ANALYSIS AND IMPROVEMENT IN SCALABILITY OF CLOUD-BASED APPLICATIONS

Manjit Kaur[1] (M. Tech CSE), Dr. Gagandeep Kaur[2]
Department of Computer Science
Punjabi University
Patiala, India.

Abstract: A study of the cloud computing provides with an over view that could be considered good approach over the past or the traditional networking systems. We aim at studying the concept of cloud computing which is the new advancement in this era, how it is originated from the grid computing. It provides a lot of services like virtualization and many benefits like high reliability, cost, scalability etc. Scalability is the key feature or the service provided by cloud computing. It is the way to scale your cloud system. There are many types of scalabilities like load scalability, space scalability, space-time scalability, structural scalability are some of them and various methods to achieve these scalabilities like auto scaling, scale the database scalability by load balancing are some of them. Scaling is not just to scale the system whenever or how much u required but various factors lies behind scaling the system. A system should scale the in such a way that it also fulfill the needs of the users and also may not affect the performance and cost of our system. Thus making it essential to achieve the scalability in the best possible way, it becomes an important factor to work upon. Moreover, impact of scalability on the performance of scalability and methods will be designed to improve the cloud-based scalability. We have five existing methods to achieve this cloud-based scalability. We hybrid two of them i.e. thresholding method and learning queuing theory to produce the best result and for proper resource utilization during dynamic scaling and cost optimization.

Scalability is the key service provide by the cloud computing environment. It is the way to scale our system according to our requirements here we study the importance of scalability in cloud environment, its effects on the performance of cloud based applications. We will study the various types of scalabilities and existing methods of achieving these scalabilities and try to find out the way to improve them.

- ***To study and analyze the importance of scalability for various services in cloud environment.***
- ***To study the impact of scalability on the performance of cloud-based applications.***
- ***Designing ways to improve scalability for cloud-based applications.***
- ***Quantitative Evaluation of the scalability***

I. INTRODUCTION

The concept of computing comes from the grid computing, public computing and SaaS. It is the new methods that share the basic framework in the cloud computing environment the data is stored and maintained on the web based environment rather than physically stored in the home of the user. Cloud Computing is the new kind of computing model is coming. It is an extend of changing with needs, that is to say the manufacturers provide relevant hardware software and services according to needs that user put forward. It provides secure, quick, convenient data storage and net computing services cantered by internet. Dynamic expandability is refers to virtualization, so it describes highly scalable computing resources provided as an external service via the internet on a pay-as-you-go basis. The cloud is simply a metaphor for the internet, based on the symbol used to represent the worldwide network in computer network diagrams. *Scalability* can be horizontal or vertical depending upon the situation we can choose any one. Certain algorithms are there to decide which type of scalability is required. Besides all these this automatic scalability or you can say dynamic scaling have some problems yet to be solved. As we have already mentioned that we can add or remove any number of machines from the data center. This feature is commonly known as auto-scaling or Dynamic Scaling. There are certain techniques to achieve this feature in a best possible way, even then there are certain problems like cost factor and wastage of resources. Keeping these factors in view we try to find some more close ideas for the optimization of these two factors. There are five existing techniques to achieve dynamic scaling like Static threshold-based policies, Reinforcement Learning, Queuing theory, Control theory, Time-series analysis. In the Dynamic Scalability it is possible to add resources like CPU/RAM to any vm and all these can be possible with the help of vmware or xenserver. We can add or remove vm or vm resources from the cluster but if the requirement increases above the virtual machines in the cluster then we have refuse or say no to the requesting end, so we can add or remove vm or resources to or from the specified cluster. In the earlier thresholding technique when the load increases it directly scale up the system in which what happen some of the applications logout or leave the server without doing any job so in this way there is the wastage of resources so to overcome this problem we hybrid the two algorithms i.e some concepts of queuing theory and the thresholding techniques together, in which already running applications are supposed to wait for a very short time and the free machines and resources are allocated to the some new requesting applications and then only the resources or virtual machines are added in the server pool from the cluster of virtual machine only for the long running applications. This all can be resources cost can be managed properly.

II. IMPLEMENTATION OF EXISTING TECHNIQUE

Dynamic Scaling aims to scale the cloud infrastructure automatically when it is required. There are five main techniques in "Auto-scaling Techniques for Elastic Applications in Cloud Environments" by Tania Lorido-Botran, Jose Miguel-Alonso, Jose A Lozano or methods to achieve this goal but here two techniques one is Static Threshold Based Rule and another one is Queuing Theory are considered.

