International Journal of Engineering and Advanced Technology (IJEAT)
ISSN: 2249-8958 (Online), Volume-8 Issue-3, February 2019

An Experimental Evaluation of Running Cost Analysis for Web Application on Cloud Using Queueing Model

N.Neelima, B.Basaveswar Rao, K.Gangadhara Rao, K.Chandan

Abstract--- This paper explores the dynamic behaviour of the Running Cost when Web Application is hosted on the cloud using M/M/S/K queueing model through experimental study. As a part of earlier work the analytical model for Running Cost was proposed and numerically illustrated with respect to virtual machine operation cost and web request holding cost. There is a need to validate this model in real time scenario by conducting a experimental study to design the experimental setup and Test bed for conducting experiments on Amazon EC2 using Elastic Bean Stalk. The performance measures like end-to-end response time, utilization of computing resources consumed, blocking probability, average time spent in the system and average waiting time in the queue are considered to assess the dynamic nature of the cost model. Experimental results are compared with analytical results, it is observed that there is no significance difference between both the results.

Keywords: Cloud computing, Queueing Theory, Performance Measure, Running Cost

1. INTRODUCTION

Cloud Computing (CC) has become very popular during the last several years because, this technology has largely evolved to process and store large data volumes of web and for accessing distributed computing resources and it delivers utility oriented IT services on "pay-per-use" basis [1]. The Cloud has a large pool of resources and provides a development platform which can be reconfigured vigorously to adjust to a variable load for better performance and for optimum utilization. Commercial providers such as Amazon, IBM and Microsoft, all offer environments for developing and deploying applications in the cloud.

Aaron Orendorff has predicted global retail ecommerce sales will be $4.5 trillion by 2021 [2]. The business-to-business electronic commerce is the marketing, selling, and distribution of products from one business to another through an online or digital portal. This business-to-business sale dominates business-to-consumer sales. The web application hosting is playing the important role to promote the business applications and services. The web application hosting mainly dependent on elastically allotted resources and application deployment cost. The cloud providers are introduced different cloud services to host the web application and supporting tools, for example Amazon's EC2 [32], S3 [3], Hadoop [4],Cloud Store [5], and Sector/Sphere [6]. The Software as a Service type (SaaS) such as a web application service are considered which there is number of application services are provided in the data center. These services are based on SLA, cloud resources are provided to customer in restricted or unrestricted mode [33]. Profitable Cloud Computing platform depends upon its capacity to deliver guaranteed Quality of Services (QoS) and assessing the performance by measurements, simulation or by demonstrating is becoming important Apart from all these QoS parameters there are other performance indicators like response time, job blocking probability, possibility of on the spot provision, and imply wide variety of commitments inside the cloud [7].All these performance measures can be computed by applying varied queueing models [8], which has been proved successfully in various walks of life where the issue of service provider and consumer comes into the picture. When a user directs a request to a web application on cloud, the appeal join the queue if the servers are busy and waits for the service, then it is advanced on to VMs which act as servers to process the requests in the queue. Queueing models offer the analyst with a commanding tool for designing and evaluating the performance of cloud computing in terms of cost. The cost includes deploying cost and running cost for the web application live in 24/7 on cloud. Deploying cost is one time task and is static, whereas running cost is dynamic and depends up on the load and number of virtual machines currently running. There is a need to study and forecast the running cost for the performance of web application with minimum cost as well as better usage of cloud resources. This paper focuses on these issues and gives bird-eye-view of the running cost for the cloud vendors. Several researchers have done research on the analysis on infinite queues [12], [13], [15], [16], [17], [18], [19] which in reality most of the networks follows finite queues like M/M/S/K. Some of the researches done Experimental analysis on Amazon AWS cloud for various attacks against cloud computing services [30], [31]. In the real scenario the running cost plays an important role for online web application hosting on the cloud. When the web application is utilized by the clients, the running cost dynamically changes according to the user requests , capacity of the virtual machines completes the user tasks for a period of time and the number of virtual machines which are operated elastically for a period of time .

