be configurable in terms of data collected and sample rate.

RRDtool provides a lot of facilities for handling sample data in general and those collected by monitoring HPC parallel/ distributed applications. Fixed size data storage and consolidation functions are specifically designed for time series data. Our proposed tool is intended to go beyond standard tools such as Ganglia in that it will show and consolidate information by users or by applications at several extra levels of details. At this point the tools already have implemented data collection by users in order to aid decision making to users and/or a HPC center manager/s. It is worth pointing out that even this basic information is not directly available using standard tools such as Ganglia, which require at least specific customization for handling and generating reports of this kind of data. Up to this point, the greatest contribution would be the tuned usage of well-known tools for clusters and applications analysis. The tuning is specifically related to two key aspects: 1) integration with the runtime system, so that measurements are made by application/user, and 2) user parametrization (as opposed to fixed by design) of data to obtain and/or selection of sampling data.

As a proof of concept, a synthetic application was monitored with our tool. The application is focused in memory: it requests and set to a fixed value 10MB blocks every 5 seconds. The total amount of RAM is 16GB, and the values selected for monitorization were: a) total memory, b) available memory, and c) memory used for cache (memcache in /proc filesystem). Fig. 1 shows the memory evolution that our tool is able to show from beginning to end of runtime. The units are KB, so 10M represents 10GB RAM (the total amount of RAM is 16GB). The application does not have any I/O, so memcache evolution is related only to OS decisions. At about 1/3 of the runtime, there is almost no free memory, but it is due to OS allocation/usage of memcache. The chart graph in Fig. 1 provides information for the user which lets to conclude that at that point in runtime if there is a problem for memory, it is not related to its application but related to OS memcache management. As expected, the application is able to keep running despite there is not "clean" free memory, since the OS is able to release memory from memcache depending of its I/O management. It is left to the user to analyze if its application performance is affected by the time there appears to be no free memory in spite of the fact it is not using all the memory and it is not doing any I/O. It would be possible to relate memcache usage to application runtime if other process activities are monitored, e.g. I/O operations.

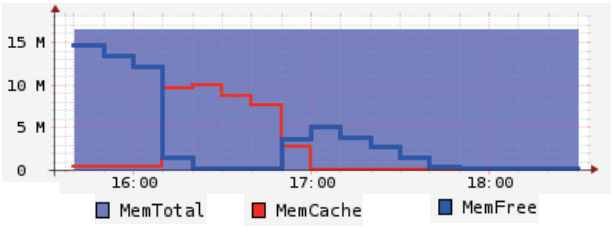


Figure 1: Monitoring Memory

Fig. 2 shows the last minutes of process execution, where the free memory decreases as the used by the process increases, coming to the point that there is more memory used by the process than free memory in the system. Also, Fig. 2 shows that we are able to identify per process memory assignment beyond the "system wide" information (which would be "free").

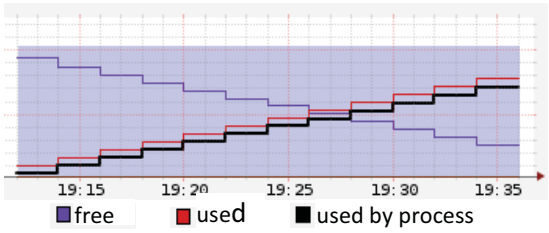


Figure 2: Free and Process Memory Evolution.

Fig. 3 shows another view of the memory usage evolution as well as how the free memory begins increasing after the process has ended.

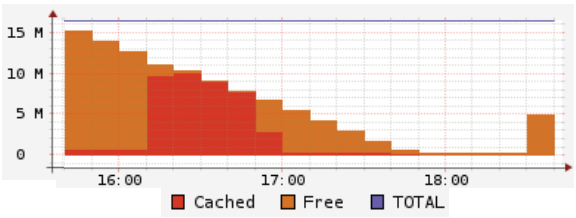


Figure 3: Memory Evolution.

The tool is now being enhanced to collect more specific data at runtime, just like that provided by hardware monitoring counters now available in standard processors [6] [7]. It is worth noting that information as that in Fig. 1 is not directly available in standard profilers or monitoring tools, beyond that it is expected to have even better/more specific data as the tools is enhanced. Also, summarizing this information in a per cluster base is straightforward, since the data is already available with the tool in its current state.

3 Parallel Application Data

Currently, Ganglia [8] is widely used for cluster monitoring and usage visualization in the HPC field. Classical computational resources involved in HPC are easily identified

in a cluster as well as in individual nodes. Once installed, Ganglia begins to collect data on each of the Grid / Cluster nodes, and data are stored in files RRD. Ganglia collects data referred to CPU and memory usage and network interconnection traffic. Large scale (in number of nodes) reports are possible and reports can be customized and sorted by date, for example. However, there are many interesting/ useful metrics missing, or for which there is a huge tuning work for them to be collected. Most of the missing information is related to identifying users and applications. Examples of missing information are (parallel) program using specific cores (or some fraction of cores) and the owners (users) of processes at runtime in a given time frame. As in the example given in the previous section, we are integrating different sources of monitoring data in order to provide better/more useful information than the currently available with standard tools. Also, given that the methodology (e.g. sampling, time series data) and basic tools (e.g. RRDtools) are maintained there is no overhead beyond that of current and *de facto standard* tools, thus maintaining quality of service in general.

Cluster/distributed resource managers such as Torque (aka Torque PBS –Portable Batch System) [13] are a very good source of valuable information for HPC users as well as cluster managers. Resource managers usually are a suite of applications that help to manage and organize the jobs in a cluster and/or grid. Users are able to interact in a rational and organized way with the computational resources, avoiding downtime, for example. Job submission, monitoring, and visualization are performed via simple commands. Most resource managers (including the aforementioned Torque) let users to specify required resources such as nodes, cores per node to be used (or processes to be started at each node), memory, and other specific details such as command name and (usually command line) parameters to execute a job. Statistical data provided by the resource managers are very useful. Most (if not all) current monitoring tools do not integrate such information, and we think it has to be integrated to those provided by the operating system, mostly because they refer to particular (parallel) jobs, directly related to specific applications and users.

Our monitoring tool currently integrates data provided by Torque as a proof of concept: we are able to take advantage data provided by resource managers. Thus, the information provided to users and system managers is enhanced, combining data provided by resource managers to that provided by the OS accounting system. Actually, the integration is made at the level of data stored and handled in RRD and by RRDtools, i.e. as time data series. As a first step, our tool provides further information beyond that provided by standard monitoring tools about CPU/cores usage. Specifically focused on parallel application performance monitoring, it is important to identify the user/s running processes on our cluster and the number of cores actually being used. Standard tools usually provide the % of cores and or nodes being used. Fig. 4 shows the evolution in time of cores utilization per user in a cluster.

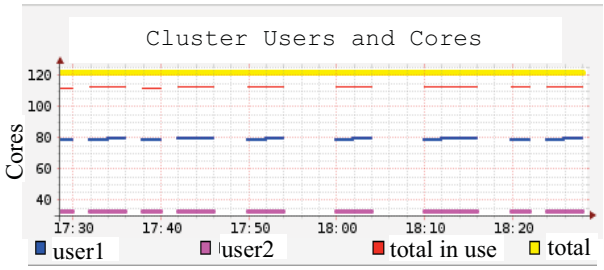


Figure 4: Cluster Cores Usage.

Fig. 4 shows/confirms that user1 is most likely running a parallel application (classical SPMD processing with a fixed number of cores) while user2 is running a sequential job. The parallel application can be definitely confirmed using data from the specific cluster resource manager (queue manager) being used to start cluster processing jobs. Fig. 4 also shows the total number of cores being used as well as the total number of available cores, as standard in cluster monitoring tools.

The monitoring tool is currently being enhanced by including core as well cluster optimization/utilization data maintaining their uniform handling in RRD-RRDtools. One the first data we are integrating is that provided by hardware performance counters. Initially, we will take advantage of tools such as perf [14] which, in turn, takes advantage of the perf_event Linux kernel interface [15]. Even when perf is very attractive since it does not require any source code change, it implies lack of runtime profiling, since it provides post-mortem data. Hardware performance event counter data can be collected, though, if the code is instrumented with libraries such as PAPI (performance API) [16] [17]. Thus, a runtime profile can be built with low level and precise hardware behavior if the code is specifically instrumented.

4 Conclusions and Further Work

We have been able to integrate multiple sources of data in order to monitor performance behavior and provide information for HPC applications and clusters. Collected data is handled via RRD (Round Robin Databases) so it is manageable in terms of bounded data storage. RRDtools provide highly useful ways of obtaining statistical and graphical data for programmers as well as HPC cluster managers. We have shown how our tool takes advantage of information provided by OS as well as cluster/queue resource managers in order to provide insightful data for HPC optimization and hardware resource planning. Even more, we are able to tune the kind of information specifically useful for parallel programmers having running applications in clusters.

We are working on several enhancements for our tool:

1. Adding user interface capabilities so that a user will be able to easily select among a set of data to be collected, date ranges, and data visualization way/s

2. Integrate post-mortem hardware event counters data provided by tools such as perf. Even when the way in which hardware data is integrated in RRD is straightforward, hardware dependence (i.e. specific events each micro-architecture has implemented) involves another level of parametrization the tool does not currently implement.

3. Integrate profiling with runtime hardware event counters data provided by code instrumentation and library such as PAPI. This is another level of parametrization, since it implies collection of data at runtime and reconstruction of program behavior (profile) from collected data.

5 References

[1] D. H. Bailey, R. F. Lucas, S. Williams, Eds., Performance Tuning of Scientific Applications, CRC Press, 2011.

[2] J. Hennessy, D. Patterson, Computer Architecture: A Quantitative Approach, 5 Ed., Morgan Kaufmann Publishers, Inc. 2012.

[3] Fernando G. Tinetti, Andres More, "Hotspot: a Framework to Support Performance Optimization on Multiprocessors", Proc. PDPTA'15, H. R. Arabnia, H. Ishii, K. Joe, H. Nishikawa, H. Shouno, L. D'Alotto, G. A. Gravvanis, G. Jandieri, G. Sirakoulis, A. M. G. Solo, W. Spataro, F. G. Tinetti, G. A. Trunfio, CSREA Press, 2015, pp. 171-176.

[4] A. V. Aho, M. S. Lam, R. Sethi, and J. Ullman, Compilers: Principles, Techniques, and Tools, 2nd ed. Prentice Hall, 2006.

[5] J. A. Bilmes, K. Asanovic, C. Chin, and J. Demmel, "Optimizing matrix multiply using phipac: a portable, high-performance, ansi c coding methodology," Proc. of the International Conference on Supercomputing, A. SIGARC, Ed., Vienna Austria, 1997.

[6] Intel 64 and IA-32 Architectures Software Developer's Manual, Combined Volumes: 1, 2A, 2B, 2C, 3A, 3B, 3C and 3D, Dec. 2015.

[7] S. Browne, J. Dongarra, N. Garner, G. Ho, and P. Mucci, "A portable programming interface for performance evaluation on modern processors," International Journal of High Performance Computing Applications, vol. 14, no. 3, pp. 189–204, 2000.

[8] M. Massie, B. Li, B. Nicholes, V. Vuksan, R. Alexander, J. Buchbinder, F. Costa, A. Dean, D. Josephsen, P. Phaal, D. Pocock, Monitoring with Ganglia. Tracking Dynamic Host and Application Metrics at Scale, O'Reilly Media, 2012.

[9] Ganglia Monitoring System, http://ganglia.info/

[10] M. Honeyford, "Speed your code with the GNU profiler. Target the parts of your applications that take the most time", April 2006, IBM DeveloperWorks, Technical library GNU Gprof documentation.

[11] W. E. Cohen, Tuning Programs with OProfile, Wide Open Magazine, 2004, pages 53–62.

[12] http://oss.oetiker.ch/rrdtool/

[13] Adaptive Computing, Torque Resource Manager http://www.adaptivecomputing.com/products/open-source/torque

[14] perf wiki, https://perf.wiki.kernel.org, Linux profiling with performance counters.

[15] V.M. Weaver. "Self-monitoring Overhead of the Linux perf event Performance Counter Interface", IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS 2015), Philadelphia, Pennsylvania, March 2015.

[16] V. Weaver, D. Terpstra, S. Moore, "Non-Determinism and Overcount on Modern Hardware Performance Counter Implementations," 2013 IEEE International Symposium on Performance Analysis of Systems and Software, Austin, TX, April 21-23, 2013

[17] F. G. Tinetti, M. Méndez, "An Automated Approach to Hardware Performance Monitoring Counters", CSCI The 2014 International Conference on Computational Science and Computational Intelligence (CSCI'14) March 10-13, 2014, Las Vegas, USA,

Configuration and Administration of a Cray CS 400 Heterogeneous Cluster with Bright Cluster Manager

Omar A. Morris
Dept. of Electrical & Computer Engineering
Jackson State University
Jackson, MS. 39217 USA
omar.morris@jsums.edu

Khalid H. Abed
Dept. of Electrical & Computer Engineering
Jackson State University
Jackson, MS. 39217 USA
khalid.h.abed@jsums.edu

Abstract—*This paper presents configuring and administering a Cray CS 400 heterogeneous cluster using Bright Cluster Management software. It details the architecture of the computational environment and the steps taken from initial assembly of the hardware to the installation of the various software layers that comprise the OS and system management tools. The Bright Cluster Manager is an enterprise level software suite that provides bare metal to fully functioning system management of High Performance Computing clusters and Big Data systems.*

1 INTRODUCTION

As our body of knowledge has increased so has our realization that there are ever more discoveries to be made. The digital tools used to make these discoveries and expand our understanding have evolved from analog to digital and are becoming increasingly complex. The growing demands on these computational systems are driving the traditional CPU to its physical limits. To address this problem, which is the interconnected relationship between the need for more powerful hardware and the software with the ability to exploit it to handle the growing complexity and vast amounts of data in modern applications and computational models, High Performance Computing (HPC) has emerged. HPC is the aggregation of computational elements in parallel to implement large-scale mathematically intensive applications in the areas of engineering, science, mathematics and business It is geared toward modeling complex physical phenomena such as climate change models, the 3-D mapping of proteins, military applications, and academic research.

The industry has turned to the use of accelerators or coprocessing units to increase the power and efficiency of standalone traditional CPUs. Intel developed a coprocessor capable of executing highly parallel numerically intensive applications quickly and efficiently; the massively parallel Xeon Phi Many Integrated Core (MIC), based on the x86 Intel P5 processors. The Xeon Phi environment utilizes all standard programing platforms and paradigms such as OpenMP, POSIX threads and MPI as well as high level languages like C++ and Fortran [1]. This allows the developer to work with familiar tools sets and languages and does not require massive rewrites of exiting code. That is not to say that achieving the desired speed up of application execution does not require effort and specialized knowledge. The developer must understand the underlying architecture of the Xeon Phi coprocessor to exploit its strengths and to determine if the platform is the correct one to port their code onto.

2 Environment

The system used in this research is a Cray CS 400 comprised of a single head node and six compute nodes. This section gives a description of the hardware specifications of the two types of nodes and the cluster's network topology.

2.1 The Head Node

The head node has an Intel Xeon E5-2650 v3 CPU, a Connect-IB (InfiniBand) Single Port QSFP, Fourteen Data Rate (FDR) adapter card and a ConnectX-3 EN10GbE Dual-Port SFP+ PCIe3.0 x8 8GT/s Network Interface Card. It also has an Intel RS2BL08D 8-port SAS RAID controller.

2.2 The Compute Nodes

Each compute node, named node001 – node006, has the Intel Xeon E5-2698 2.3GHz and dual Intel Xeon Phi 7120 passively cooled 1.25 GHz coprocessors. The Xeon Phi 7120 has 61 in-order 64-bit cores able to execute two instructions per cycle with 16GB of GDDR memory. The cores are fully functional computational units which allow the coprocessor to run its own Linux based operating system called uOS.

Each core has a 32KB L1 instruction cache, a 32KB L1 data cache, a 512KB L2 cache and a vector processing unit (VPU) that contains 32 512-bit registers capable of processing 16 single-precision or 8 double-precision floating-point arithmetic operations or 32-bit integers in parallel [2]. Each core is capable of multithreading and contains four hardware threads. Thus each of our compute nodes is capable of running 488 threads. The cores communicate with each other and maintain L2 cache coherency through the ring interconnect

ISBN: 1-60132-436-7, CSREA Press ©

with distributed tag directories. The bi-directional ring interconnect also allows the cores to access data and instructions from main memory via the memory controller [3]. Figure 1 illustrates the main components of the Xeon Phi's architecture.

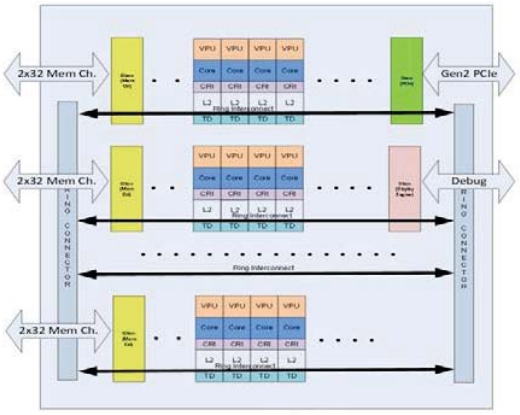


Figure 1: Xeon Phi Architecture [2].

The Xeon Phi communicates with its host processor via a Peripheral Component Interconnect Express (PCIe) bus. Because the card does not have any input and output options, all data travels through the PCIe interface and thus the PCIe bus is a source of data transfer overhead. Also, the Xeon Phi coprocessors run at approximately one third the rate of the Xeon host processors.

These factors: the large VPUs, the bi-directional ring interconnect, the PCIe bus and the 512-bit SIMD capability make it necessary to determine when and how to utilize the Xeon Phi MIC architecture.

2.3 Network Topology

The Jackson State University (JSU) cluster is connected via a10Gb Ethernet network and FDR InfinBand as shown in Figure 2.

The InfiniBand core fabric connects the compute nodes and enables high-speed data transfer between them. While the 10Gb and 1 Gb switches facilitate network management and are outward facing to the data centers switches and enable SSH communication remotely.

3 Software Installation

HPHC clusters require management software. Our software of choice is Bright Cluster Manager from Bright Computing. The recommended method of installing Bright Cluster Manager is on a bare metal system. A bare metal system is less prone to installation errors because it has no previous configuration and the OS is loaded on the machine during this process [4]. The Bright Computing management software is a robust and powerful set of tools that can seem to have a formidable learning curve for the complete novice to cluster computing and managing cluster environments in general and Bright Management software specifically. Once a new user familiarizes themselves with the various manuals and performs hands-on work with the software its power and ease of use becomes evident.