STATIC THRESHOLD BASED RULES

if x >thrUp for vUseconds then
n = n + s and
do nothing for inUp seconds (1)
f x <thrDown for v Down seconds then
n = n - s and
do nothing for in Down seconds (2)

In this algorithm they have set the threshold value on the basis of which the required action scale in and scale out will be performed. In this algorithm n is the total number of virtual machines in the resource pool s is the amount of virtual machines to be allocated or deallocated, whereas thrUp is the slighter up value of exact threshold, thrDown is the lower threshold value. This value is being checked for some seconds up in down seconds. In this algorithm when the threshold value crossed it directly adds the machine and vice versa. Sometime some of the applications stops without performing any task or without using newly added machine at that there is wastage of a machine and cost increases. To overcome this problem concept of queuing theory and threshold value is hybridised.

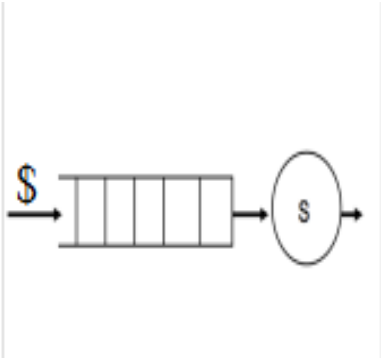
QUEUING THEORY

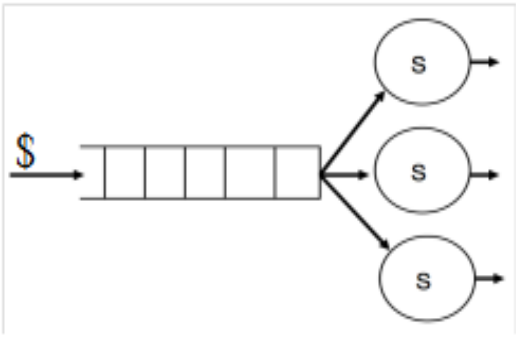
Classical queuing theory has been extensively used to model Internet applications and traditional servers, in order to estimate performance metrics such as the queue length or the average waiting time for requests. The main thing included in Queuing Theory is following:

A. System capacity or queue length: It refers to the maximum number of customers allowed in the system including those in service. When the number of application is maximum, further arrivals are turned away. If this number is omitted, the capacity is assumed to be unlimited, or infinite.

B. Calling population: The size of the population from which the customers come. If this number is omitted, the population is assumed to be unlimited, or infinite. When the requests come from an infinite population of customers, we have an open queuing model, whereas a closed queuing model is based on a finite population of customers.

C. Service discipline or priority order: The service discipline or priority orders that jobs in the queue are served. The most typical one is FIFO/FCFS (First in First Out/First Come First Served), in which the requests are served in the order they arrived in. There are alternatives such as LIFO/LCFS (Last in First Out/Last Come First Served) and PS (Processor Sharing). Useful performance metrics can be estimated, including the arrival rate $, the inter-arrival time, the average number of requests in the queue or in the whole system, the average time waiting in the queue, and the service time. There are two main approaches to obtain those metrics: analytic methods (usually valid for simple models), and simulation (that can be applied to more complex scenarios). Analytic methods are only available for relatively simple queuing models, with well-defined distributions for arrival and service process (e.g. standard statistical distributions such as Poisson or Normal). A typical formula in this context is the Little's Law. It states that the average number of customers (or requests) E[N] in the system is equal to the average customer arrival rate _ multiplied by the average duration of each customer E[T]. The formula is as follows: E[N] = $* E[T]. A similar definition for the Little's Law says that the average queue length can be calculated as the product of the mean arrival rate and the average waiting time in queue.





Queue can be serving to the single server or it can be to the multiple server as there are many servers in one data center. In the literature, both simple queuing models and queuing networks have been used to model applications. For example, general Internet applications have been usually formulated using simple queuing models, and even a cloud infrastructure can be modeled as a G/G/N queue, in which the number of servers N is variable. The model is used to estimate different parameters, for example, the necessary resources required for a given input workload $, or the mean response time for requests. In this way Queuing Theory works in the cloud scenario we will extract some of the concepts from this Queuing model for our proposed algorithm.