Manuscript published on 28 February 2019.
* Correspondence Author (s)
N.Neelima, Assistant Professor, Velagapudi Ramakrishna Siddhartha Engineering College,A.P, India.
B.Basaveswar Rao, Computer Centre, Acharya Nagarjuna University Guntur, A.P, India.
K.Gangadhara Rao, Professor, Department of CSE, Acharya Nagarjuna University Guntur, A.P, India.
K.Chandan, Professor, Department of Statistics, Acharya Nagarjuna University Guntur, A.P, India.

To study the running cost dynamics an experiment is conducted on AWS based on analytical queueing model presented in [23].The substantial performance metrics are end-to-end response time, utilization of computing resources consumed, blocking probability, average time spent in the system and average waiting time in the queue are considered. The total Cost derived based on the performance metrics that are analysed with four different scenarios is also verified through experimental test bed. These results provide a systematic approach and in advance to provide an idea about the running cost to the cloud vendors. So this paper is a forecast analysis of the maintenance of the web application on cloud

The rest of the paper is organised as follows, section 2 discusses about related work, and in section 3 the Overview of the Architecture is presented, the experimental Test-Bed depicted in section 4 and Running Cost model explained in section 5 follows results and observations are in section 6 finally conclusions are given in section 7.

2. RELATED WORK

Some little work has been done on analysis of performance of cloud computing infrastructures as experienced by users from different parts of the world. With the increasing usage of cloud computing, more and more users are looking for answers to questions on which type of cloud service they should choose and from which vendor**.** Amazon's EC2 platform [32][3] provides an economical substitute for certain types of cloud computing research. In this section certain related work for performance analysis for cloud computing using Queueing models and Amazon EC2 based experimental studies are discussed. Garfinkel [9] investigated the performance of different cloud services provided by Amazon and their interoperability. The author measured the data throughput and transactions-per-second (TPS) for read/write operations between S3 and altered end-hosts located either in EC2 or at distant places. In [10] a queueing model for performance study of web applications on a Cloud Environment with IaaS cloud platform is modelled as multiple queues and the virtual machines VMs are modelled as service centers. The web applications running on the Cloud platform were analyzed to determine the dynamics of the significant parameters and research was conducted for validation of the model by running the web applications on amazon EC2.In[11]Results of performance analysis of Cloud Computing Services for Many-Tasks Scientific Computing on four commercial Cloud Computing platforms have been reported and in [12] performance measurement of scientific applications on Amazon EC2 has been carried out .Performance Estimation of Amazon EC2 focusing on non functional properties has been described in [13].Simulation of Scalable Cloud Computing Environments, its Challenges and Opportunities with Cloudsim are discussed in [14].In [15]the performance and storage costs of running the Montage workflow on EC2 are detailed by Deelman et.

In addition Modelling of a Cloud Computing platform for the performance analysis has been done by few researchers. Brebner and Liu proposed a Service Oriented Performance Modelling Technology for modelling the performance and scalability of Service Oriented Applications architected for a variety of platforms in [16]. In [17] The performance of queueing system is calculated analytically to accomplish performance factors like mean number of tasks in the system with single task arrivals and a task request buffer of infinite capacity using Cloud centre as an [(M/G/1) : (∞/GD)] queueing system. In [18] a different cloud computing model with emphasis on the improvement of allocation of resources dynamically following request dependent strategy under non homogeneous condition with time dependent arrival of jobs which is greatly beneficial for evaluating the cloud more effectively and efficiently to increase performance measures of cloud. In [19]they proposed a queueing model M/M/m and studied the performance of the optimization and the parameter for evaluating the service in cloud computing and developed a synthetically optimization method to optimize the performance which results in less wait time, queue length and more customers gaining the service.. In [20-22] the author proposed a model to evaluate the performance of cloud computing centre using G/M/s queueing model. The author noticed that when the arrival rate grows, the length of queue also rises, and the waiting time of a customer increases linearly with the arrival rate. All these work concentrated on calculating the QoS measurements through analytically, by simulation and conducting experiments but not concentrated on cost point of view .In this study to analyse the dynamic cost behaviour using Queueing models with experiments conducted on real time.

3. OVERVIEW OF ARCHITECTURE

The Elastic beanstalk is one of the web services of AWS to host web services, since Elastic Bean Stalk is pre-configured with multiple services these would not be any explicit server configuration overhead besides that it has benefits like powerful customization, and cost-effectiveness. This service automates the setup, configuration, and provisioning of other AWS services like EC2, RDS, and Elastic Load Balancing to create a web service and also creates a fairly standard configuration for a modern rails application. Amazon does not close for any of the properties like auto scaling, elasticity etc.