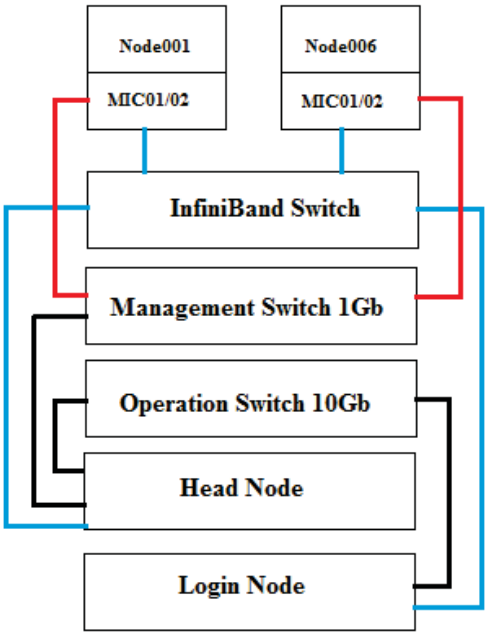


Figure 2: JSU CS 400 Cray HPHC Cluster.

We followed the add-on installation method because after receipt of our system we installed the OS in order to configure it for integration into the university's network. We installed Redhat Enterprise Linux Server version 6.5 (Santiago) kernel 2.6.32-431.el6.x86_64.

The add-on process initially lead to several mistakes and was a definite learning by trial and error experience. It required the removal of an improper install of Bright Management and starting over completely. This paper will not go into the many wrong paths we took but will emphasize the most valuable lessons learned in proceeding in an incorrect manner and will illustrate the final successful installation of Bright Cluster Manager. The most important lessons gleaned from the frustrating initial incorrect install is that Bright Cluster Manager must be installed via a package installer when the add-on method is chosen and that Bright Cluster Manager must be allowed to override any conflict with anything that is currently running on the system [4].

When installing Bright Cluster Manager, the system must have repository access to the distribution of the OS. In our case this required registering with Redhat Network (RHN) and obtaining a subscription to Redhat Subscription Manager. From the command line we entered:

rhn_register

This command initiates the registration process and requires an account sign-in to the RHN portal after which the hardware and necessary package profiles are registered and transferred to RHN. Our system required the additional configuration of the *rhel-x86_64-server-6* and the *rhel-x86_64-server-optional-6* RHN channels.

After the registration is reviewed and accepted, the system requires a reboot and the Redhat daemon *rhnsd* starts. This allows the system to synchronize with Redhat and enables the *yum-rhn-plugin* which in turn makes it possible to use yum for repository access for updates and package installations.

To ensure registration completely successfully and that the system has full access to the repositories, we ran the command:

yum repolist

This command generates a response that confirms the system's registration to Redhat Subscription Manager and that it has access to the base repositories needed to package downloads and updates.

Upon completion of registration Bright Cluster Manager, installation can begin. The Bright Cluster Manager package can be installed directly via DVD or ISO without the package installer but this will lead to errors that make the target system prone to less than complete or useful functionality. The *bright-installer 7.1* can be pulled from the DVD and then installed with rpm as root on the head node with the following command:

rpm –ivh bright-installer-bright-129_cmbright.\

We were prompted to install several packages, which we did with yum and then began the install with:

#install-bright –n

This initiates an install with an internet connection. The installer will then alert to any software conflicts and take whatever action the user decides and then requests the user's license key. After inputting the key, the configuration stage is reached. The installer has default vales which we changed to the correct values for our network. After entering the interface values and addresses the Bright Cluster Manager packages are installed. After a successful install the user is told that the install is complete and they have successfully configured their cluster's head node.

3.1 Compute Node Image Creation

The compute nodes receive a software image from the image directory on the head node. Our image was the standard default image customized with the particular software packages we installed for our requirements. We created our custom image with this command:

#cm-create-image

We gave the custom image a relevant name and placed it with the default image in the images directory on the head node. Then, we set our new custom image as the default image to be provisioned to the compute nodes upon booting.

After powering up, the nodes receive the bootloader from the image distributed by the head node which loads the Linux kernel and allows the compute nodes to complete the boot process. The nodes boot from the network via the 10 GB Ethernet connection.

3.2 Many Platform Software Stack (MPSS)

The MPSS is the software that is required for the Xeon Phi to function. Installation is performed through the Bright software manager and was completed using YUM. The entire set of packages for our OS was installed using the following command:

#yum install intel-mic-*-rhel6.5 –
installroot=/cm/images/michost-image

This command was based on the version of the cluster's OS and the path and name of the custom compute node image. After the system installs MPSS it is then necessary to add the MIC environment variables with this command:

#module add intel/mic/runtime

After this executing this command, the MPSS is ready to configure the Xeon Phi coprocessors. This is done by using Bright's MIC configuration tool. Entering the following command invokes the tool:

#cm-mic-setup

The configuration tool can also be accessed through the Bright *Create MICs* GUI wizard. The JSU MICs were configured via the command line in interactive mode. During this process, the tool requested several pieces of information, including the addresses and various settings of the hosts and coprocessors that form the cluster. After the nodes are configured and integrated into the cluster, provisioning roles were assigned. In the case of the JSU cluster the head node kept its default role of boot and provisioning node. The size of the cluster did not require multiple boot nodes or nodes that need to handle

provisioning and image distribution to ordinary computational nodes. After the roles were assigned and images are set in the correct directories, the cluster was ready to be rebooted and set up for its first set of computation tasks.

3.3 Module Environment

After logging into the head node via SSH, it is necessary to complete several steps to prepare the cluster for program execution. Programs are stored centrally on the head node and are accessed by using the *modules environment* software package. This third party piece of software was installed with the Bright Cluster Manager install and works seamlessly with the Bright software. From the command line the list of available programs that can be shared across the nodes is accessed via the following command:

module avail

This command will list the available packages that the user can load once logged into a compute node, also accessed via SSH. To obtain the full set of modules this command is executed:

module load shared

This makes all programs that have been loaded onto the head node available. It is possible for the system administrator to create persistent profiles for users or to write shell scripts that specify modules to be loaded by default when a particular user completes a bash login.

3.4 Intel Compiler

The Intel compilers rely on environment variables to function properly. First, we needed to execute the setup script to configure the Linux runtime environment. We used ***compilervars.sh*** for BASH shell. We have to source this with the following command on the head node:

#source /opt/intel/compilers_and_libraries_2016/linux/bin/compilervars.sh intel64 or source opt/intel/bin/compilervars.sh.

We then checked to make sure that environment is properly set up by running:

#icc –V

Which if the compiler environment is active, it returns the version and build number.

Next it was necessary to start the flex license service on the head node to enable usage of the Intel compiler. This was accomplished by locating folder which holds the Flexlm license file which was: ***opt/intel/licenses***. In this folder, the lmgrd process was started with the following command:

#: /lmgrd – c

After executing this command, it is now possible to compile and execute software on the head node. However, the head node does not have MICs attached to it so our application execution was performed on node001. Several additional steps were required to allow this. From the head node's command prompt, we used SSH to login into node001.

No additional authentication is necessary once a user with sufficient privileges is logged into the system. Next the previously described command ***module load shared*** was run making available the necessary software packages. After the modules were made available, the following command was executed:

#module load intel/compiler/64/16.0.1/2016.1.150

This command loads the Intel Complier suite from tiger onto the node. The finals steps consisted of copying the Flexlm license file, creating both a directory for the Flexlm license file and an environment variable that points to the file location. The directory that was created matches the Flexlm directory on tiger and is also: ***opt/intel/licenses.*** A copy of the Flexlm license file was transferred to this directory from tiger with the secure copy command:

#scp flexlm.lic node001: opt/intel/licenses

After logging back into node001, the environment variable was set to point to the location of the Flexlm license with the following command:

#export INTEL_LICENSE_FILE=opt/intel/licenses

Upon completion of the aforementioned steps node001 was ready for program compilation and execution.

4 Research

An important step in evaluating new approaches in high performance computing hardware is selecting the correct testing applications. As it is often difficult to port full-scale production quality science and engineering applications to

new platforms, benchmarking tools are normally used. The benchmark tools (or kernels) are greatly reduced small code fragments that only contain the performance intensive computations of the full-scale application [5].

The full-scale applications are, of course, the best performance indicators, but they are usually too large or complex for use in the beginning stages of hardware development and may require a well-trained and experienced software developer to port them to a new system. For this reason, they are usually not implemented until the major design decisions are undertaken and the system is nearing production levels of maturity. However, this presents a problem. Crucial design designs must be made in the beginning stages rather than at the end, and computational kernels and benchmarking tools are becoming more inadequate as accurate measurers of system performance as the platforms and the data sets grow more complex. Developers have noticed an increasing gap between benchmark results and actual application performance on production-level HPC platforms [6].

Factors that affect performance should be defined and understood as early as possible in the development of new hardware systems. The necessity to address the need for software between the two poles of inadequate benchmarking tools and full-scale applications that will enable developers to test their designs and make informed decisions has led to the exploration and development of scaled down versions of production level applications.

4.1 The Mantevo Project

These smaller versions contain the performance-intensive computational kernels of the full-scale applications, like benchmarks, but they also contain the context of the computational components of the full application and include libraries wrapped in a test driver providing representative inputs [4]. These compact self-contained proxies, called mini-apps (proxy application), were developed under the auspices of the Mantevo Project led by Michael Heroux and Richard Barrett of Sandia National Laboratories. The Mantevo Project has several collaborators. Among them are: Livermore computational physicists David Richards and James Belak, various researchers from Los Alamos and Sandia national laboratories, NVIDIA Corporation, and three British institutions—the University of Bristol, the University of Warwick, and the Atomic Weapons Establishment [5]. Mantevo mini-apps are reliable and accurate predictors of application and system performance. The mini-apps average 5,000 lines of code as opposed to many full-scale applications which can easily contain over a million lines of source code. As such, they are much easier to understand, deploy, change or rewrite and are becoming widely used in HPC design because their implementation requires a fraction of the time, effort and training that porting a full-scale application demands.

4.2 MiniMD

MiniMD is a simplified, miniature version of the popular Sandia National Laboratories Large-scale Atomic Molecular Massively Parallel Simulator (LAMMPS) program. MiniMD's source code is less than 3,000 lines as compared to the full-scale size of LAMMPS which is over 130,000 line of C++ [4]. Similar to LAMMPS MiniMD uses spatial decomposition MD. The user can also specify several parameters including problem size, temperature, atom density and number, timestep size and number of timestep, potential cut-off, skin distance and frequency of Neighbor List rebuilds. These parameters are defined in an input file prior to compilation [7].

Unlike LAMMPS, MiniMD only supports the Lennard-Jones (LJ) inter-atomic potential. No other pair interaction features such as long-range electrostatics or molecular force field calculators are available. Combining these features would have made MiniMD unnecessarily complex and too unwieldy for a novice to MD and would have also resulted in a program difficult to port to new hardware for testing purposes. Despite MiniMD's reduced size, it performs its role as well as, and in many cases better than, much larger and more complex programs.

4.3 Porting MiniMD to the Intel Xeon Phi™

The primary reason for obtaining the Cray CS 400 cluster was to perform research on HPHC by analyzing and comparing the performance of the miniMD mini application on a conventional multicore Xeon CPU with its execution on the Intel Xeon Phi architecture. The objective is to achieve and demonstrate a significant increase in speed and efficiency in runtime on the MIC hardware over the stand-alone CPU.

The research consisted of executing the algorithm in several configurations: entirely on a conventional multi-core Xeon processor; in native mode where computation takes place entirely on the Xeon Phi coprocessor; in offload execution or heterogeneous programming mode in which the host CPU offloads all or part of the data from one or several host threads to the Xeon Phi coprocessor. Computation starts on the host and as it moves forward the host can send the data to the Xeon Phi coprocessor where the coprocessor and the host can work on it in parallel.

5 Conclusion

The scope of this paper is not to discuss the performance of the cluster but rather to give an overview of the process and software used to configure and manage it. What is relevant to the focus of this paper is that Bright Cluster Manager worked seamlessly with Intel's software and drivers. Future work will detail actual cluster performance in addition to improving the install and administration process.

6 Acknowledgements

This work was supported in part by the U.S. Department of Defense High Performance Computing Modernization Program under the U.S. Army Engineer Research and Development Center (ERDC) contract number W912HZ-15-2-0001 entitled: "*Strategic Cyber Science, Warfare, Security, Application Development and High Performance Computing Research and Development*," and the research project is entitled: "*Investigating High Performance Heterogeneous Computing with Advanced Architectures*," and in part by Army Research Office HBCU/MSI contract number W911NF-13-1-0133 entitled: "*Exploring High Performance Heterogeneous Computing via Hardware/Software Co-Design.*"

7 REFERENCES

[1] T. Cramer, D. Schmidl, M. Klemm and D. an Mey, "OpenMP Programming on Intel® Xeon Phi™ Coprocessors: An Early Performance Comparison," Many-core Applications Research Community (MARC) Symposium at RWTH Aachen University , pp. 38-44, November 2012.

[2] Intel. *Intel Xeon Phi Coprocessor System Software Developers Guide.* [Online]. Available: http://www.intel.com/design/literature.htm, March 2014.

[3] R. Rahman *Intel Xeon Phi Coprocessor Architechture and Tools.* New York: Apress, 2013.

[4] Bright Computing, *Bright Cluster Manager 7.1. Installion Manual.* [Online] Available: www.brightcomputing.com: http://support.brightcomputing.com/manuals/7.1/installation-manual.pdf. December, 2015.

[5] Michael A. Heroux, Douglas W. Doerfler, Paul S. Crozier, James M. Willenbring, H. Carter Edwards, Alan Williams, Mahesh Rajan, Eric R. Keiter, Heidi K. Thornquist, and Robert W. Numrich. *Mantevo Overview.* [Online] Avaialable: Mantevo: https://mantevo.org/MantevoOverview.pdf. September, 2009.

[6] R. Hansen, *Mini-apps Accelerate Hardware and Software Development*. Science and Technology Review. October 2013.

[7] S. J. Pennycook, C. J. Hughesy, M. Smelyanskiyy and S. A. Jarvis "Exploring SIMD for Molecular Dynamics, Using Intel® Xeon Processors and Intel® Xeon Phi™ Coprocessors." 2013 IEEE 27th International Symposium on Parallel and Distributed Processing, pp. 1085-1097, 2013.

SESSION

POSTER PAPERS

Chair(s)

TBA

ISBN: 1-60132-436-7, CSREA Press ©

ISBN: 1-60132-436-7, CSREA Press ©

Visualization tool for batch job analysis on KISTI Supercomputing center

SungJun Kim[1], JaeKoon Lee[1], and TaeYoung Hong[1]
[1]Supercomputing Center, Korea Institute of Science and Technology Information, Daejeon, Rep. of KOREA

Abstract – *Parallel systems such as supercomputers are valuable resources that are each commonly shared among users. System administrators are wanted to know their systems situations like how many jobs running on system? how many resources used to user jobs? Normally batch job schedulers (sun grid engine, load leveler, slurm etc.) are used to handle user jobs on systems. They also provide GUI tools for analyze system status. In this paper, we design visualization tools for our batch job scheduler (sun grid engine:SGE) to understand system status. Some system problems may be caused by user jobs of causing heavy IO or abnormal behavior. This tool will be helpful to administrator to figure out cause of troubles. It could be navigated user jobs allocations on each node at specific time.*

Keywords: visualization, batch scheduler, supercomputer

1 Introduction

As the only national supercomputing center in Korea, we provide HPC systems to the researchers in industries, academia, institutes and government organizations [1]. The main system named Tachyon2 is composed of over 3,200 computing nodes and it's a theoretical peak of 300 teraflops, detail system specification are describing Table 1. It also used as a scheduler Sun Grid Engine (SGE) to carry out the batch job of the cluster [2].

Table 1. Tachyon2 System specification

Section	*Specs.*	
#Nodes	3,176	Computing node
#Racks	34	Computing Racks
Total Cores	25,408	
Processor	Intel Xeon X5570 2.93GHz(Nehalem)	
Memory	DDR3 76.8TB	24GB/node
OS	Linux(Kernel 2.6.18)	
Batch Scheduler	SGE 6.2	

Actually, we were planning to develop visualization tools for monitoring and analysis system so called vCLAM (very large Cluster Log Analysis and Monitoring system). It would be collect various system failure logs and user job logs. Some system failure has related with user behavior. So we were wanted to watch all logs from related with failures. Using this tool, we could be understood correlation of system failures by visualizing the error logs that occurred at the same time.

In this paper, we will be focus on visualization of user batch jobs. For this work, we were stored SGE logs to Mongo DB and were visualized using D3.js and node.js.

2 Design

When we were designing this system, we were considering two features: analysis & statistics. For analysis, we were mapping user job's node allocation information on the rack layout of our systems. For statistics, we were collecting various metrics from the batch job account information.

Figure 1 shows SGE log data process flows of our system. When user job is finished, SGE write job logs to raw log file named accounting. We extract account log & node allocation information from raw logs every morning by cron. Account log has contained the information to make account data like consumed cpu time, number of cpus, job submission time and etc. Node allocation information contains allocation node list per each jobs.

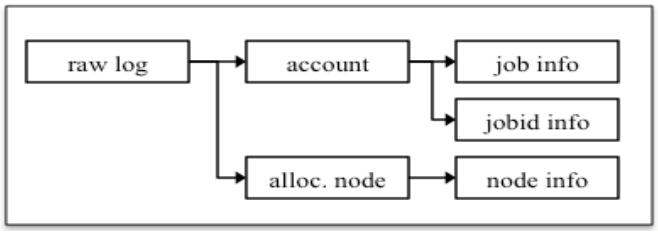


Figure 1 SGE Log data process flow

Figure 2 shows Mongo DB collection's example data of our proposed system. (1) Sge_raw_log collection has contained user job's information extracting from account file of previous steps. It has contained only a part of the account information to generating the statistics. (2) Job id collection has contained job id list which running at specified time. (3) Job node collection has contained allocated node lists per each job.

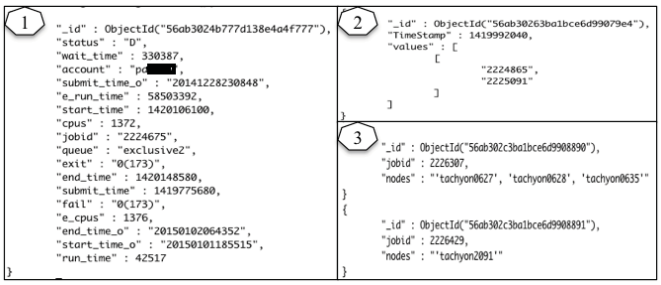


Figure 2 Examples of Mongo DB collections

Using timestamp & allocated node list, we could be time-based navigation and could be indicated allocated node to the user job on rack layout. This feature would be distinguished from other monitoring solutions.