III. PROPOSED METHODOLOGY

The main goal of the dynamic scaling is to scale up or scale

down the cloud infrastructure according to the requirement of the application or when the number of applications increased or decreased.

fig 1 Schametic representation of proposed methodology

Fig: 1. Represents the schematic view of the proposed methodology.

A. Queuing and Prioritizing the Application

Dynamic scaling functioning starts when we have applications which will enter to the environment for their tasks to be performed. When applications came to the infrastructure or the executing environment are stored in the queue (array list) along with their arrival time. Priority to the applications will be assigned on the basis of their arrival time i.e they are served on the FCFS basis

Checking Application Status: Some of the applications occupy virtual machines which are initially in the server pool. Some of them waiting for their turn. It is checked which applications are running and which are not. They can be named as new and old applications which can be determined from their CPU utilization time.

Set Threshold Value: Threshold value is the constraint according to which we it decide that system should scale in or scale out. Any performance matrix like CPU utilization can be considered on which threshold is set.

Virtual Machine Management: CPU utilization is the performance matrix which is taken as the threshold value. If CPU utilization increases from the threshold value then following steps are performed:

- Store the name of the running processes named them old processes in the queue and the new processes arriving in the queue.
- When threshold value increases then let the old processes to wait in the queue for "t" secs and the free machines will be allocated to the new processes.
- Some of the processes may complete their task in "t" secs and some may destroyed or leave the pool due to some reason and some will continue.
- The processes that continue, the virtual machines are added only for those processes.
- After adding machines the old processes in the waiting queue will came back to the pool automatically after "t" secs.
- If this condition fails means threshold value decreases then the required number of machines are removed from the pool.
- This process will be recursive.

B. Steps of Proposed Algorithm

In this algorithm shown above there are four different queues Cqu which will store the all the coming applications along

1. *create Cqu(contain all coming applications)*
2. *Create namequ (store old applications) have CPUt >0*
3. *Create name1qu (store new processes) have CPUt<0*
4. *Create C1qu(contain move to waiting state)*

If thr > z

then swap c1qu=name for t secs(5 sec)

vms empty(now)

now if CPUt of name1qu>t1 sec(4.50sec)

then V=v+s

(1)

swap name= C1qu

else V=v-s

(2)

with their information like application id, arrival time, C1qu that store the applications that we let to wait for some seconds, namequ is the queue that store the old applications, name1qu that store the new applications. Thr is the threshold which can be set over any performance matrix z (ex. Z can be CPU load). V is the total number of machines in the server pool now and v is number of machines before any action i.e scale in or scale out and s is the number of machines requires be allocating or reallocating.

IV. RESULTS AND DISCUSSIONS

The major factors which affect the quality of the cloud infrastructure during dynamic scaling the system as the requirement of the cloud based applications increases or decreases during executing environment.

No Resource Wastage: This reflects that when to scale in or scale out the system there should be no wastage of resources i.e no resource should remain free, free resources must be send out of the pool.

Cost: more the resources remain idle and more will be the cost. So the Dynamic process affects the cost of the infrastructure as well.

User Satisfaction: If the request of the user is fulfilled in the right way and on time directly saying it affects the response based on any application. Based on these factors we will evaluate our case studies:

A. How the Applications Run and Reaches to the Threshold Value:

evaluation of the result of Applications run on the sever and after the load of five Application the CPU load increases to 75% which we set as a threshold value to decide to scale in or scale out our system. The graph that when it met to the thresholds value it is unable to enter the application in the server pool or we can say cannot assign resources or virtual machine to the Application. So we need to scale our system.

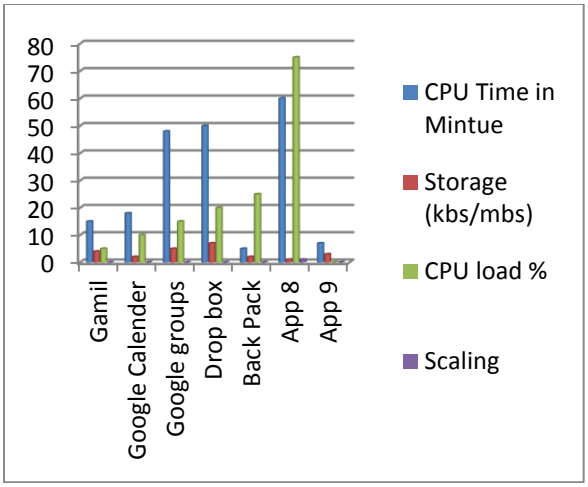


Fig. 1: Impact of Application arrival on Threshold

B. Evaluation of Arrival of New Long Run Applications in Place of old Applications:

The following graph evaluates the Applications running only for very short time on the server just only to decide which applications are long term applications. App 9 and App 13 and App 14 are not exceeding that time limit so we discard these Applications and will not add resources for these applications so the machines will not be added for these applications.