AWS Elastic Beanstalk decreases management complexity without limiting choice. For using Elastic Beanstalk the user has to create an application and upload the application in the form of an application source bundle (for example a java .war file) in to Elastic Beanstalk and provide necessary information about the application . After that Beanstalk automatically create an environment and configures the resources required to run an application code.

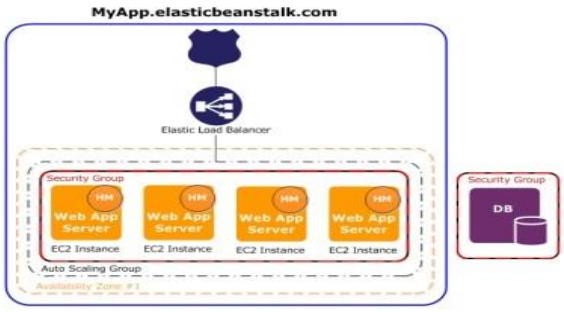

Figure 1: Elastic Beanstalk architecture for a web server

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)

Figure 1 shows the sample architecture of web application hosting in the Elastic Beanstalk and it is adopted from AWS [24]. This data centre contains Route 53, internal load balancer, web app servers which configured with auto scaling and storage server [25].

Route 53 is one of the scalable and highly available web service provided by Amazon intended for the purpose DNS. Using Route fifty three users can connect to the cloud resources like Amazon EC2, Amazon S3 for managing the traffic globally [29] .Load balancer distributes traffic among the running instances in the current environment .When you enable load balancer with Elastic Beanstalk console; it by default creates an application load balancer for your environment without disturbing the application's overall request flow [26]. The auto scaling group consists of amazon EC2 instances which are organised to run web apps on the platform that you choose. Auto scaling automatically starts extra instances to house the growing load on your application and automatically decrease the instances when the load on the application reduces. The Amazon EC2 instances are the part of a security group which states the firewall procedures for the running instances [27].

Amazon offers a variety of cloud storages which is used for stacking and retrieving applications according to user requirements. Amazon simple storage service is one type of storage which is used to store and retrieve any amount of data at any time from the internet. It provides the user with the ability to access the highly scalable, reliable, fast and inexpensive data storage. Amazon S3 stores infinite amount of data to a bucket in the form of an object .Each object can be up to 5 TB. Each object can store and access the data using a unique developer generates key. Amazon S3 provides authentication procedure to keep the data safe from unauthorized users [28].

4. EXPERIMENTAL TEST-BED

AWS Elastic Beanstalk is one of the hyped services to host the web applications on the elastic cloud and it supports several platform configurations for different web programming applications, including multiple versions of programming languages with the application server. AWS cloud provides several tools for hosting and monitoring the web applications and Elastic Beanstalk. This paper presents java a web application to host on Elastic Beanstalk on the AWS. Figure 2 shows the experimental test-bed design on AWS cloud. This design contains micro ec2 instances, elastic load balancer, RDS database server, route 53, Elastic Beanstalk and client requests.

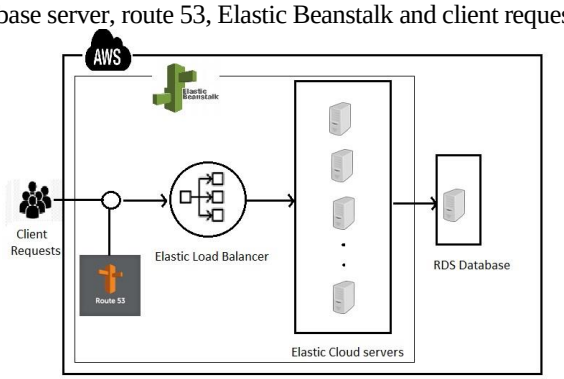

Figure 2: Experimental architecture for hosting java application in the cloud

Figure 3 shows the steps to host java application on the Elastic Beanstalk. Before to get start deploy java applications on AWS Elastic Beanstalk, it need to configure an application source bundle to deploy to an environment. When create an environment, Elastic Beanstalk allocates all of the AWS resources needed to run a scalable web application. To launch an environment with sample application go to Elastic Beanstalk console and choose web server, web server environment or worker environment, platform, app code, environment name, domain name , platform configuration, VPC, tier, instance type, root volume, key pair, IAM role, internal Amazon RDS database, Load balancer.