For drawing rack layout of our system, we were defining JSON structure to represent our rack position and structure as Figure 3. It contains whole size of layout, position of racks, size of rack and rack label.

```
{
  "layout" : {"xDim" :  number, "yDim" : number },
   "rack" : [
              { "x" : number, "y": number ,
                "cols" : number, "rows" : number ,
                "name" : String }
            ]
}
```

Figure 3 Layout definition using JSON

3 Implementation

We were used D3.js JavaScript library for manipulating documents based on data and were used node.js for developing server-side web applications.

Figure 4 show node information that was allocated for user job at specific time. Allocated nodes per jobs would be indicated by small rectangles on above of rack layout images by different colors. We could be confirming the information of the node used by the user job according to the change of time, using the instance play function.

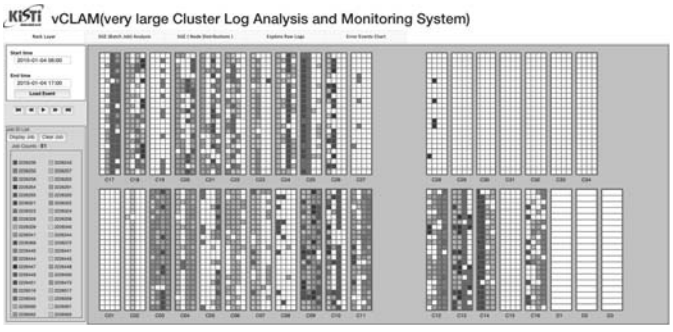


Figure 4 Allocated node information

Figure 5 shows occupied node information given time intervals. (A) shows a part of short runtime & small resources. (B) shows a part of short runtime & many resources. (C) shows a part of long runtime & small resources.

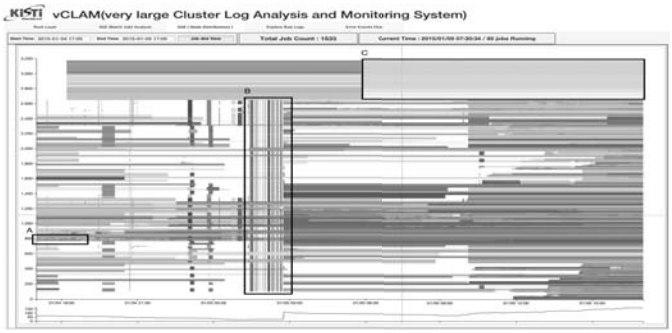


Figure 5 Node allocations of given time range

Figure 6 shows statistics of user jobs for a time periods. These statistics show various user jobs features and could be more helpful to understand of user job's characteristic to administrator.

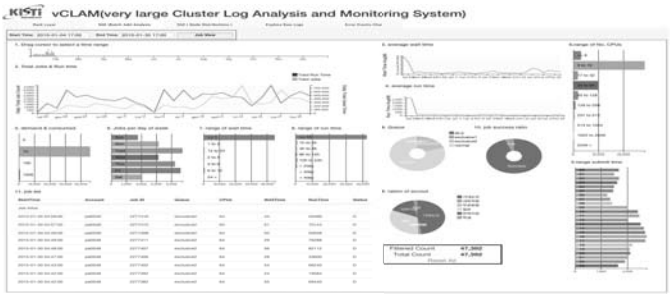


Figure 6 Statistics of user jobs

4 Conclusions

In this paper, we describe our batch job analysis tools. By using this tool, System administrator could be understood user job's characteristic more easily. He could know node allocation information at specific time like system failure time and might be find correlation between them. He could understood pattern of user job's of running durations and consumed resources. After that he could be modified configure of SGE to make more efficiently.

5 References

[1] Sung Jun Kim, et al. "Analysis of user job statistics to maximize HPC resource usage ". The 3rd International Conference on Small & Medium Business ,323-324,2016

[2] JunWeon Yoon, TaeYoung Hong, et al. "Batch Job analysis to Improve the success rate in HPC", Journal of Next Generation Information Technology, Vol.4, No.8, 162-163,Oct 2013

SESSION

LATE BREAKING PAPERS

Chair(s)

TBA

ISBN: 1-60132-436-7, CSREA Press ©

ISBN: 1-60132-436-7, CSREA Press ©

Extending the Performance Evaluation Framework for Auto-Scaling (PEAS)

Kester Leochico and Eugene John
Department of Electrical and Computer Engineering, The University of Texas at San Antonio, San Antonio, TX, United States of America

Abstract – *The Performance Evaluation framework for Auto-Scaling (PEAS) is a framework in which cloud auto-scalers are analyzed in terms of a chance constrained optimization problem that is solved using scenario theory, providing probabilistic guarantees on the obtainable performance of a given auto-scaling algorithm. This represents a major improvement over previous ad-hoc approaches to evaluating auto-scalers by providing a uniform set of metrics and evaluation protocol. This paper represents a first step towards extending the PEAS framework by introducing additional performance metrics from the time series forecasting community as a proof-of-concept of the flexibility of the framework and the reproducibility of the framework's results.*

Keywords: cloud computing, auto-scaling, scenario theory

1 Introduction

Cloud computing, a high performance computing paradigm in which dynamically scalable and virtualized computing resources are provisioned to remote customers over the internet as a service on an as-needed basis [5][7][11], represents a major shift in the way customers and businesses pay for computing resources. By pairing the older software-as-a-service (SaaS) notion of client-server computing with the idea of *utility computing* (the notion of paying for access to computing resources on a pay-as-you-go, as-needed basis), it gives customers access to virtually unlimited amounts of computing resources without the overhead of running and maintaining their own servers, allowing companies to start small and scale up readily as their computing requirements increase [3].

The key property of a cloud computing system that enables the utility computing paradigm that is central to cloud computing is *elasticity*. Not to be confused with the notion of *scalability* (which merely deals with the ability of a system to scale up to meet large workload requirements, regardless of how responsively it does so), elasticity is the property of a computing system or application to dynamically and automatically allocate computing resources at runtime in a timely, responsive manner based on the current resource demand [2][5][6][9]. This quality is a large part of what enables clouds to provide the economic benefits that they do; by scaling quickly, one can respond to changes in the load without violating service-level agreements (SLAs), and by provisioning only as many resources as the system needs to satisfy SLAs, one can reduce the cost (both economic and environmental) of acquiring and using computing resources [3].

Because this is a very difficult task to handle properly when done manually, the task of managing elasticity is typically assigned to cloud auto-scalers, which are subsystems within a cloud that decide how many computing resources to provision in response to current/future demand [9]. However, the current state of the art with respect to auto-scaler evaluation is immature. This is due to a number of factors:

- *A lack of consistent benchmarking metrics.* The issue of cloud auto-scaler evaluation is difficult to properly resolve due to the large number of different aspects to consider (quality of service, cost, etc.), and most extant metrics only measure a few aspects of the problem at a time [5].

- *The lack of formal, standard evaluation methodologies for auto-scaling algorithms*. The lack of widely accepted evaluation scenarios or standardized scoring metrics makes comparing different auto-scalers difficult, if not impossible [5]. Different papers in the literature use different metrics (such as response time, the auto-scaling demand index [ADI] [10], etc.), and different testing procedures (simulation vs. production cloud tests) for determining what constitutes a good cloud auto-scaler. What experimentation is carried out is limited to a handful of experiments that are difficult to generalize for, and are carried out not by comparing a given auto-scaler to other auto-scalers, but to pre-defined response times or static provisioning [1].

- *The use of only a few short workloads to characterize auto-scaler behavior.* Due to the lack of suitable, publically available cloud workloads, much of the auto-scaling literature uses less than three real workload traces, often covering only a few seconds, minutes, or days. This is problematic, both because the workload profile can dramatically affect the performance of a cloud auto-scaler and because the limited amount of workload traces used makes it impossible to draw general conclusions about the behavior of the auto-scalers being evaluated [1].

One proposed solution to the issues facing cloud auto-scaling evaluation is the Performance Evaluation framework for Auto-Scaling (PEAS), first proposed in [1], which aims to address the aforementioned issues with the current state-of-the-art in cloud auto-scaler performance evaluation by providing a formalized, standardized, and mathematically rigorous methodology for providing probabilistic guarantees on the goodness of an auto-scaler's performance based on its distance from the ideal capacity outlay. PEAS achieves these aims in part by running as many as hundreds of workload traces to account for the effects of the workload profile on auto-scaler behavior.

This paper is a first step in extending the the PEAS algorithm beyond the scope of the original paper by introducing several additional metrics for used with PEAS that are pulled from the work of the time series forecasting community and inspired by the work in [13] as a proof-of-concept that the PEAS algorithm can support additional distance metrics beyond.

The rest of this paper is organized as follows: Section 2 provides the requisite background information on the PEAS algorithm, the underlying queuing models and principles behind it, and the auto-scalers and distance metrics used in the experiment. Section 3 describes the research methodology used for the experiment, including the simulation parameters, workloads, and tools involved. Section 4 covers the results as well as commentary on them. Section 5 presents the conclusions of the study.

2 Background Information

2.1 PEAS

PEAS is a framework for providing probabilistic guarantees on auto-scaler performance. It is based on scenario theory, a technique for providing approximate solutions to chance-constrained optimization problems in which instead of solving for an exact solution to a Chance-Constrained Optimization Problem (CCP) (which is NP-hard), one can solve for the best solution under the constraint that the probability that the actual value is less than or equal to that solution is greater than or equal to $1-\epsilon$, where ϵ is the probability that the actual distance value will exceed that of the calculated solution [1].

2.1.1 Underlying Queuing Model

In order to develop the CCP, [1] models the cloud infrastructure as a G/G/N stable queue with a variable number of servers N (which represent the number of VMs) operating in discrete time $k \in \mathbb{N}$ as shown in Figure 1. An elasticity controller attempts to match the current number of servers y(k) such that it satisfies the resource demand brought about by the incoming number of requests λ(k), the number of queued requests that have yet to be serviced q(k), the average service rate per server (in number of requests per time unit) r_{app}, and the required capacity to service long-running requests that require more than 1 time unit to complete C_{lrr}. The ideal number of servers required to meet all currently running requests y°(k) can be modeled by the following equation:

$$y^{\circ}(k) = \left\lceil \frac{\lambda(k)+q(k)}{r_{app}} \right\rceil + C_{lrr}(k) \quad (1)$$

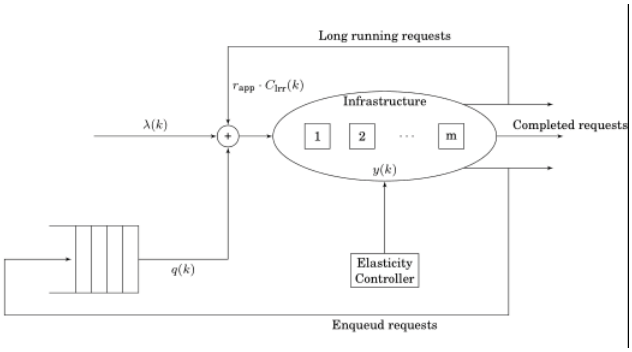


Figure 1: PEAS Queuing Model [1]

2.2 Formulation of the CCP

[1] formulates the CCP for evaluating the goodness of an auto-scaler in PEAS in terms of the following conditions:

$$\text{CCP: } \min_{\rho} \rho$$
$$\text{subject to: } \mathrm{P}\{\mathrm{d}_{\mathcal{T}}(\mathrm{y}, \mathrm{y}^{\circ}) \leq \rho\} \geq 1-\epsilon, \quad (2)$$

where ρ is the probabilistic solution to the CCP and $\mathrm{d}_{\mathcal{T}}(\mathrm{y}, \mathrm{y}^{\circ})$ is a distance function that represents the difference in behavior between the time series representing the actual and ideal capacity respectively y and y°. This distance function is used as the performance metric to be considered by the PEAS algorithm, and any distance function that matches takes in those two inputs can be used.

2.3 Application of Scenario Theory

2.3.1 Algorithm

To solve for the aforementioned CCP, PEAS runs N experiments using N different time series, each of which represents a different representation of the stochastic input, in order to generate the time series traces $y^{(i)}$ and $y^{\circ(i)}$ for each experiment before generating the distance values, filtering out the κ largest distance values from the set, and returning the largest remaining value in the set. The exact process for doing so is as follows [1]:

1. Get N different time series $\lambda^{(i)}(k),\ k = 1, 2, \ldots, |\mathcal{T}|,\ i = 1, 2, \ldots, N$ to use with the auto-scaling algorithms and let $\kappa = \lfloor \eta N \rfloor$.

2. Run the auto-scaling algorithms against each time series to generate y(k) and y°(k) for each time series.

3. Compute the distance values $\hat{\rho}^{(i)} := d_{\mathcal{T}}(y^{(i)}, y^{\circ(i)})$ for each time series $i = 1, 2, \ldots, N$.

4. Determine the indices in the set of all input time series $(\{h_1, h_2, \ldots, h_\kappa\} \subset \{1, 2, \ldots, N\})$ of the κ largest values of the set of all distance values $\{\hat{\rho}^{(i)}, i = 1, 2, \ldots, N\}$

5. Return the largest value of $\hat{\rho}^{(i)}$ from the set of indices that are not in the κ largest values of the distance values set $(\hat{\rho}^* = \max_{i \in \{1, 2, \ldots, N\} \setminus \{h_1, h_2, \ldots, h_\kappa\}} \hat{\rho}^{(i)})$

The number of experiments N and the number of results to disregard κ are in turn affected by the empirical violation parameter η and the confidence parameter β. κ is set by the following equation [1]:

$$\kappa = \lfloor \eta N \rfloor \quad (3)$$

β, in turn, is used as part of solving the following theorem along with η to produce a value for N [1]:

If N is such that

$$\sum_{i=0}^{\lfloor \eta N \rfloor} \binom{N}{i} \epsilon^i (1-\epsilon)^{N-i} \leq 1 - \beta,$$

Then the solution to the PEAS algorithm satisfies the restriction

$$\mathbb{P}\{d_{\mathcal{T}}(y, y^\circ) \leq \hat{\rho}^*\} \geq 1 - \epsilon \quad (4)$$

The following guidelines hold when choosing values of β, η, and ϵ [1]:

- Increased η tends to result in increased N, as N scales as $\frac{1}{\epsilon - \eta}$, so the value of η depends on the desired value of N.
- β controls the probability that the size of the violation set will be larger than ϵ. Since the value of N is logarithmically proportional to $\frac{1}{\beta}$, β can be set as low as 10^{-10} without a significant increase in N.
- ϵ has a significant impact on the number of time series to run the algorithm with.

3 Research Methodology

3.1 Simulation Parameters

The experiments that are the subject of this paper reuse the original parameters from [1] where feasible. In particular, the following simulation parameters are used for the PEAS algorithm:

- $\eta = 0.01$
- $\epsilon = 0.05$
- $\beta = 10^{-10}$
- $N = 796$
- $\kappa = \lfloor \eta N \rfloor = 7$

The only difference lies in the workload scaling factor (which is set to 1 in this set of experiments) and the average service rate r_{app} (which is set to 22 requests/second = 79,200 requests/hour, in order to reflect the service rate used in [1] while accounting for the lack of workload scaling). The hourly Wikipedia traces from [1] are used as the source of the time series used for each experiment. Finally, a custom Python script was used to do the PEAS post-simulation analysis algorithm described earlier.

3.2 Auto-scalers Used

The experiments described in this paper use the following auto-scaler simulation code courtesy of the original author of [1]. All auto-scaler algorithms are used with the default parameters provided by the simulation code and are briefly described as follows:

- *React* [4]: React is a simple reactive dynamic scaling algorithm that uses threshold-based auto-scaling to add/remove VMs. It is one of the simplest scaling algorithms used in [1], and also the one that generated the best results in that paper.
- *Hist* [12]: Hist uses a histogram-based predictive technique that uses histograms of historical arrival rates to determine the number of resources to provision per hour. It also uses reactive provisioning to correct for prediction errors.
- *Adapt* [2]: Adapt adjusts the number of VMs based on both monitored load changes and predicted load changes. Predictions are based on the rate of change of the workload, and aims to adapt to sudden load changes while preventing premature resource release.

3.3 Distance Formulas Used

The following distance formulas originally from [1] are listed as follows with a brief description. Note that as these are all representing the distance between a desired value and the actual value, the lower the distance value the better:

- *Normalized Distance*: The normalized distance penalizes under-provisioning and over-provisioning identically. It uses the squared 2-norm of the difference vector $\|y^\circ - y\|^2$. To account for the difference in lengths between each time series, the normalization term $\frac{1}{|\mathcal{T}|}$ is introduced. It is represented by the following equation:

$$d_{\mathcal{T}}^{norm}(y, y^\circ) = \frac{1}{|\mathcal{T}|}\sum_{k\in\mathcal{T}} \|y^\circ(k) - y(k)\|^2 \quad (5)$$

- *Directional Hausdorff Distance*: This distance metric uses a modified Hausdorff distance that can account for the maximum discrepancy between the ideal and actual behavior and the probability that y will enter some set within the time horizon $\mathcal{T}$. It is represented by the following equation:

$$d_{\mathcal{T}}^{sup}(y, y^\circ) = \sup_{k\in\mathcal{T}} \|y^\circ(k) - y(k)\| \quad (6)$$

- *Over/Under Provisioning:* These distance metrics account for the degree of over- and under-provisioning for a given auto-scaler, and are represented by the following equations:

$$d_{\mathcal{T}}^{over}(y, y^\circ) = \sup_{k\in\mathcal{T}} \|\max\{y(k) - y^\circ(k), 0\}\| \quad (7)$$

$$d_{\mathcal{T}}^{under}(y, y^\circ) = \sup_{k\in\mathcal{T}} \|\max\{y^\circ(k) - y(k), 0\}\| \quad (8)$$

- *Adapted Auto-Scaling Demand Index (ADI):* The ADI [10] is a measure representing the degree to which the auto-scaler is outside a given bound of acceptable utilization levels as represented by the parameters L and U, representing the lower and upper bound of acceptable utilization levels respectively. It is, in effect, a measure of the degree of over- and under-utilization. A time-normalized version of ADI is used as described in [1] and can be represented by the following equations:

$$u(k) = \frac{y(k)}{y^\circ(k)} \quad (9)$$

$$\sigma(k) = \begin{cases} L - u(k) & \text{if } u(k) \le L, \\ u(k) - U & \text{if } u(k) \ge U, \\ 0 & \text{otherwise.} \end{cases} \quad (10)$$

$$\sigma = \sum_{k\in\mathcal{T}} \sigma(k) \quad (11)$$

$$\sigma_{\mathcal{T}} = \frac{\sigma}{|\mathcal{T}|} \quad (12)$$

In addition, three extra distance metrics are used based on the work of the time series forecasting community as described in [8] and [13]. They are briefly described as follows:

- *Root Mean Square Error (RMSE):* RMSE is a scale-dependent metric used in time series forecasting circles, and the formula in terms of the target capacity $y^\circ(k)$ and the actual capacity $y(k)$ is as follows:

$$RMSE = \sqrt{\frac{1}{|\mathcal{T}|}\sum_{k\in\mathcal{T}}\left(y(k) - y^\circ(k)\right)^2} \quad (13)$$

- *Root Mean Square Error (RMSE):* RMSE is a scale-dependent accuracy metric used in time series forecasting circles that is considered to be less sensitive to outliers than RMSE, and is given by the following equation:

$$MAE = \frac{1}{|\mathcal{T}|}\sum_{k\in\mathcal{T}}|y(k) - y^\circ(k)| \quad (14)$$

- *Mean Absolute Percentage Error (MAPE):* MAPE is a percentage error-based accuracy metric used in time series forecasting circles, and is represented by the following equation:

$$MAPE = \frac{1}{|\mathcal{T}|}\sum_{k\in\mathcal{T}}\frac{|y(k) - y^\circ(k)|}{y^\circ(k)} \quad (15)$$

4 Results

Figure 2 shows the solutions to the CCP using equation (5) for each auto-scaling algorithm considered in this paper. As mentioned in [1], the normalized distance provides a good idea of the overall performance, but not any indication whether or there is any over/under-provisioning. Based on these results, React provides the best overall performance, especially in light of the relatively bursty behavior observed in this experiment due to the lack of workload scaling. Curiously, both React and Adapt are tied for best performance for the d^{sup} metric in Figure 3, while the other metrics that are some measure of the overall error (ADI, RMSE, MAE, MAPE) display similar behavior to the d^{norm} metric.