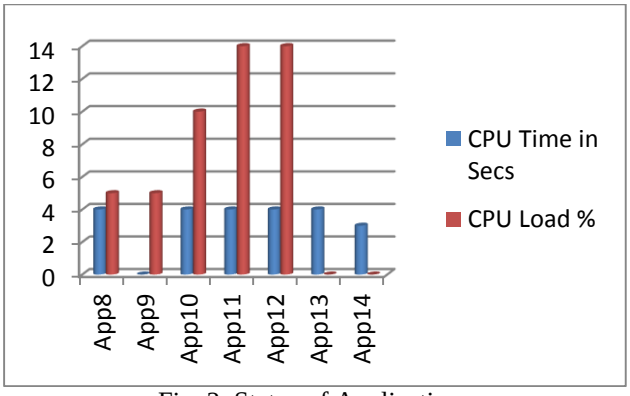


Fig. 2: Status of Applications

C. Evaluation of Total Applications Running on the Server Pool after Scale in:

These are the applications that are running now after scale in process in the server pool and the above processes will be repeated for these applications on the server pool again when needed.

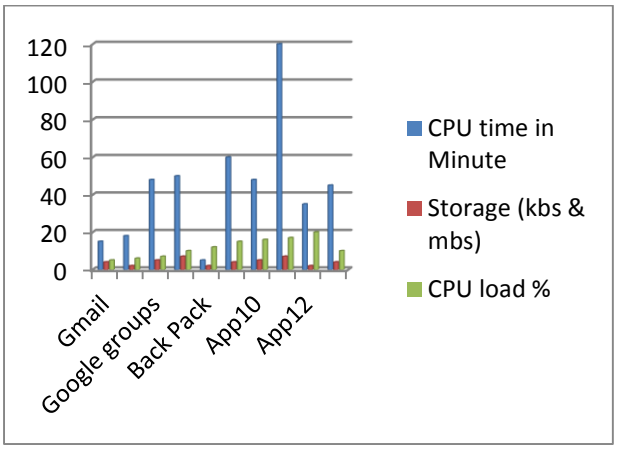


Fig. 3: Status of Applications after Scale in Process

D. Evaluation of Scale down Process:

There may be the case when there are only few applications running or some application leave the pool or the few applications running on the pool and the threshold value decreases needs to remove the virtual machine from the server pool.

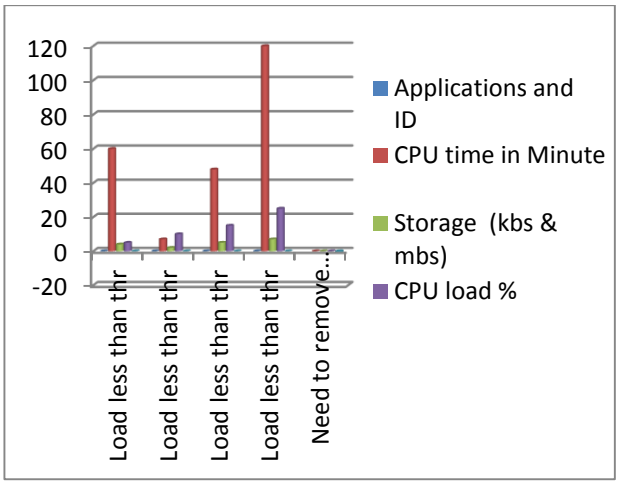


Fig. 4: Status of Applications after Down Scaling

In this way hybridising the queuing theory and static threshold algorithm as depicted in the above graphs when threshold value increases and it need to scale the system so instead of scaling the system directly as in static threshold algorithm, the old running applications are put into a waiting queue for "t"secs and the those empty machines are allocated automatically to the new processes. The main reason of doing so is to determine the applications which cross this time limit machines are added only for these applications in his way there is a proper utilization of the resources, no machine wastages and cost can be optimized.