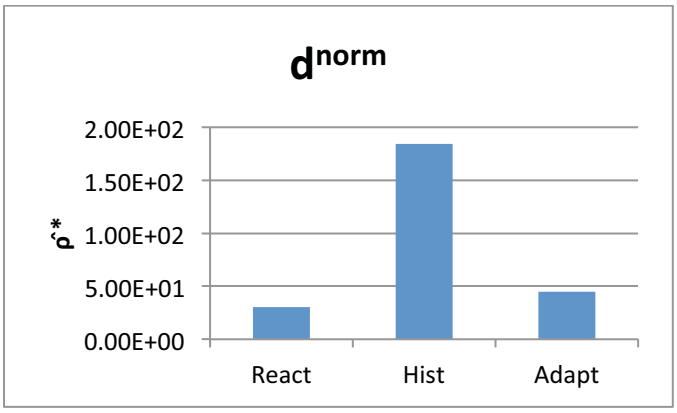


Figure 2: Results of the scenario approach with d^{norm}

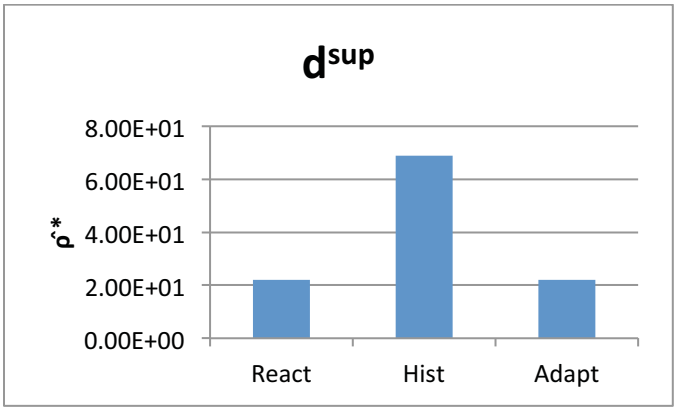


Figure 3: Results of the scenario approach with d^{sup}

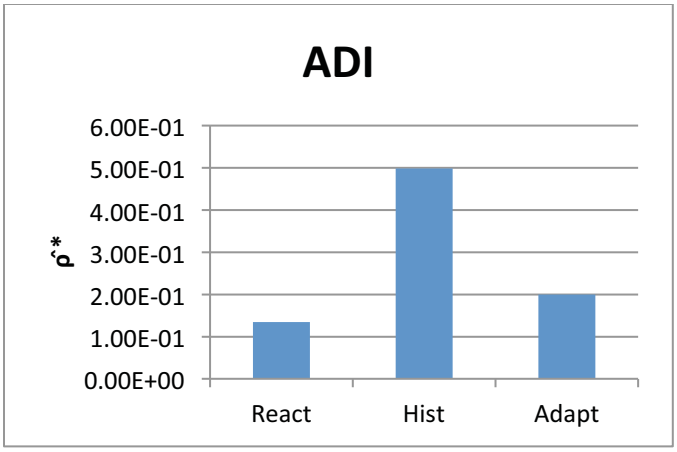


Figure 4: Results of the scenario approach with ADI

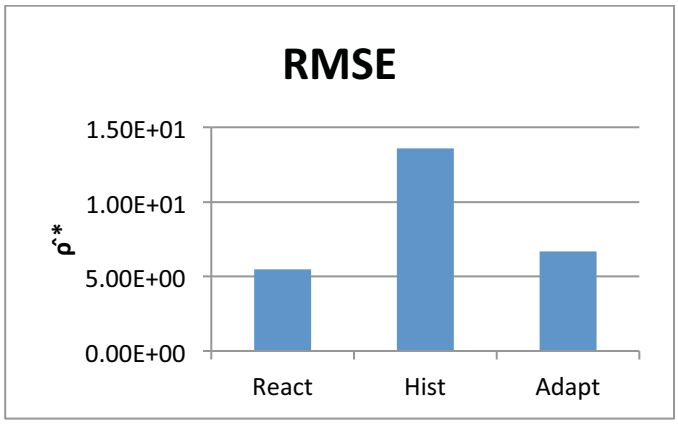


Figure 5: Results of the scenario approach with RMSE

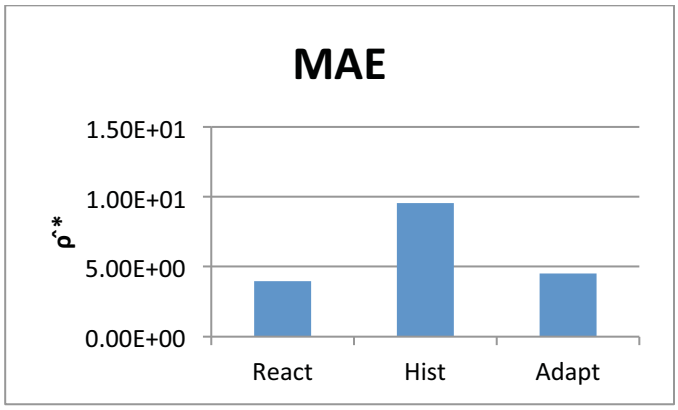


Figure 6: Results of the scenario approach with MAE

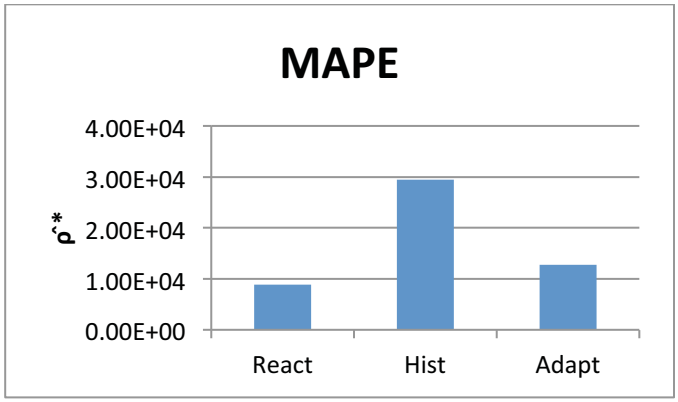


Figure 7: Results of the scenario approach with MAPE

In keeping with the observations in [1], most of the auto-scaling algorithms in Figure 8 tend to over-provision more than under-prevision with the exception of the Hist algorithm. Continuing a trend observed with the previous metrics, the React algorithm displays the lowest overall over-provisioning, while React and Adapt display identical under-provisioning.

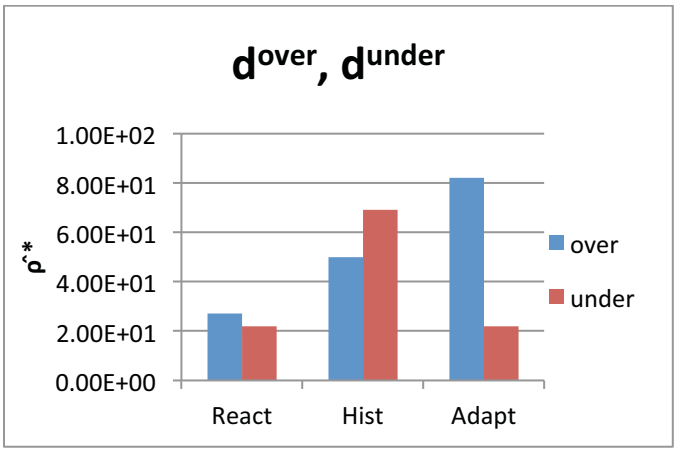


Figure 8: Worst-case over-provisioning and under-provisioning

Table 1: Summary of the obtained results. The best values are highlighted in bold.

d(y, y)	React	Hist	Adapt
norm	**30.06**	184.1	44.57
sup	**22**	69	**22**
over	**27**	50	82
under	**22**	69	**22**
ADI	**0.1342**	0.4981	0.1997
rmse	**5.482**	13.57	6.676
mae	**3.985**	9.541	4.504
mape	**8866**	29460	12780

5 Conclusions and Future Work

In keeping with the results of [1], the React algorithm performs significantly better than other, newer algorithms in just about every metric available in this study, with only the Adapt algorithm providing roughly comparable levels of performance on the d^{sup} and under-provisioning metrics. In addition, the RMSE, MAE, and MAPE metrics fail to provide any appreciable difference in the information provided over the d^{norm} metric in this limited study, suggesting that their utility as performance metrics might not be appreciably better for the purposes of uncovering unique information about auto-scaler performance over the existing ones covered by the original paper.

Nonetheless, the fact that additional metrics could be added to the PEAS framework with little additional reworking demonstrates the flexibility of this approach with regards to analyzing auto-scaling policies. One possible avenue for future work could be to continue to study finding additional metrics from the time series forecasting community that could have considerable utility in uncovering additional information about auto-scaler performance. Another approach could be to extend the work in [13] with regards to analyzing time series forecasting models with the PEAS algorithm discussed here to provide more rigorous performance metrics that are not as dependent on a single workload as in that paper.

6 Acknowledgment

The authors of this paper would like to thank Ahmed Ali-Eldin Hassan from Umeå University for providing the original source code for the Python-based event simulator from [1], source code for the auto-scaling algorithms used in this paper, and the Wikipedia web traces that he used for the original PEAS paper. This paper would not be possible without his support and advice.

7 References

[1] A. Ali-Eldin Hassan, "Workload characterization, controller design and performance evaluation for cloud capacity autoscaling," Umeå University, 2015.

[2] A. Ali-Eldin, J. Tordsson, and E. Elmroth, "An adaptive hybrid elasticity controller for cloud infrastructures," in 2012 IEEE Network Operations and Management Symposium, 2012, pp. 204–212.

[3] M. Armbrust, I. Stoica, M. Zaharia, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, and A. Rabkin, "A view of cloud computing," Commun. ACM, vol. 53, no. 4, p. 50, Apr. 2010.

[4] T. C. Chieu, A. Mohindra, A. A. Karve, and A. Segal, "Dynamic Scaling of Web Applications in a Virtualized Cloud Computing Environment," in 2009 IEEE International Conference on e-Business Engineering, 2009, pp. 281–286.

[5] E. F. Coutinho, F. R. de Carvalho Sousa, P. A. L. Rego, D. G. Gomes, and J. N. de Souza, "Elasticity in cloud computing: a survey," Ann. Telecommun. - Ann. des télécommunications, vol. 70, no. 7–8, pp. 289–309, Aug. 2015.

[6] N. R. Herbst, "Workload Classification and Forecasting," Karlsruhe Institute of Technology, 2012.

[7] H. Hussain, S. U. R. Malik, A. Hameed, S. U. Khan, G. Bickler, N. Min-Allah, M. B. Qureshi, L. Zhang, W. Yongji, N. Ghani, J. Kolodziej, A. Y. Zomaya, C.-Z. Xu, P. Balaji, A. Vishnu, F. Pinel, J. E. Pecero, D. Kliazovich, P. Bouvry, H. Li, L. Wang, D. Chen, and A. Rayes, "A survey on resource allocation in high performance distributed computing systems," Parallel Comput., vol. 39, no. 11, pp. 709–736, Nov. 2013.

[8] R. J. Hyndman and A. B. Koehler, "Another look at measures of forecast accuracy," *Int. J. Forecast.*, vol. 22, no. 4, pp. 679–688, Oct. 2006.

[9] T. Lorido-Botran, J. Miguel-Alonso, and J. A. Lozano, "A Review of Auto-scaling Techniques for Elastic Applications in Cloud Environments," J. Grid Comput., vol. 12, no. 4, pp. 559–592, Dec. 2014.

[10] M. A. S. Netto, C. Cardonha, R. L. F. Cunha, and M. D. Assuncao, "Evaluating Auto-scaling Strategies for Cloud Computing Environments," in *2014 IEEE 22nd International Symposium on Modelling, Analysis & Simulation of Computer and Telecommunication Systems*, 2014, pp. 187–196.

[11] D. Patterson, "Cloud computing and the RAD lab," in Proc. Microsoft Research Cloud Futures Workshop (Redmond, WA, 2010), General Session Keynote,< http://research. microsoft. com/en-us/events/cloudfutures2010/videos. aspx>, 2010.

[12] B. Urgaonkar, P. Shenoy, A. Chandra, and P. Goyal, "Dynamic Provisioning of Multi-tier Internet Applications," in Second International Conference on Autonomic Computing (ICAC'05), 2005, pp. 217–228.

[13] C. Vasquez, "Time Series Forecasting of Cloud Data Center Workloads for Dynamic Resource Provisioning," M.S. thesis, Dept. Elec. and Comp. Eng, Univ. Texas, San Antonio, 2015.

Evaluating Different Alternatives for Delivery of Digital Services

Sakir Yucel

yucel@bluehen.udel.edu

Abstract - *Our research envisions a framework to analyze dynamics of service delivery over different environments including cloud platforms. Our research takes into account the business objectives, service development and deployment dynamics, data center and cloud dynamics for analysis, prediction, simulation and comparison of value generated by the delivered services to the organization. This paper focuses on the comparison of different environment alternatives to service delivery. In this paper, we explore a methodology for how to compare different service delivery alternatives. This methodology helps decision makers in choosing the best among various service delivery options.*

Keywords: service delivery evaluation, service delivery comparison, service delivery metrics, cloud, data center, business objectives

1 Introduction and Problem Definition

The overall performance of the delivered digital services directly influences the organization's overall success. Businesses need to make sure the delivered digital services meet the expectations of the consumers and that every infrastructure resource is utilized economically.

Data centers have provided infrastructure for the delivery of digital services. There have been recent modernization efforts in data centers such as migrating to converged infrastructures or investing in software defined data center, but the bigger trend is to take advantage of cloud services. Delivery of some digital services is migrating over to clouds for many organizations. Now it is very common for organizations to deliver their digital services over both the infrastructures hosted in data centers and clouds. Most organizations are facing significant decision whether to keep the digital services hosted in traditional premises, or in private cloud, in public cloud, or in hybrid. For this paper, we will consider the following different environments for delivering digital services:

- On-premises: this is the traditional data center approach although various different options can be considered (e.g. more traditional, converged infrastructure, software defined).
- Private cloud: private cloud can be owned by the organization on-premises, or it can be provided/provisioned by providers.
- Public cloud
- Hybrid cloud

Businesses may choose one or some combinations of the above for delivering their digital services. A challenge is to evaluate and compare different environments for the delivery of the services. There are challenges on how to compare different options and decide on the best ones, particularly how different investments and monitoring metrics can be compared to understand if there is significant difference between different options, such as delivering the services on premises versus in cloud. In this paper, we will consider the challenges below:

- What metrics to consider and collect to compare different environments for digital services
- How to make use of the collected metrics
- How to compare the set of metrics

This paper is about how different sets of metrics can be compared to understand if there is significant difference between different options. It also is about what metrics are suitable for such comparison. Such a comparison is helpful for decision makers to compare any different service delivery environments. It is also helpful for decision on migrating which services to cloud.

We believe organizations should follow a methodology for such comparison. In this paper, we will outline a methodology to address the challenges above. The methodology involves:

- Decide on what metrics to compare
- Process the metrics to make them comparable, that is, normalize them before comparison
- Use statistical hypothesis testing for comparison

Regarding first challenge of "what metrics to consider and collect" we will identify several layers in abstracting different sets of metrics. These layers are:

- Business Layer metrics: metrics related to business objectives
- Services/Applications Delivery metrics: these metrics include
  - Development metrics: metrics related to planning, design, development, maintenance, troubleshooting, support of services
  - Usage metrics: metrics related to reliability, availability, quality of services
- IT Layer metrics: metrics related to platforms and infrastructures supplied by IT
- Facilities metrics: metrics related to planning, designing, implementing, operating and maintaining the facilities that host IT platforms and infrastructures.
- Cloud metrics: metrics related to delivering digital services over cloud(s)

Metrics should be defined over multiple layers but that is not enough. Metrics should also be defined across multiple domains, such as over different environments involved in the delivery of services, and additionally any networking/ telecommunication and CDN services that the organization uses. This paper concerns only the layers specified above, and does not incorporate domains.

How to monitor, how often to monitor, collect, aggregate, correlate, predict and forecast metrics will not be discussed in this paper. In the rest of the paper, we will provide list of metrics organizations may choose for each layer, then outline the suggested methods for comparing them.

2 Metrics in Layers

Normally, the organization should treat different types of comparable piece of information differently by distinguishing among them. All different types of such information may have different names, however, we will use the term *metrics* in this paper to mean anything measurable or could be assigned a value subjectively. This is to simplify the overall methodology in this paper.

There are many metrics for consideration. Each organization is different and therefore should choose according to their needs for all the above layers as unified teams, which is a challenge. Another challenge is to understand the use cases for the applications/services and choose the most relevant metrics for comparison based on the use case. In this paper, we will mention a rich set of metrics as categorized into different layers. It is expected that an organization works with a small subset of these metrics. We compiled these metrics from various sources but we find [3] very helpful for identifying workable metrics and strongly recommend it for any professional dealing with metrics.

2.1 Business Layer Metrics

These metrics relate to vision, mission, goals and objectives of the organization. In this paper, we consider businesses that heavily depend on their offered digital services and digital products for their revenues, which we may refer to as digital enterprises. Therefore, the overall health of the digital services directly impacts the success of the digital enterprises. In general, businesses care about many metrics related to predicting and improving availability of digital services, staying eco-competitive, managing capacity constraints, lowering operating costs, improving resource utilization and efficiency, managing carbon footprint and other purposes. These metrics can be used to predict higher layer business metrics related to new revenue generation, profitable growth, customer satisfaction and loyalty, staff productivity, cost management, ability for transformation and innovation. In addition, we find metrics under the title of "Business relationship management metrics" from [3] relevant to business layer since they are closely aligned to service level management providing a more strategic viewpoint to services and the business. We highly recommend the reader to refer to [3] for understanding of each of the metrics under that title.

Business layer metrics that are related to business initiatives should be considered together with the digital services of the organization. Such business initiatives include business transitions, product launches, marketing campaigns, advertisement and promotion campaigns. Also macro-economic changes affecting the

digital enterprise should be considered together with the digital services of the organization. The main goal with identifying metrics in this paper is to understand how these initiatives are satisfied by the alternative service delivery environment (e.g. on-premises delivery vs cloud delivery of digital service). For example, how the offering of a digital service on-premises versus in cloud will impact a product launch. The task here is to define metrics for each objective and initiative and assign a score for each metric. These values indicate the level of importance for each objective and initiative per <service, environment>. Later, these values will be used for comparing different environments.

2.2 Applications/Services Delivery Metrics

The idea with this category of metrics is to collect, measure or estimate metrics related to digital services when they are delivered over different environment so that they can be compared. For example, estimate the scores of these identified metrics if/when service is delivered over cloud versus over on-premises for comparison. We categorized these metrics into two: service development and deployment metrics, and service usage metrics.

2.2.1 Service Development and Deployment Metrics

These metrics are related to developing digital services over different environments. These metrics encompass development activities including defining requirements, planning, designing, implementing, maintenance, troubleshooting and support of those digital services. Although there are common activities in the development of digital services irrespective of the end delivery environment, there are also differences in activities when the digital service is developed to be hosted in different environments. These metrics could be evaluated for each digital service and for each possible delivery environment for the service, and then comparison can be made to see the value of digital service to the organization if the service is deployed over different environments. We can summarize the different activities in the development of digital services as follows:

1. Planning activities such as planning for project, procurement, inventory, budget, SLA, platform and infrastructure configuration, overall infrastructure utilization as well as planning for compliance, governance and standardization with respect to established business and IT procedures.
2. Design activities such as designing for compute, storage and networking infrastructure and their virtualization, designing for redundancy, replication, backup and recovery.
3. Implementation activities such as implementing the service as well as designing the database, middleware and other supporting platforms.
4. Build activities involving automation of building the software and testing components, automation of OS and hypervisor build components as well as automation of firmware build components, if any. Build activities involve automating the configuration of compute, network, storage, file system components.
5. Deployment activities including all facility assembly activities (cabling, rack space, hardware assembly, cooling, etc), deployment of hardware, VM, OS, hypervisors, middleware platforms. It includes all clustering and load balancing configuration and deployment.

The task here is to define metrics for each activity and assign a score for each metric. These values indicate the level of complexity for each activity per <service, environment>. Later, these values will be used for comparing different environments.

Definitely some activities are applicable to on-premises deployment, particularly software defined data center environments. These activities are the ones about configuring components via software tools and automating the configuration.

A major activity is establishing business partnerships with suppliers, partners, contractors and all for on-premises environment options and establishing partnership with cloud suppliers for cloud environment.

For cloud, different activities are applicable such as

- Designing modular applications particularly for cloud so that services could be designed as a collection of small and finely grained resources that can be added and removed within the cloud environment. This allows customization of on-cloud deployments which brings in advantages for cost savings based on usage patterns.
- Designing services for avoiding cloud lock-in: services should be designed to be able to migrate to another cloud environment so that the cloud service

provider cannot enforce lock-in kind of pricing when it comes to negotiating with the organization.
- Planning for cloud brokerage services: every enterprise may somehow use multiple cloud environments and those environments would be delivered with heterogeneous platforms. So, it makes sense for the organizations to use capabilities of a cloud brokerage service to be able to manage different cloud environments with a consistent management framework.
- Planning for making space for cloud capabilities: the organizations should recognize cloud computing as an additional set of IT solutions and platforms, and should make space in portfolio for cloud capabilities to be used over long terms.
- Planning for organizational change management: adoption of the cloud is a strategy that requires changes in almost all aspects of the organization. So that means the change in every dimension should be considered and that some organization-wide change management approach should be considered to effectively manage this overall change.

These are additional activities for cloud. The task is to define metrics for each activity and assign a score for each metric. These values indicate the level of complexity for each activity for delivery of a service over cloud.

In addition to metrics related to the complexity of delivering digital services over different environments, metrics related to benefits of the same should be considered as well. These metrics cover the benefits of speeding up decision making, improving process, accelerating infrastructure and system changes in support of business strategies and transformation initiatives, fostering innovation, agility of the service delivery. Again the task is to define metrics for each benefit and assign a score. These scores indicate, for example, the benefit of offering a digital service over cloud on the transformation initiative.

We identified more specific metrics from [3] under this service/application development metrics category. These are Design co-ordination metrics, Release and deployment management metrics, Service validation and testing metrics, Change evaluation metrics, Project management metrics.

All metrics in this section are to be defined and measured (or assigned a score) one by one per <service, environment>.

2.2.2 Service Usage Metrics

These are to be collected, measured or estimated while services are in use. One can compare usage metrics when same services run on different environments, and thereby can compare the value of the <service, environment> for the business. The service usage metrics are collected as part of:

1. Operation activities: all upgrades of components including firmware upgrades, hardware maintenance, facilities maintenance, all service level monitoring activities.
2. Troubleshooting and maintenance activities: Patch management, troubleshooting and root cause analysis, incident management, redeployment of any failing software component.

The task is to define metrics for each activity and assign a score for each metric.

We consider further metrics for service use and categorize them into two groups: (1) a category of metrics for all services combined, (2) and a category of metrics per service.

2.2.2.1 Metrics for All Services Combined

These metrics are considered for business objectives, rather than individual service comparison. The idea for including this category in the paper is that businesses will have better picture about their service portfolio by looking at service catalogs to see what services face the consumers most, what services have strong availability plans, what services are vital for business functions. Using the metrics in this section and following the same methodology we advise in previous sections by assigning scores for each metric, it is possible to make comparisons among the values generated by different delivery environments for all services combined.

We consider the following classes of metrics from [3] under this category: Service catalog management metrics, Availability management metrics (ones that are applicable to all services).

2.2.2.2 Metrics for individual Services

These metrics relate to a specific service and to a particular delivery environment. Consider them for each individual service and environment, that is, per <service, environment>.

Three best known metrics when it comes to service performance are (1) load or demand on the service by service consumers, which usually corresponds to business transactions, (2) response time which is a measurement of performance and highly related to consumer satisfaction, (3) utilization which indicates how busy the service delivery system is. There are many more to consider though. Again we will refer to [3] for other classes of metrics of importance under this category: Service level management metrics, Availability management metrics (ones that are applicable to individual services), Change management metrics, Event management metrics, Incident management metrics, Request fulfillment metrics, Problem management metrics, Access management metrics, Service desk metrics, Risk management metrics. We strongly recommend going over these classes of metrics in [3].

2.3 IT, Facilities, Cloud Layer Metrics

For most large organizations, data centers are managed by IT and facilities groups where these two groups share the budget and responsibilities. IT and facilities metrics are valuable for comparing traditional, converged, software defined and private cloud environments for service delivery. Due to space constraints in this paper, we will not elaborate on specifics of IT, facilities and cloud metrics. In general, metrics should be defined and assigned scores for IT, facilities and cloud on capacity management, service delivery availability and performance, service continuity management, resource management and utilization.

Asset management metrics are applicable to both IT and facilities. For IT, further metrics should be collected on the effectiveness of load balancing, data/service relocation, caching and mirroring, cloud bursting. For facilities, further metrics are applicable about energy efficiency, power usage effectiveness or carbon usage effectiveness, power availability, power chain resilience and redundancy, temperature control effectiveness, coefficient of performance (COP), air flow energy star rating, data center compute efficiency, physical location usage efficiency. Such metrics are helpful when comparing different approaches in data centers, for example, when comparing a more traditional one versus software defined.

Regarding cloud, we summarized some metrics in earlier section and will not elaborate any further due to space constraints. Many references exist for cloud metrics in the literature [4].

3 Comparison of Alternatives

The metrics outlined in previous sections provide a foundation for comparing different delivery environments for digital services. When scores are collected on an ongoing basis for the metrics, statistical and machine learning tools can be applied to assess the effectiveness of different environments for delivery of digital services, and to compare the pros and cons of each. Exploratory analysis can be performed to calculate and visualize correlations among different metrics. Predictive analysis can be performed to forecast the overall value-add of different environments over longer terms.

Since there are many metrics for different delivery environments, it is hard to calculate a utility value out of all the metrics for delivering a digital service over an environment. Hence a set of metric values are to be compared.

In this paper, we discuss how to use the defined metrics and their scores to compare different environments by employing statistical hypothesis testing. Before statistical tests can be executed, metrics values must be pre-processed. Pre-processing the metrics values involve assigning scores to missing values or deciding on what to do for missing values, assigning weights to different metrics based on significance of the metric for the business, and normalizing the values as needed. Normalizing involves making sure a common scheme is followed for values. For example, one approach is that low values always indicate negative effect whereas high values indicate positive effect into the business. Another approach is to convert all values into ordinal values, or make sure the metrics of different schemes are not compared together, and only metrics of same schemes are compared. Another normalization consideration is for cumulative versus non-cumulative values. If a metric is cumulative whereas most others are non-cumulative, then make that metric also non-cumulative when comparing.

Hypothesis testing is a decision-making process for evaluating claims about observations. In hypothesis testing, the researcher defines the observations under study and state the particular hypothesis about the observations. The researcher then collects data, chooses an appropriate statistical test for the observations,

performs the calculations required for the test and reaches a conclusion.

There are two types of statistical hypotheses for each test. The null hypothesis, denoted by H_0, states there is no difference between two parameters. The alternative hypothesis, denoted by H_1, states there is a difference. A statistical test uses data from observations to make a decision about whether the null hypothesis should be rejected. A level of significance, denoted by α, is specified by the researcher and it is the maximum probability of committing an error in rejecting the null hypothesis when it is true. Tests could be one-tailed or two-tailed. One-tailed test is used when the alternative hypothesis states that one parameter is less than the other, or one parameter is greater than the other. The former one is a left-tailed test whereas the latter one is right-tailed. Two-tailed test is used when the alternative hypothesis states the parameters are different, without explicitly stating first one is less than the second or vice versa. Most tests yield a p-value (probability value), which is the probability of getting a sample statistic in the direction of the alternative hypothesis when the null hypothesis is true. When p-value is less than or equal to the significance level, null hypothesis is rejected. When p-value is greater than significance level, null hypothesis is not rejected [5].

Statistical tests can be categorized into parametric and nonparametric tests. Parametric tests are tests for population parameters such as means, variances, and proportions. When the sample size is large or the sample is normally distributed, a parametric test is usually used. Nonparametric tests are used to test hypothesis when assumptions about normality cannot be met. Nonparametric tests are more suitable when data are nominal or ordinal. However, they are less sensitive compared to parametric tests when the assumptions of the parametric methods are met [5].

How do we apply these tests for comparing different service delivery environments? For comparing the two service delivery environments for specific categories where the number of samples is not very large, nonparametric tests are more appropriate when the data is ordinal and no explicit assumption is made on normal distribution. When comparing the service delivery environments in general for all categories combined, we can use parametric tests since sample size will be large enough for these tests.

One question we need ask is about the independence of the samples, that is, whether scores for different service delivery environments are independent or not. Scores are assigned for different service delivery environments and therefore can be considered independent. However, since the metrics are inherently tied to the same business objectives and procedures, they can be considered dependent. We will not make assumption on independence of the samples and will use tests that are appropriate for dependent samples.

In all the tests, our null hypothesis is that two service delivery environments are statistically the same in terms of value to business based on our metrics. The alternative hypothesis states they are of different value to the business. We will choose a significance level of 0.05, which is equivalent to 95% confidence interval.

Many web based tools and software packages are available for running statistical tests [7] [8].

Paired Samples t-test

This is a parametric test used to compare the means of two sets of scores that are directly related to each other. We use this test to compare the service delivery environments in general for all categories. When we compare for all categories, our sample is size is big enough to justify using a parametric test.

The Wilcoxon Test

This is the non-parametric equivalent to the t-test used test whether two dependent samples have been selected from two populations of the same distribution. Wilcoxon test is appropriate for comparing the two service delivery environments with a smaller size, for example, to compare two environments for a select activity.

4 Conclusion

We presented a framework for defining metrics over multiple layers for the delivery of digital services over different environments and comparing them. Main contributions of this paper are:

1. Formulate the different layers of abstraction for metrics
2. Identify and classify metrics into layers
3. Suggest using statistical hypothesis testing for comparison and decision making
4. Devise a methodology for the whole

The methodology presented in this paper can be applied for variety of use cases thereby helping decision makers including:

- Apply the same statistical hypothesis testing for metrics in different categories, even for different activities within categories, or in different layers, or as a whole. Same hypothesis testing can be applied to compare per application/service, per department, per different IT approach. Same hypothesis testing can be applied to compare metrics for different purposes, such as comparing sets of tactical metrics together, or sets of strategic metrics, or sets of operational metrics, provided the metrics are classified respectively.
- Same hypothesis testing can be applied to compare changes to established baselines. A use case for this comparison is comparing new approaches to digital service delivery against standard procedures. Another use case is justification of deviation from standard approach for the purpose of demonstrating improvement, or justification for a new approach.
- With this testing, it is possible to conduct a hierarchy of comparisons. One can compare smaller sets (e.g. activities) at lower layers and aggregate their results for higher layer comparison. Also, one can compare similar set of metrics at any layer and then normalize their results, then compare the results for higher layer reporting. This way, dis-similar sets of metrics are aggregated together to enable higher layer comparisons.
- Before and after comparisons: This method can be used to compare before and after provided "before" and "after" metrics exist. There are many use cases for this type analysis for the delivery of digital services, such as, before and after infrastructure changes, before and after service enhancements.
- This statistical approach is a useful tool in our overall framework. Not just for comparing metrics, but also we use to compare different sets of analytics predictions, and also to compare results of sensitivity analysis exercises in our systems dynamics models.

Further work includes building a more general model for analyzing many dynamics of service delivery over cloud and the methodology presented this paper will be part of this general model. The general model will address cost/benefit aspects of the service delivery, which we omitted in this paper due to its complexity. Our research employs different techniques to understand and model the different dynamics involved in service delivery and its value to the business. We employ system dynamics modeling to combine business objectives and service delivery at a business perspective. System dynamics modeling has been used for analyzing complex nonlinear systems [1]. We employ various machine learning, forecasting and statistical methods for effective management of activities at the IT and facilities layer as well as in the cloud domain [2]. Further work is to be conducted on these efforts.

5 References

[1]. Ibrahim Yucel; Sakir Yucel, "A Model for Commodity Hedging Strategies", The 13th International Conference on Modeling, Simulation and Visualization Methods, MSV'16: July 25-28, 2016, Las Vegas, USA

[2]. Sakir Yucel, "Delivery of Digital Services with Network Effects over Hybrid Cloud", The 12th International Conference on Grid, Cloud, and Cluster Computing, GCC'16: July 25-28, 2016, Las Vegas, USA

[3]. Kurt McWhirter; Ted Gaughan, "The Definitive Guide to IT Service Metrics", Publisher: Itgp, 2012. ISBN-10: 1849284059

[4]. Maitreya Natu; Ratan K. Ghosh; Rudrapatna K. Shyamsundar; Rajiv Ranjan, "Holistic Performance Monitoring of Hybrid Clouds: Complexities and Future Directions", IEEE Cloud Computing (Volume:3, Issue:1), Jan.-Feb. 2016

[5]. Ken Black, "Business Statistics: For Contemporary Decision Making", 8th Edition, Publisher: John Wiley & Sons

[6]. Adapted from articles in CIO Magazine, http://www.cio.com/

[7]. http://www.socscistatistics.com/Default.aspx.

[8]. http://www.r-project.org

Scheduling Based on Overclocknig in Large Clustered Reservation Systems

Ismail Ataie
Islamic Azad University
Parand Branch
Tehran, Iran
ataie.e@piau.ac.ir

Tania Taami
Islamic Azad University
Science and Research Branch
Tehran, Iran
taami.tania@ieee.org

Amir Masoud Rahmani
Islamic Azad University
Science and Research Branch
Tehran, Iran
rahmani@sr.iau.ac.ir

Ahmad Khademzadeh
Iran Telecommunication
Research (ITCR) Center
Tehran, Iran
zadeh@itrc.ac.ir

Abstract— **Advance reservation of resources plays pivotal role in maintaining Quality of Service (QoS) of requests. Resource allocation and management for reservation of requests help us to reach both optimal utilization and QoS enhancement. In spite of sufficient capacity, a reservation request for a resource type may fail with no chance for resolving conflicts. Inflexibility of reservation request in support of replacement according to time axis leads to rigid resource utilization and even poor QoS of the system; however, with the aid of new overclocking technologies for doing overclocking on some current scheduled reservation chunks, new chances emerge to overcome such inefficiency. Model and simulation results show QoS of reservations was improved as a result of using strict overclocking schema with traditional processors in limited time in cluster of servers. This also resulted in utilization of resources improvement and accepted reservations increase without any side effects on processing and reliability of computation.**

Keywords— scheduling; overclocking; thermal behaviour; advance reservation; cluster; QoS;

I. INTRODUCTION

In the center of every conventional system, there is a scheduler to manage and allocate resources to the clients within appropriate elapse of time. The most essential resources in any system, either single or orchestrated system, is processing unit. A challenging task for scheduler is to allocate requests in appropriate duration of time on the appropriate nodes. The focus of this paper is on overclocking computing resource to tackle deficiency of allocating and managing system's resources which eventually lead to improving QoS of reservations.

Several studies have been conducted regarding scheduling in clusters or grid systems [2, 6, 7, 8, 9, 10, 11] as well as scheduling with overclocking capabilities in the single node systems for real-time (periodic and non-periodic) task [1, 5]. Nevertheless, there is no research that integrates these two scheduling processes yet.

In reliable overclocking, computing resource should be controlled so that it does not pass the thermal threshold of equipment [1]. In this paper, simple model for reliable overclocking of processors was introduced to overcome complexity of real thermal model of processors and reduce complexity of computation of thermal radiated from processors. The former can influence any algorithms in real time, and the simplification of later leads to computation time reduction at each stage of algorithm.