V. CONCLUSION AND FUTURE WORK

We have analyzed and tried to improve the scalability in cloud based applications. Virtual machines can be added or removed

from the server pool according to the requirement of the applications based upon the threshold value. If the threshold value increase then the machines are added only for the long running applications for the proper utilization of resources means machines are added only for the long running Applications as some applications discard without doing any task and in this way if there is proper utilization of resources no machine remain is added useless cost is decreased hence it affects the optimization and user satisfaction. In future the value of threshold can be varied to study its impact on the dynamic scaling of cloud based applications. Moreover different attributes or feature of applications can be considered which may affect the degree of scalability

VI. ACKNOWLEDGMENT

The satisfaction and euphoria that accompany the successful completion of my thesis would not be complete without mention of people who made it possible because success is epitome of hard work, undeterred missionary, study, zeal, unperturbed concentration and above all guidance. The idea of presenting this material without adequate thanks to those who pointed me in the right direction seems simply indefensible. I owe my special thanks to Dr. Gagandeep, Department of Computer Science & Engineering, Punjabi University, Patiala, who helped and guided me for this work. Her encouraging remarks from time to time greatly helped me in improving my skills. I am highly grateful to Dr. Rajesh Kumar Bawa, Head, Department of Computer Science & Engineering, Punjabi University, Patiala, for providing necessary facilities and comfortable environment to accomplish this task in time. Above all I render my gratitude to the Almighty who bestowed me self-confidence, ability and strength to complete this work. I also owe sincere thanks to my parents and friends for their support and encouragement.

REFERENCES

[1] André B. Bondi: AT&T Labs, "Characteristics of Scalability and Their Impact on Performance", Proceedings of 2nd International Workshop on Software and Performance, pp. 195-203, 2000.
[2] Divyakant Agrawal, Amr El Abbadi, Sudipto Das and Aaron J. Elmore, "Database Scalability, Elasticity and Autonomy in the Cloud", DASFAA (1), 2012.
[3] Erich Knoor, Galen Gruman, "What Cloud Computing Really Means", infoworld.com, pp. 1-7, June 2010.
[4] Fabrice Troilo, Xavier Blanc, "Patterns for Scalability in the Cloud", University Bordeaux, 2010.
[5] Galen Gruman and Erich Knoor, "What is Cloud Computing Really Means", infoworld.com, April 2008.
[6] Jonathan Kupferman, Jeff Silverman, Patricio Jara, Jeff Browne, "Scaling Into the Cloud", University of California, 2011.
[7] J. Srinivas, K. Venkata Subba Reddy, Qyser, "Cloud computing basics", ijarcce.com, pp. 344-346, July 2012.
[8] Kim-kwang Raymond choo, "Cloud Computing Challenges and Future Direction", journal of Trends and Issues no.400, pp. 1-7, October 2010.
[9] Luis M. Vaquero, Maiklinder, "A Break in Cloud: Towards a Cloud Definition" Published in ACM SIGCOM, pp. 50-54, Vol. 39, January 2009.
[10] Luis M. Vaquero, Luis Rodero Merino, Rajkumar Buyya, "Dynamically Scaling Applications in the Cloud", Published in IEEE journal, Vol. 41, January 2011.
[11] Michel Armbrust, Rean Griffith, "A View of Cloud Computing", in communication of ACM, pp. 54-57, Vol.53, April 2010.
[12] Manjula K A, Karthikeyan, "Distributed Computing Approaches for Scalability and High Performance", International journal of engineering science and technology, Vol. 2(6), 2010.
[13] Rajkumar Buyya1, Rajiv Ranjan and Rodrigo N. Calheiros, "Modeling and Simulation of Scalable Cloud Computing Environments and the Cloud Sim Toolkit: Challenges and Opportunities", IEEE, pp. 1-11, 2008.
[14] R. Buyya, C. S. Yeo and S. Venugo bal, "Market Oriented Cloud Computing: Vision Hype and Reality for Delivering IT Services as Computing Utilities", IEEE/ACM Grid cong, pp. 23, 2009.
[15] R. Anandhi, K. Chitra, "A Challenge in Improving the Consistency of Transactions in Cloud Databases – Scalability", IEEE, Vol.52, No.2, August 2012.
[16] Shuai Zhang, Shufin Zhang, xuebin chen, "Cloud Computing Research and Development Trends", IEEE computer society, pp. 94-102, 2010.
[17] http://en.wikipedia.org/wiki/Cloud Computing.
[18] Tania Lorido-Botran, Jose Miguel-Alonso, Jose A. Lozano "Auto-scaling Techniques for Elastic Applications in Cloud Environments", pp. 3-36, 2012.