Typical physical architectural model of cluster is group of nodes that connected by a shared backbone for processing a bunch of workloads [12]. In this model, each of workloads is divided into two subdivisions. In the first subdivision, workload(s) is deployed to node or nodes and in the second subdivision, workload(s) is started and continued until it is finished. After transferring workload(s) to target(s), computation(s) starts and terminates at the end of its workload(s). There are two constraints on this model: computational capacity of each node and bandwidth capacity of network infrastructure.

Using overclocking, each of reservations or allocations on computing nodes could be relocated to reduce the execution time of each workload. Overclocking of computing resources requires to beware of troubles that might be affect the reliability of results or even chips. On the other hand, solving thermal equations of node(s) is costly for the scheduler in run time [1], therefore, a simple and dependable model should be employed to utilize resources efficiently to improve performance of schedulers.

The layout of this research was drawn as follows: section II describes the system model, reservation model, overclocking concepts and strict overclocking schema. Section III proposes an algorithm that combines overclocking and scheduling mechanisms into harmony. Section IV is dedicated to evaluate the performance of proposed algorithm with the simulation and results. Finally, in section V conclusions of algorithms and proposed overclocking schema presents.

II. MODELS AND OVERCLOCKING CONCEPTS

A. System Model

In the selected model in this research, there is one type of requests which is reservation requests. According to definition each reservation request (R) has five parameters including Rc, Rs, Re, n, Rio. Where Rc is coming time of reservation request,

ISBN: 1-60132-436-7, CSREA Press ©

Rs is start time of reservation, Re is end time of reservation, n is number of processing units that must be allocated to reservation and Rio is required ratio of time to transfer the reservation request to processing units. In this model, within interval of Rs and Re, requests should be guaranteed to get service with n processing units. Reserves could not come into system earlier than Rc time, but they can leave system earlier than Re if all of their jobs on computing nodes has been completed.

The system model in this paper is considered as a cluster of nodes that are connected by a single shared media backbone similar to a LAN network. The cluster consists of one node as a coordinator and n agent nodes A1, A2, .., An. The coordinator node receives requests, reservations, and possible plans to schedule requests on agent nodes by its scheduler module. On the other hand, each agent node has also two major parts: local scheduler and processor frequency controller. The scheduler of coordinator dispatches scheduling timetables and requests that are to be ran on the nodes to their agent schedulers. According to received timetables, local scheduler gives control of processing unit to the request (the reservation). Fig. 1 shows structure of cluster of nodes along with a master or coordinator that manages several agent nodes which all of them are connected to a single backbone.

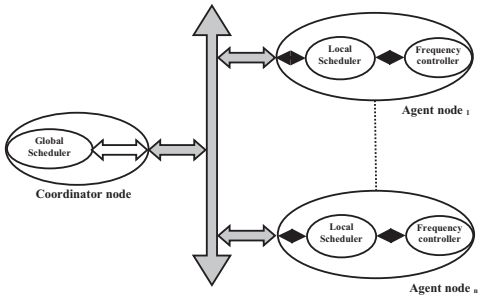


Fig. 1. Topology of cluster of nodes with a coordinator and agent nodes.

According to this model of computation, there are two resources, computing resource and network resource. Based on these two types of resources, there are some conflicts over accessing and utilizing them. Since only one of requests is able to access the network and transfer its data to the destination node, first conflict appears when two or more requests want exclusively to access the network media for communicating and deploying their workloads to a destination node. Another resource is computing power of the nodes. When a request takes control of a node completely to use its processing power in a proper time interval, other requests could not access to the node until the fulfillment of current request.

B. Thermal Model

Relation between processor speed and thermal behavior of each chip can be estimated by the following equation [1]:

$$T'(t) = \frac{\kappa s^{\alpha}(t)}{C} - \frac{T(t)}{R \cdot C} \qquad (1)$$

Where T(t) is temperature at time t and $s(t)$ is speed of processor at time t. The parameters R and C are the thermal resistance and capacitance of chips, respectively (with fan or any peripheral attached to the chips, like heat sink).

The parameter α and κ relate power consumption of processor to its speed. The α parameter has a value of roughly 3.0 [1,3]. For the safety of the system, processor temperature should not reach critical point of temperature due to damaging effects on chip operation.

According to the thermal model in the (1), (2) can be derived for calculating temperature at each elapse of time [1,3]:

$$T_E = T_F + (T_0 - T_F)e^{-t/\tau} \qquad (2)$$

Where, in general, $T_F = R\kappa s_F^{\alpha}$ is constant temperature at overclocking speed of s_F and $T_E = R\kappa s_E^{\alpha}$ is the temperature during this time corresponding to the speed of s_E after elapses of t, and T_0 is the temperature at lowest level at the initial time. Parameter Δ is equal to $R \cdot C$ and t is elapsed time since the temperature was T_0.

By this equation, the t value can be calculated:

$$t = \tau \ln\left(\frac{T_o - T_H}{T_E - T_H}\right) \qquad (3)$$

To avoid complexity and delays of computations at run time in scheduler, simple and effective strict overclocking schema was employed. Consequently, in this schema, three phases in support of CPU frequency scaling, which consist of underclocking phase, normal-clocking phase, and overclocking phase. In underclocking phase (i.e. idle mode) frequency of processor is reduced to minimum possible value which leads to temperature reduction to near minimum possible value. In the overclocking phase, frequency of processor transiently is increased to maximum value until the temperature reaches to the normal point. Finally in the normal-clocking phase, frequency backs to the nominal value in order to continue reminded workload of the request. Based on the fact that the temperature would not pass the normal value, so reliability and continuance of computing operations are preserved. In addition, two working modes in the schema are included, normal load mode and idle load mode. To reduce temperature much more quickly in idle mode; any workload is never deployed to the processor to reduce the frequency to the lowest limitations, i.e. underclocking phase. This situation was exploited by expanding subsequent overclocking interval to the maximum possible value. Using the (3) t and ratio of underclocking to overclocking periods can be calculated.

III. ALGORITHM

In this section a scheduling algorithm is introduced which is useful for describing strict overclocking schema in situations where conflicts are appeared between current reservation request and previous scheduled requests (reservations parts).

As previously discussed, the three-step strict overclocking schema were used in order to overclock each time period of the processors, in the first step, processor node get underclocking frequency with idle workload, in the second step, the node get overclocking frequency, and in the last step, the node get normal-clocking frequency. Only the timeslots of a processor can be overclocked if enough timeslot exists before it which has not been allocated to any request.

In the following algorithms, there are two overclocking approaches: other-overclocking and self-overclocking. In the other-overclocking approach, timeslot of processor belongs to the other previous requests, the reservations, is overclocked, but in the self-overclocking approach, current request in nodes is overclocked.

```
boolean doReserve (R)
1  if (isFreeIO(R.Rs, (R.Re- R.Rs)·R.Rio) == false)
2    return false;
3  AvailabeNodes ← findAvailabeNodes(R.Rs, R.Re);
4  if  (#AvailableNodes < R.n)
5    return doReserveWithOverClock(R);
6  else reserveNodes(AvailabeNodes, R.Rs, R.Re, R.n);
7    return true;
```

Fig. 2. Top level of reservation algorithm.

The *doReserve* algorithm (Fig. 2) firstly tries to schedule reservation R in a cluster of nodes without any overclocking. If it cannot be proceeded, it tries to apply overclocking techniques. The *doReserveWithOverClock* algorithm (Fig. 3) implements a strict overclocking schema that was explained before. First, it finds eligible nodes; the nodes could be overclocked during period of some scheduled tasks or reservations. It proceeds if the request can be scheduled by available nodes with normal-clocking or overclocking other possible nodes, either self-overclocking or other-overclocking, otherwise it fails. Value of Δ is the amount of time that the end of a request shifts to the left on time axe after getting service and completing its workload sooner than normal-clocking because of overclocking schema. The T_{idle} parameter is the required time period which the processor needed to be underclocked with idle workload.

Overclocking schema can be applied to the reservation from the start time of its computation to the end time. That is to say, overclocking cannot be applied in time of communication for each request because communication time is depended on network specification of cluster (i.e. bandwidth) and it cannot be altered or increased without changing physical characteristics of underlying network's components.

```
boolean doReserveWithOverClock (R)
// find and set Eligible Allocation scheduled slot of nodes for
// overcloking
1  EligibleAllocs ← ∅
2  for i = 1 to n
3      Alloci=null;
4      Δ=min((R.Re-R.Rs-R.Rio), maxOCTime)*OCRate;
5      if (Rid = cpuOverlap(nodei, R.Rs, R.Re)) != null and
5.1              isFree(nodei, Rid.Rs-Tidle-Rid.Rio, Rid.Rs) and
5.2              (Rid.Re - Δ) ≤ Rs and
5.3              isFree(nodei, Rid.Re, R.Re) )
6          TimeIntervalnodei, R←( Rid.Rs - Tidl, R.Re-Δ);
7          Alloci= (nodei, Rid, TimeIntervalnodei, R);
8      end if;
9      if (Alloci !=null)
10         eligibleAllocs ← eligibleAllocs + Alloci;
11 end for
12 AvailabeNodes ← findAvailabeNodes(R.Rs, R.Re);
13 if (#EligibleAllocs + #AvailabeNodes ≥ R.n)
14   reserveNodes(AvailabeNodes, R, #AvailabeNodes);
//   reserve nodes with overclocking
15   for i=1 to R.n - #AvailabeNodes
16     RE=EligiblesAlloci.R
17     Δ = min((RE.Re-RE.Rs-RE.Rio),      maxOCTime)*OCRate;
18     EligibleAllocsi.interval.start -= Tidle;
19     EligibleAllocsi.interval.end -= Δ;
20     updateAllocOnNode(EligibleAllocsi.node,EligibleAllocsi);
21     allocateNode(EligibleAllocsi.node, R.Rs, R.Re, R);
22   end for;
23   return true;
24 else
// find nodes that have self OverClocking condition for
// Reservation R
25   selfOCNodes ← ∅
26   for i=1 to n
27    if (isFree(nodei, R.Rs- Tidle, R.Re-(Δ))
28      selfOCNodes +=nodei;
29   end for
30   if (#EligibleAllocs+ #selfOCNodes+#AvailabeNodes
                                                ≥ R.n)
31     reserveNodes(AvailabeNodes, R, # AvailabeNodes);
// reserve nodes for R reservation with overclocking other
// scheduled requests
32     for i=1 to R.n - #AvailabeNodes
33       RE=EligiblesAlloci.R
34       Δ = min((RE.Re-RE.Rs-RE.Rio),maxOCTime)*OCRate;
35       EligibleAllocsi.interval.start -= Tidle;
36       EligibleAllocsi.interval.end -= Δ;
37       updateAllocOnNode(EligibleAllocsi.node,
                                          EligibleAllocsi);
38       allocateNode(EligibleAllocsi.node, R.Rs, R.Re, R);
39     end for;
// reserve nodes for R Reservation with Overclocking R itself
40     for  i=1 to  R.n- (#EligibleAllocs+ #AvailabeNodes)
41       Δ = min((R.Re-R.Rs-R.Rio), maxOCTime)*OCRate;
42       allocStartTime = R.Rs - Tidle;
43       allocEndTime= R.Re - Δ;
44       allocateNode(nodei, allocStartTime, allocEndTime, R);
45     end for;
46     return true;
47   end if;
48  end if;
49 return false;
```

Fig. 3. Strict overclocking schedular algorithm.

IV. Performance Evaluation

For analyzing mentioned strict overclocking schema, a cluster of nodes with various number of processing nodes and reservation requests were simulated. In all simulations, maximum number of requested nodes by each reservation request was number of nodes in cluster. The reservation requests deployed its workload to the nodes by using multicasting approach with the aim of maximizing bandwidth utilization.

For simulating previous mentioned algorithms, the following parameters were used: Arrival time of reservation requests which have Poisson distribution with average of 50 units of time. Initially, the length of requests which being near to overclocking period, i.e. in interval of [40 .. 50], with uniform distribution which is named Δ, was considered. This value of Δ is nearly twice greater of overclocking time length. Secondly, multiples of the Δ in both the utilization of system and the acceptance ratio of system was considered. For calculating the ratio of idle time to overclocking time, Dell Latitude D810 (with Centrino processor) and (3) were used. Based on this conditions, this ratio calculated as 3 to 2, 3 units of time for idle time and 2 units of time for overclocking time. As previously mentioned, the number of requested nodes in each reservation is in the interval of [1 .. number of nodes], i.e. with increasing number of nodes, requested nodes for each reservation request raised. The model was simulated and evaluated for 11 hours. Also the ratio of communication time or the Rio is 0.1 of total workload. Although advance reservation technique was used to guarantee QoS of reservation request in mixed typical tasks and reservation requests, in this model the start of service and the start of request for both conforming with advance reservation models, and simulation purposes (FIFO model) were separated.

Results (Fig. 4) show that using strict overclocking schema improves both of utilization of resources and acceptance ratio of reservation request in scalable form.

Overall, because of multi node reservation requests, which are better responded through dynamism and elasticity of overclocking schema, overclocked schema leads to more utilization of resources in comparison to normal-clocking schema, in spite of convergence of overclocked and normal schema together with decreasing number of requests.

Comparing Fig. 4 to Fig. 5, it is proved that increasing number of nodes do not have any impact on improving utilization and have less impact on acceptance ratio, the ratio of acceptance which are similar to normal-clocking schema.

In other way, with increasing average length of reservation workloads, overall utilization of system in overclocked schema in comparison to the normal clocking increases; the reason is that with increasing the workloads, the side effects of idle time slices, which occurred before each overclocking part of workloads decreases. However, with the growing number of requests at the constant workload rate, this growth starts to decrease since side effects of unused idle times before each overclocked time slice has raised.

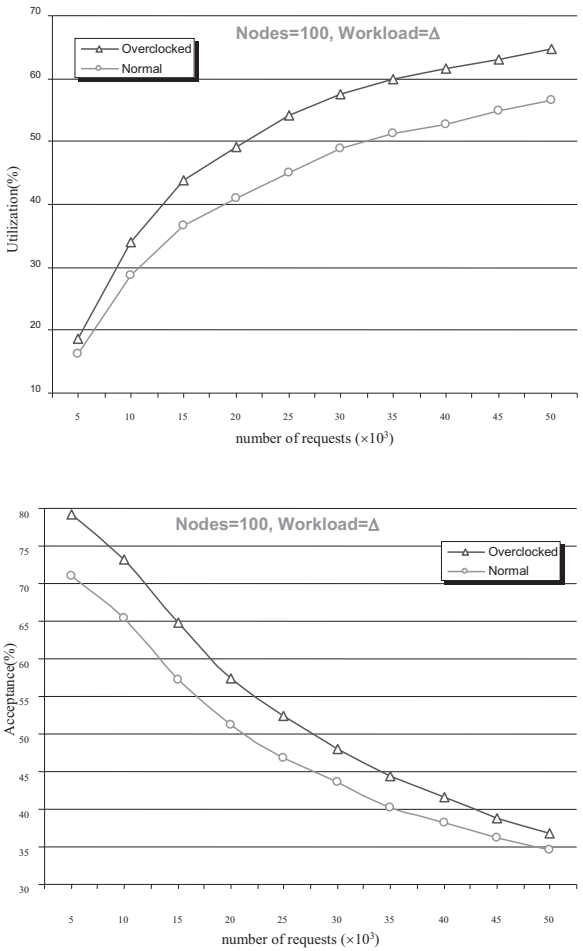
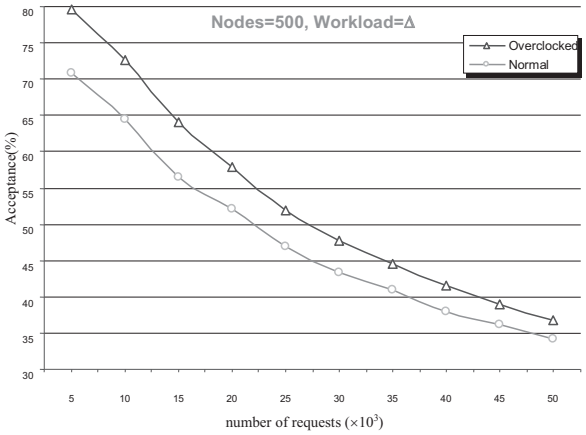
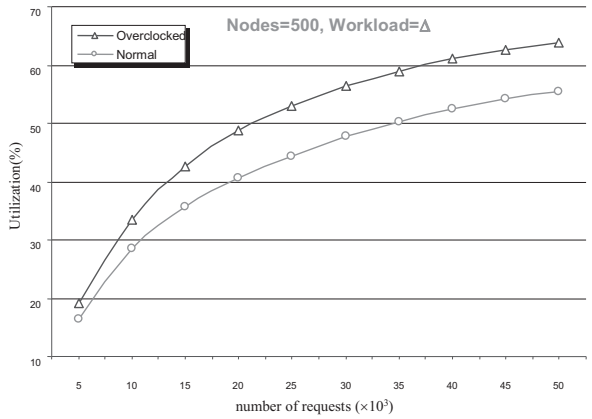


Fig. 4. Acceptance and utilization in 100 nodes.





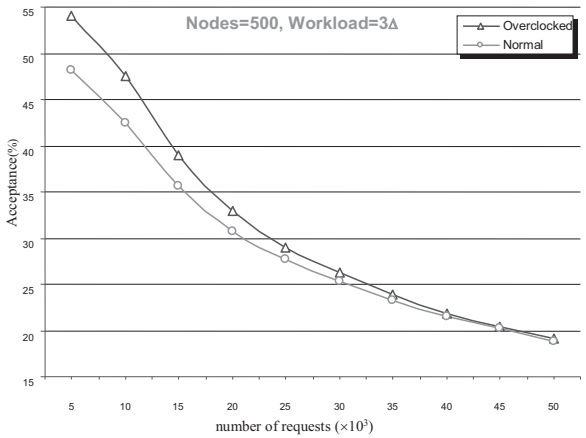
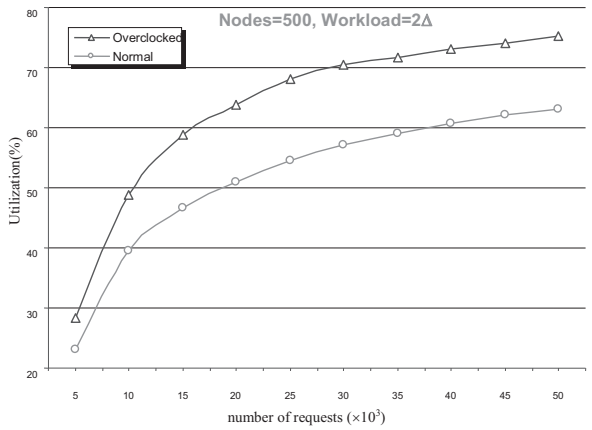
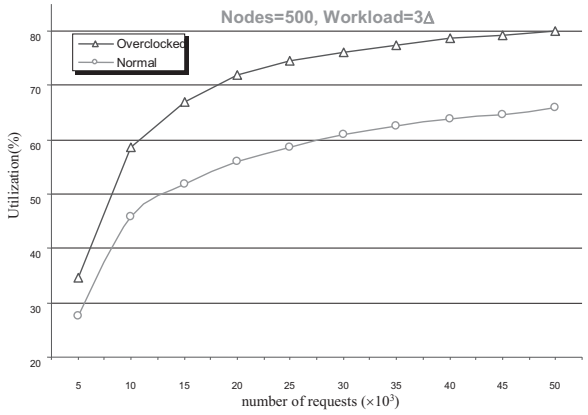


Fig. 5. Acceptance and utilization in 500 nodes.

In all cases, normal and overclocked schema, increasing average length of reservations leads to dropping of acceptance ratio of reservation requests. Such results are obvious; because when the length of reservations increases, the probability of facing reservations with each other increases simultaneously.





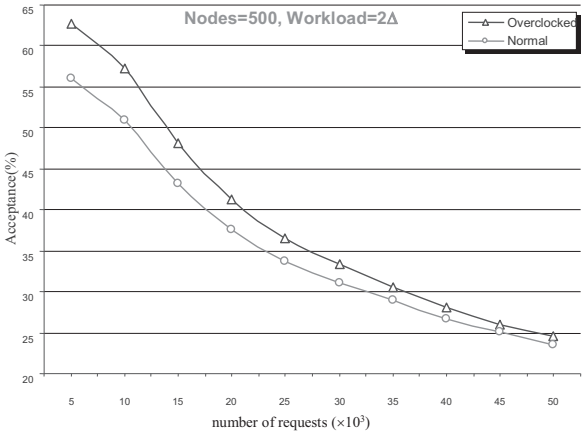


Fig. 6. Acceptance and utilization in 500 nodes with workload of 2Δ and 3Δ.

Indeed, with increasing the workload length of reservations both normal and overclocked schemas improve quickly much more than before until reaching the saturation point. At this point, the overclocking does not have any other influences with increasing number of requests. Fig. 5 and Fig. 6 show this trend. Based on default value of Δ, 2Δ, and 3Δ, graphs of Fig. 7, as a result of increasing average workload of requests, peak point of improvement is shifted to the left, i.e. towards less reservation request numbers. It means that, with increasing workload, probability of happening collision between end time of requests and required idle time intervals which are located before overclocking time phase increases.

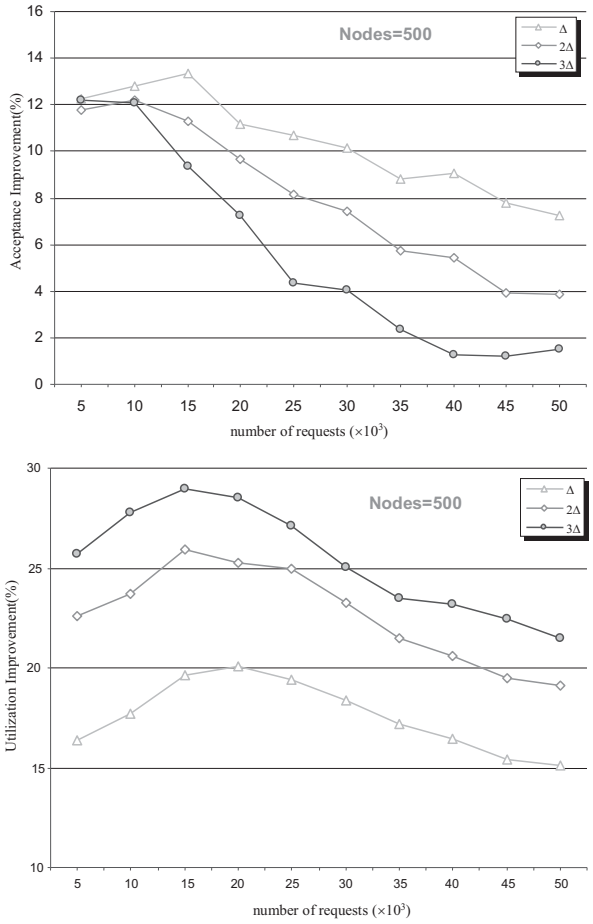


Fig. 7. Acceptance and utilization improvement in 500 nodes with workload of Δ, 2Δ and 3Δ.

V. Conclusions

The results show that utilization of system absolutely increases in contrast to normal-clocking by proposed strict overclocking schema in controlled boundary. Also, the acceptance rate of system with limited conditions improves. In addition, as temperature of processing nodes cannot reach to critical point, the reliability of computation is preserved. With preserving power of processor, economical and commercial aspect of power consumption maintains.

With expanding networks and resources, this schema can be used in large grid networks instead of clusters, since resources are exclusively provided to requests, this model and algorithms is more suitable for private grids that all resources are available for commercial purposes.

Acknowledgment

This work was supported by Iran Telecommunication Research Center (ITRC).

References

[1] Y. Ahn, R. Bettati, "Transient Overclocking for Aperiodic Task Execution in Hard Real-Time Systems", Euromicro Conference on Real-Time Systems (ECRTS '08), Prague, p. 102, 2008.

[2] D. G. Feitelson, "Scheduling parallel jobs on clusters", High Performance Cluster Computing, vol. 1, Architectures and Systems, pp. 519–533, 1999.

[3] N. Bansal and K. Pruhs, "Speed scaling to manage temperature", in Symposium on Theoretical Aspects of Computer Science, 2005.

[4] N. Bansal, T. Kimbrel, and K. Pruhs, "Dynamic speed scaling to manage energy and temperature", IEEE Syposium on Foundations of Computer Science, 2004.

[5] S. Wang, R. Bettati, "Reactive Speed Control in Temperature-Constrained Real-Time Systems", Proceedings of the 18th Euromicro Conference on Real-Time Systems (ECRTS 06), Dresden, Germany, pp. 161-170, July 2006.

[6] L. Eyraud-dubois , G. Mounié , D. Trystram, "Analysis of Scheduling Algorithms with Reservations", Proceedings of the 21st IEEE International Parallel and Distributed Processing Symposium, USA, 2007.

[7] J. Blazewicz, P. Dell'Olmo, M. Drozdowski, P. Maczka, "Scheduling multiprocessor tasks on parallel processors with limited availability", European Journal of Operational Research, vol. 149, pp. 377–389, 2003.

[8] J. Blazewicz, P. Dell'Olmo, M. Drozdowski, P. Maczka, "Scheduling multiprocessor tasks on parallel processors with limited availability", European Journal of Operational Research, vol. 149, pp. 377–389, 2003.

[9] K. Jansen. "Scheduling malleable parallel tasks: An asymptotic fully polynomial time approximation scheme", Algorithmica, vol. 39, pp. 59–81, 2004.

[10] O.H. Kwon, K.Y. Chwa, "Scheduling parallel tasks with individual deadlines", 6th International Symposium on Algorithms and Computation, Springer-Verlag, vol. 215, pp. 198–207, 1995.

[11] V. Subramani, R. Kettimuthu, S. Srinivasan, P. Sadayappan, "Distributed Job Scheduling on Computational Grids Using Multiple Simultaneous Requests", IEEE Computer Society, p. 359, 2002.

[12] A. Mamat, Y. Lu, J. Deogun, S. Goddard, "Real-Time Divisible Load Scheduling with Advance Reservation", Euromicro Conference on Real-Time Systems (ECRTS '08), Prague, pp. 37-46, 2008.

Delivery of Digital Services with Network Effects over Hybrid Cloud

Sakir Yucel

yucel@bluehen.udel.edu

Abstract - *Our research envisions a framework to analyze dynamics of service delivery over different environments including cloud platforms. Our research takes into account the business objectives, service development and deployment dynamics, data center and cloud dynamics for analysis, prediction, simulation and comparison of values generated by the delivered services to the organization. This paper focuses on predicting the load of digital services with network externalities and balancing the load onto on-premise and cloud resources. First a model for digital services with network effects is developed for predicting the expected load. Then how to distribute the load over the on-premise and cloud resources is discussed.*

Keywords: digital service delivery, network effects, hybrid cloud, system dynamics modeling

1 Introduction and Problem Definition

The performance of the delivered digital services directly influences the organization's overall success. Businesses need to make sure the delivered digital services meet the expectations of the consumers of such services and that every infrastructure resource is utilized economically.

Delivery of some digital services is migrating over to clouds for many organizations. Now it is very common for organizations to deliver their digital services over both the infrastructures hosted in data centers and clouds. Most organizations are facing significant decision whether to keep the digital services hosted in traditional premises, or in private cloud, in public cloud, or in hybrid. For adaptive application, cloudbursting can be used to distribute the load to the digital services over the on-premise and cloud resources.

The problem is how to predict the load for services with network externalities. Another problem is how the business product operations such as advertising and customer service effectiveness impact the attractiveness of the digital services with network externalities. Another question is how to balance the total load to digital services over the on-premise and cloud resources for satisfaction of digital service consumers. This paper presents a general model and then uses the model to address these problems.

We will address the above problems from the perspective of businesses offering digital services, digital products and digital goods as their core revenues. We assume the business would like to handle the load to digital services utilizing the on-premise capacity as much as possible, and rent capacity from cloud on-demand in a hybrid cloud deployment model.

2 Dynamics Considered

Many dynamics influence the service delivery on hybrid cloud in general and particularly the prediction of the overall load generated by consumers using the digital services of business. Below we listed some dynamics and briefly mention how they influence the problems this paper is addressing. The paper will consider a select subset of the big list below

1. Business objectives and strategies: Overall business objectives determine what digital services to offer, what market and consumers to target. Business strategy for the overall service delivery affects the delivered digital services as well as the businesses' digitization strategy, mobile strategy and cloud strategy. For most businesses, transformation is a reality in a fast changing business environment and transformation strategy affects the delivery of digital services. Macro-economic changes and seasonality of business have impact on the same.
2. Business operations for products/services: The overall product development process of the business such as for producing products, offering them, servicing them has impact on delivered digital services. Typical business operations that influence the load of digital services include product/service launch, marketing, advertisement and promotions. Also any supply-chain management, manufacturing, inventory, distribution channels for products have impact on the digital services. In this paper, we focus on digital products, and therefore, development of such digital products. For most businesses, digital products correspond to physical products in the background.
3. Service popularity: How the demand side popularity of the services by consumers can be predicted and can be managed? How the customer satisfaction, retention, churn, new customer adoption influences the load on the digital services? The utility of the digital service depends on service's intrinsic utility for example due to its quality and the network effects.
4. Pricing: How the pricing for products/services with network externalities should be managed? How the pricing in competition for example in Cournot competition influences the service adoption? How special pricing for example penetration pricing to reach the critical mass impact the total load due to consumers' use of the

delivered digital services? Pricing can be considered within the business product operations dynamics.

5. Service development: These dynamics are related to total time to offer service and make service enhancements during Planning, Design, Implementation and Provisioning phases of the digital service development. There are some differences in these dynamics for on-premises vs on-cloud development.
6. Service delivery options: how the total load should be balanced over IT/Facilities resource, over private clouds and hybrid environments?
7. Geographic distribution of service delivery: How the estimated total load to the delivered digital service should be distributed over geographic regions for best performance of the service (e.g. for lowest latency to the consumer locations) over on-premise resources in dispersed data center facilities of the business, over CDNs and over Cloud locations?
8. Service capacity requirements: how the required capacity from IT and facilities and clouds should be estimated to serve the demand including the capacity to serve the load on-premises and on-cloud? The latter is the capacity to rent on demand. Additionally, consider the capacity of on-premise resources to handle business product operations, customer service, corporate IT and facilities requirements. The last is included in the model when combining the internal usual business and corporate operations with the requirements of delivered digital services offered to external customers.
9. Cost: how the overall cost for delivering digital services, including the cost of CDN, on-premise resources and cloud can be estimated? How the dynamics of shared and digital economy apply in estimating the cost to deliver the services? How the dynamics of cost of business product operations such as advertisement are incorporated into the total cost?
10. IT: How IT dynamics are incorporated for effective delivery of digital services, especially considering the IT support for such services via Load balancing within and across data centers, Relocation, Replication, Mirroring, Caching, Cloud brokering, Hybrid cloud deployment, Cloud bursting?
11. Facilities: how the dynamics of facilities in supporting the IT infrastructures that host the delivered services should be considered for effective delivery of such services?
12. Cloud: how the dynamics of cloud services in supporting the infrastructures that host the delivered services should be considered for effective delivery of such services?
13. Predictions: What predictions should be tried using System Dynamics, Machine Learning and Statistical methods, and how?
14. Monitoring/analytics roadmap: How the analytics roadmap should be set forth to estimate business impact/value (e.g. generated revenue) of the digital services as well as to measure effectiveness of the presented model in this paper?

3 Intelligence Framework

We employ a layered intelligence approach for analyzing dynamics of service delivery over different environments and for predictions. Layers in our approach are:

1. Business Layer
2. Service Layer
3. Service Delivery Environment Layer: IT/facilities, Cloud

In business layer, we model the business objectives and strategies in Dynamics Considered section using system dynamics modeling. System dynamics (SD) is a useful analysis tool for analyzing and studying the behavior of complex nonlinear dynamic systems by identifying the cause and effect relationships and the feedback control mechanism [4]. Most predictions depend on qualitative inputs at this layer. Prediction is done by building system models and running simulations, performing sensitivity analysis. Qualitative system dynamics approach improves system understanding and prediction for various scenarios, even in the absence of quantitative data. We use SD for modeling various business strategies including service delivery strategy, digitization, mobile, and cloud strategy as well as business transformations and macro-economic changes.

In the service layer, we again use system dynamics modeling but our SD model is integrated with machine learning and statistical methods for various analysis and predictions, for example, for service metrics. We use SD for modeling the dynamics of business operations for products/services, service popularity, pricing, cost, geographic distribution of service delivery, service delivery options and service capacity requirement in Dynamics Considered section.

In service delivery environment layer, we use again SD modeling. We use machine learning and statistical methods in this layer as integrated into SD model and also separately from the SD model in order to model the dynamics of IT, facilities and cloud in Dynamics Considered section [3].

IT and facilities are considered together for the on-premise resources. They work together to realize all on-premise infrastructure could offer in terms of all resources for the digital services. Cloud is also part of this layer as it provides resources for hosting the digital services. In this paper, we consider the IT resources which are utilized for the delivery of digital services as well as IT resources utilized for product operations such as resources for advertising and marketing teams, the software development teams which in turn work on delivering the digital services. IT provides the complete stack for delivering the service. It handles management of multiple disperse data centers for hosting the service. It also mediates the CDN service. For cloud, we assume the business uses IaaS for service delivery.

4 Model for Service Layer

This paper presents the model in the service layer. The model implements only several dynamics from Dynamics Considered section for delivery of digital services over hybrid

cloud environment. In this model, digital products, digital services and/or digital goods with network externalities are considered. The model is modular. In subsequent sections, different modules of the model will be presented.

4.1 Network Effects

This section talks about the network effect dynamics for the delivered digital service. For these services, the customer experience is not a one-time purchase but rather continuous use of the digital service. The utility of product/service for the consumer depends on interconnections among the consumers and potential adopters in addition of the intrinsic utility from the service itself. Interconnection depends the number of users and how the interactions occur among them. In digital services, adoption process does not end by purchase. It continues. The consumer may continue utilizing the service with satisfaction and this will create positive network effects, but also may leave due to dissatisfaction, and this leads to negative effect.

System dynamics based approach for modeling the diffusion of goods considering network externalities has been studied in the literature [1]. Our model is based on the final model in [1] but incorporates few additional variables. Alpha variable for innovators and beta variables for imitators are used in our model as in [1]. Alpha initially helps for increasing the number of users at the beginning to reach the critical mass. Service launch strategy affects the alpha variable. One objective with the model is to simulate the early phase of delivered digital service till reaching the critical mass for the success of the product launch and other business product operations through attempts to activate innovators and imitators by free trial for imitators, referral and/or complementary gifts, and through support from other business product operations during this phase. The model continues the simulation after that as well.

The number of potential users may change based on demographics, economic, or other factors as well as consumer behavior. Consumers adopt if their utility is higher than a level of desired utility. Potential users decide based on not only actual utility but also on expectation of future utility. Similarly for churn of existing adopters.

The patience variable influences the discontinue rate. Existing adopters may become quitters when their patience level becomes low due to low service quality and poor customer service. Quitters may become adopters based on the come-back fraction variable which is dependent on the expected future utility.

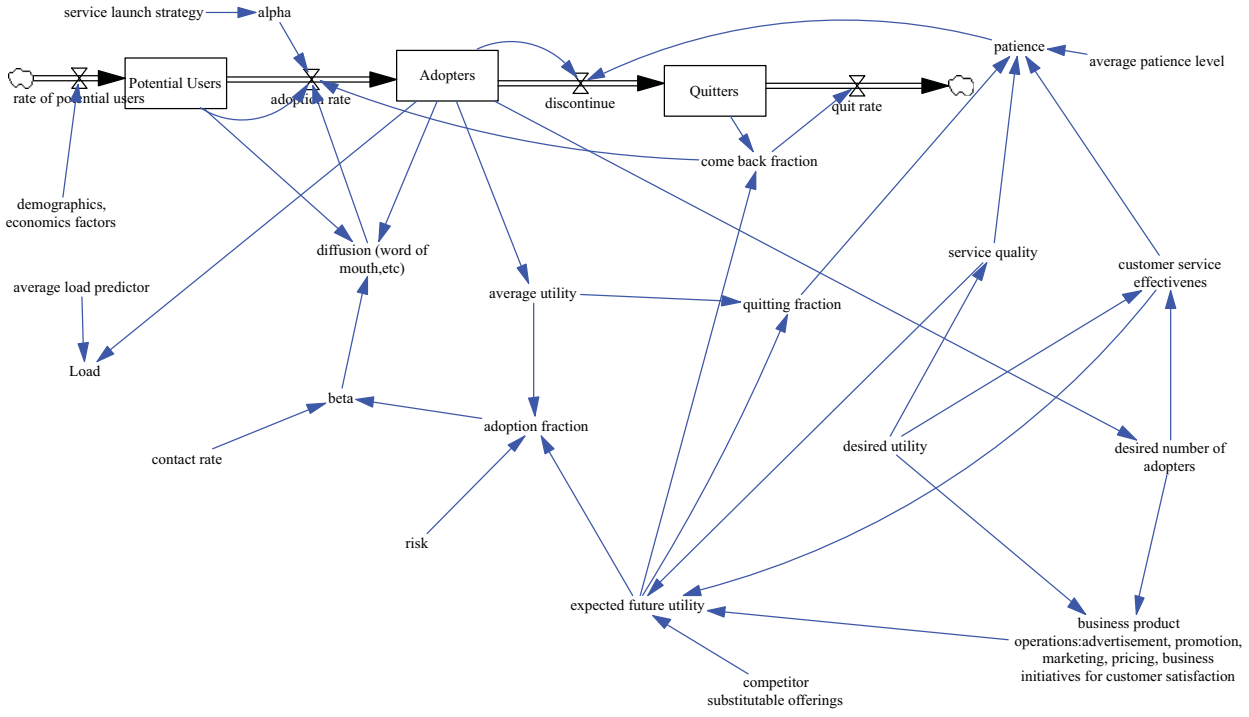


Figure 1 SD Model for Utility to Delivered Service with Network Effects

Desired number of adopters is used to control the load. The load should be managed based on capacity of handling the load of the digital service by IT and cloud resources and by business objectives. This will in turn have an impact on the business product operations such as advertisement, marketing and promotion efforts as well as pricing. Based on this value, businesses may control the amount of such business product operations, for example, not to make large advertisement that will create large loads that will not be handled by the existing resources with quality service, or large loads that will not be effectively supported the underlying product operations (e.g. manufacturing, developing the service, to be able to handle

customer service support inquiries, to fix anomalies, to maintain the business). This variable is also used to manage the additional production capacity to bring in (or contracting the development, outsourcing for development of the service, its maintenance and its delivery, etc). Note that in in our case, it is not physical production/manufacturing but it is about developing the digital service (software, models, digital goods), and bringing the whole business process and infrastructure together to deliver the service.

Desired utility is a variable that management could set to realize it by employing customer service effectiveness, service quality and other business product operations for the existing adopters, and in turn control the churn rate. The management can specify this variable together with the desired number of adopters for a period of time and thereby control the amount of additional business product operations such as advertising.

Following input variables are supplied to this module of the model from external sources or from other modules of the model. The modules supplying these variables are not addressed in this paper.

- Service launch strategy
- Demographics, economic factors, consumer behavior:
- Average load
- Risk (or the opportunity cost)
- Competitor substitute offerings
- Customer service effectiveness: how effectively the customer issues are resolved via customer service.
- Service quality: software development for quality, addressing the defects and customer cases on a timely manner.
- Business product operations such as advertisement, promotion, marketing, pricing
- Business initiatives for customer satisfaction
- Desired number of adopters

One output of this module is the load variable given out by the simulation. This variable indicates the total load due to all requests to the delivered digital service by consumers. When the model is run for different scenarios, strategies and with different values of input variables, different values for the load variable are generated. Part or all of this load would be handled by the IT resources constrained with the capacity of IT resources. When IT resources become not adequate to handle all load, resources will be rented from the cloud provider to handle the remaining load that cannot be handled by the IT resources. Before discussing how to distribute the load in simulation, we will briefly discuss two additional modules that influence the demand to the digital services: advertisement and customer service effectiveness.

4.2 Impact of Advertisement

Advertisement is among business product operations that influence the demand to the product where in this paper we consider the digital services as the main product of the business. The module that models the influence of advertising on the utility of the digital services addresses several aspects: The effect of advertising on the expected future utility, the amount of advertising efforts as determined by the desired utility, the cost due to advertising, the advertisement channels, their characteristics and their effectiveness, and the advertisement spillover that increases utility of the competitors' substitutable offerings.

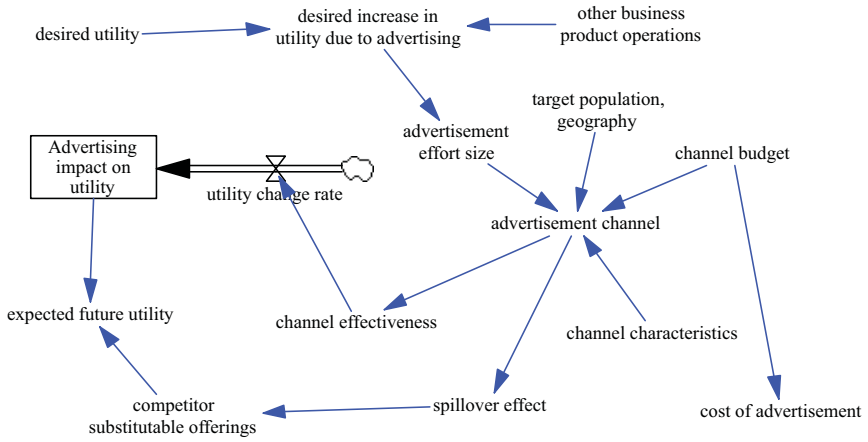


Figure 2 Impact of Advertisement on the Utility of Delivered Service with Network Effects

In this module, desired increase in utility due to advertising is determined by the desired utility and in coordination with other business product operations. Based on the desired increase, the amount of advertisement effort is determined, multiple channels together with their budgets are identified. Each channel has its own inherent characteristics, target population and geography. Channel effectiveness and the spillover are estimated based on existing research on advertisement [2] [5]. Spillover is one of the dynamics that impacts the competitor substitutable offerings. The advertising impact on utility stock variable is the accumulated value returned back to the main module that influences the expected future utility. There is a delay in the realizing the advertising effects.

The cost of advertisement is a separate variable in order to keep each such cost separately so that each can be measured for effectiveness and proper actions to deal with each can be considered. Also note that advertising impact on utility stock variable can be used to estimate the revenue increase due to advertising.

How advertisement influences the service perception? How advertisement alters the utility for a service with network externalities? Although it is not the subject of this paper, we believe advertisement efforts should attempt to increase the utility by advertising the intrinsic utility of the service and also by exploiting the network effects into advertisement.

4.3 Impact of Customer Service Effectiveness

Many times, businesses' delivered services are subject to interruption and downtime. Business continuity is significant for such businesses that heavily rely on the delivered services for their revenues. Also many times customers have issues with effectively utilizing the delivered digital services of the businesses, and for variety reasons, customers file requests. Customer service is very crucial for service oriented businesses. In this section, we will address how effectiveness of customer service impacts the overall service perception and user satisfaction, and in turn, impacts the adoption, quitting and re-adoption.

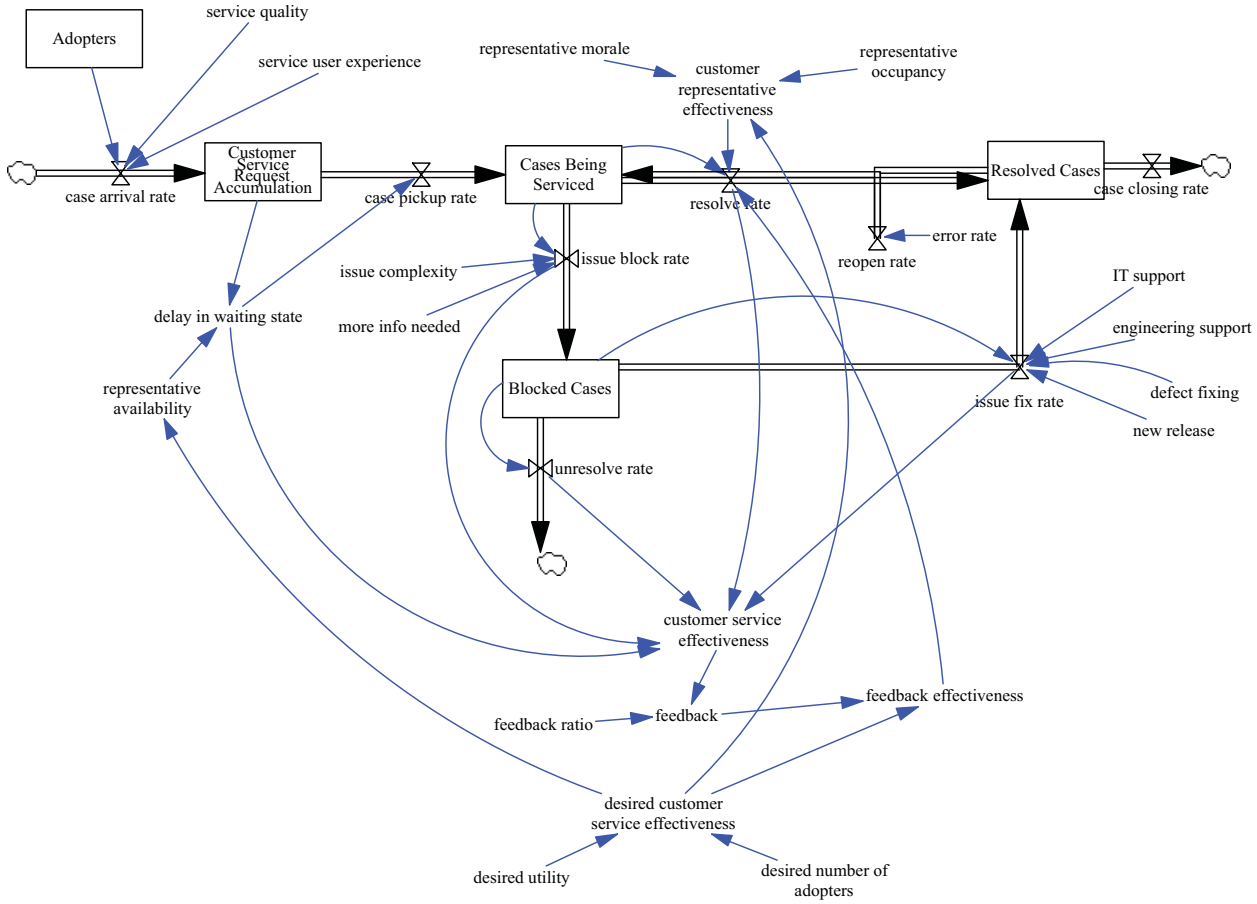


Figure 3 SD Model for Customer Service Effectiveness

For simplicity, all issues are considered at the same priority and severity. The users have a varying level of satisfaction for the provided customer service based on the service's performance. Customer service effectiveness variable holds the positive and negative experience in the form of satisfaction and accumulated frustration. Customer service effectiveness, how effectively the customer issues are resolved via customer service, has impact on patience and expected future utility in Figure 1. This module has two inputs from other modules in in Figure 1 and they are desired utility and desired number of adopters at a given time. With these inputs, the desired customer service effectiveness is determined and it controls representative availability and the amount of customer feedback to incorporate, and customer representative effectiveness. The last is managed by enhancing the representative morale and proper training, among other things. Customer feedback is used to enhance the request resolve rate.

Resolved cases may be re-opened due to errors in handling the cases. Some issues are blocked maybe requiring more input from the customers, or maybe complex issues waiting for expertise. Such cases can be fixed via IT support, engineering support, by fixing the defects followed by re-deployment, by making new release of the delivered service.

4.4 Service Delivery over Hybrid Cloud

The load is obtained from the model by simulation and by performing sensitivity analysis on various variables. This load is to be handled by IT and cloud resources with respect to implicit and specified service level objectives of the consumers. The module responsible for determining how much of this load would be served by IT resources and how much would be rented from cloud uses a number of input variables and implements the load balancing algorithm. It implements a geo-matcher using customer profiles, service level objective expectations and geographic load distribution policies. It includes the inventory of IT and facilities resources in the optimization. It estimates the utilization of IT and facilities infrastructures. It predicts the service delivery performance based on estimated latency and throughput values. The service delivery performance variable is used to predict the service quality in Figure 1. It estimates the marginal cost of service delivery per on-premises, CDN and cloud environments for the given load at a given time. In this module, we employ modified ARIMA, other statistical and machine learning methods into our SD models. Particularly we combine the neural network algorithms with system dynamics modeling for policy development and enforcement in the optimization algorithm. We believe this module together with the optimization algorithm deserve a separate dedicated paper and therefore we will not describe in any more detail.

4.5 Running the Model for Predictions

This model can be used for predicting many outcomes with simulating variety of scenarios. Such predictions are valuable for business intelligence and for effective service management of the delivered Some of the predictions that can be made using the model and addressed in this paper include:

- Adoption rate due to diffusion
- Predicted utility of the service to the adopters by the simulation
- Predicted expected utility of the service to the potential adopters, adopters, quitters.
- Predict customer satisfaction, retention, churn, new customer adoption.
- The impact of business product operations on the expected future utility
- The impact of expected future utility on the quitters for re-adopting the service
- The impact of advertising on the desired and expected future utility
- The impact of customer service effectiveness on patience, on the desired utility and expected future utility
- The impact of the patience in keeping the adopters and leading them to quit
- How much more load can be handled by the services combining the IT and cloud resources together
- How much of the load should move to cloudbursting

Some predictions that can be made when other modules not presented in this paper are incorporated:

- The effectiveness of service launch strategy for reaching the critical mass
- The effectiveness of pricing on the desired utility and expected future utility. Similarly for the promotions, marketing, other business initiatives for customer satisfaction.
- The impact of desired number of adopters on managing the business product operations and business initiatives for customer satisfaction
- The impact of competitor substitute offerings on the expected future utility
- The impact of service quality on the expected future utility and desired utility
- The impact of risk on the adoption of service by potential adopters
- The impact of opportunity cost on the adoption of service by potential adopters
- The impact of competitor substitute offerings on the opportunity cost
- The cost of delivering service
- The cost due to business product operations
- How the quality of the delivered services is influencing the business
- Revenue from delivered digital services
- The impact of delivered digital services on the business
- Estimation of the utilization of IT and facilities infrastructures
- Prediction of the service delivery performance based on estimated latency and throughput values.
- Estimation of the marginal cost of service delivery per on-premises, CDN and cloud environments for the given load at a given time.

As the digital services become the core revenue sources, organizations need a holistic view for the whole business of delivering them to the consumers. Businesses need an overall analytics roadmap for predicting, measuring all aspects of delivering digital services and tailoring the business product operations for their effective delivery. Their analytics roadmap should cover for monitoring/forecasting/prediction of all aspects of service delivery, for evaluating the decisions made, and also for evaluating this model and fine tuning it. We believe a typical machine learning and statistical methods are not enough. Therefore, we presented a more complete approach including Business Layer, Service Layer, Service Delivery Environment Layer in Intelligence Framework Section.

The intelligence including all predictions is achieved by running the model for variety of scenarios and sensitivity analysis. Prediction for a service with network effects is hard and cannot be done using analytics based on statistical and machine learning methods, especially when combined with non-quantitative objectives. Statistical and machine learning methods cannot build effective models in case of shortage of quantitative data, for example for service launch during which

no quantitative data is available. Therefore, they cannot be used to predict the demand for services or to measure the effectiveness of product launch strategy while the business is trying to reach the critical mass. They cannot easily incorporate business dynamics into models, cannot simulate policies and predict output, cannot produce intelligence based on non-quantitative data. We employ modified ARIMA, other statistical and machine learning methods into our analytics approach [3] [4], but use them within our intelligence framework and as part of our system dynamics models.

5 Conclusion

The performance of the delivered digital services directly influences the organization's overall success. Many dynamics influence the performance of the delivered digital services. Businesses need tools and models that allow them to view and simulate different dynamics involved in managing the delivered digital services. In this paper, we outlined different dynamics of service delivery over different environments and defined a layered intelligence approach for analyzing such dynamics. We addressed some of the dynamics in the service layer of the layered intelligence approach by presenting different modules of a system dynamics model. One module is for predicting the load for services with network externalities. We further introduced two new modules into the model to address how advertising and customer service effectiveness impact the attractiveness of the digital services with network externalities. We discussed at a high level how to balance the total load to digital services over the on-premise and cloud resources for satisfaction of digital service consumers. As we outlined the variety of predictions that can be performed, the system dynamics model presented in this paper is a very strong tool to make predictions. Our approach with layered intelligence that employs system dynamics modeling as well as statistical and machine learning methods is very effective for simulating different dynamics of digital service delivery over hybrid clouds.

Our future work includes:

- We are working on a separate paper that presents our optimization algorithm for geographically distributing the total load onto to meet the service level objectives.
- We are addressing other dynamics listed in section Dynamics Considered.
- We are incorporating estimating the total cost of offering digital services into model presented in this paper. We are working on introducing dynamics of digital goods economics into the model as well as cost of developing the service and the cost of business product operations.
- We are developing models for effective cloud brokerage services. Many businesses that offer digital services use cloud brokerage services. IT can act as the broker for the businesses.
- We are developing models similar to one in this paper for migration strategy of the digital services to cloud.

6 References

[1]. Jörn-Henrik Thun, Andreas Größler and Peter M. Milling; The Diffusion of Goods Considering Network Externalities: A System Dynamics-Based Approach, 18th International Conference of the System Dynamics Society, 6-10

[2]. Malcolm Brady, Advertising effectiveness and spillover: simulating strategic interaction using advertising, 25th International Conference of the System Dynamics Society Boston, Massachusetts, 29th July to 2nd August, 2007

[3]. Yucel, S: Evaluating Different Alternatives for Delivery of Digital Services, The 12th International Conference on Grid, Cloud, and Cluster Computing, GCC'16: July 25-28, 2016, Las Vegas, USA

[4]. Yucel, S; Yucel, Ibrahim: A Model for Commodity Hedging Strategies, The 13th International Conference on Modeling, Simulation and Visualization Methods, MSV'16: July 25-28, 2016, Las Vegas, USA

[5]. A. S. Cui, M. Zhao, T. Ravichandran. Market Uncertainty and Dynamic New Product Launch Strategies: A System Dynamics Model, IEEE Transactions on Engineering Management (Volume:58 , Issue: 3)

[6]. Detlef Schoder, Forecasting the success of telecommunication services in the presence of network effects, Information Economics and Policy, Volume 12, Issue 2, June 2000, Pages 181–200

[7]. Jae Choi, Derek L. Nazareth, Hemant K. Jain; The Impact of SOA Implementation on IT-Business Alignment: A System Dynamics Approach, Journal ACM Transactions on Management Information Systems, Vol 4 Issue 1, April 2013, Article No. 3

[8]. Margherita Pagani b , Charles H. Fine; Value network dynamics in 3G–4G wireless communications: A systems thinking approach to strategic value assessment, Journal of Business Research, Vol 61, Issue 11, November 2008, Pages 1102–1112

[9]. Maitreya Natu ; Ratan K. Ghosh ; Rudrapatna K. Shyamsundar ; Rajiv Ranjan, "Holistic Performance Monitoring of Hybrid Clouds: Complexities and Future Directions", IEEE Cloud Computing (Volume:3 , Issue: 1), Jan.-Feb. 2016

[10]. R. Calheiros et al., "Workload Prediction Using ARIMA Model and Its Impact on Cloud Applications' QoS," IEEE Trans. Cloud Computing, vol. 3, no. 4, 2015, pp. 449–458.

ISBN: 1-60132-436-7, CSREA Press ©