Set up a Java development environment to test an application locally prior to deploying it to AWS Elastic Beanstalk. This is nothing but install and configure all java application supported servers and tools. To host the java application install java development kit, web container, downloading libraries, AWS SDK for Java, IDE or text editor and AWS toolkit for eclipse.

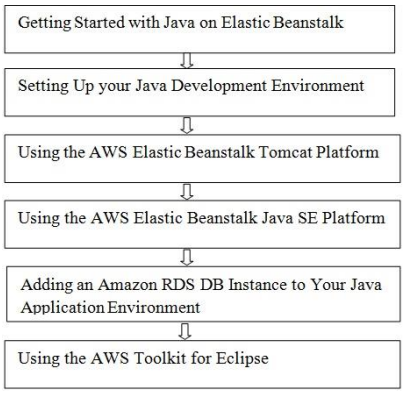

Figure 3: Experimental setup design process

The AWS Elastic Beanstalk Tomcat platform is a well-known environment configuration for Java web applications that can run in a Tomcat web container. Every configuration parallels a major version of Tomcat, like Java 8 with Tomcat 8. To launch the tomcat server in beanstalk, first of all configuring tomcat environment after that choose JVM container (Initial JVM heap size and a maximum JVM heap size). Setup the log option Instance profile and enable log file rotation to Amazon S3 and set the environment properties using tomcat configuration namespaces, bundling multiple WAR Files for tomcat environments. Assembling a WAR file with a shell scriptbuild.sh is a very modest shell script that compiles Java classes, constructs a WAR file, and copies it to the tomcat web apps directory for local testing. Inside the WAR file, you find the same structure that exists in the src directory in the former example, without the src/com folder. The jar command spontaneously creates the META-INF/ MANIFEST.MF file. Extending the Default nginx Configuration in the Elastic Beanstalk's default nginx configuration, add .conf configuration files to a folder named .ebextensions/nginx/conf.d/ in an application source bundle. The Elastic Beanstalk nginx configuration contains .conf files in this folder automatically.

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

The AWS Elastic Beanstalk Java SE platform is a set of environment formations for Java web applications that can run on their own from compiled JAR file. For Java SE platform configurations on Elastic Beanstalk, which provides a limited platform precise options in addition to the regular options it provides for all environments. These options let configure the nginx proxy that runs in front of your application to serve static files. In the SE environment configurations are log options, static files, environment properties, the aws:elasticbeanstalk:container:java:staticfiles namespace, configuring the application process with a procfile, building jars on-server with a build file and configuring the reverse proxy. Amazon Relational Database Service (Amazon RDS) DB instance to store data that application gathers and modifies. The database can be attached to web application environment and managed by Elastic Beanstalk, or created and managed externally. To connect to the database, add the appropriate driver JAR file to application for that add the JDBC driver and connect to Java SE platform, tomcat platforms and troubleshooting database connections. The AWS Toolkit for Eclipse integrates AWS Elastic Beanstalk management features with tomcat development environment to facilitate environment creation, configuration, and code deployment. The toolkit includes support for multiple AWS accounts, managing existing environments, and connecting directly to instances in environment for troubleshooting. Import the AWS web application environment into eclipse IDE and managing Elastic Beanstalk application environments using following configurations: changing environment configuration settings, changing environment type, configuring ec2 server instances, configuring elastic load balancing, configuring auto scaling, configuring notifications, configuring java containers and setting system properties using AWS Toolkit for Eclipse.

5. RUNNING COST MODEL

Several authors explored the dynamical behaviour of web application hosting on cloud through finite or infinite queueing models. In this paper a finite queueing and cost models are adopted based on [23][34]. The cost model is explained and analytically derived in the previous work [23].The Running Cost (RC) model is given below:

Notations:

λ=the arrival rate of user requests,

μ = the service rate of each virtual machine for completion of user request,

S = Number of Virtual machines

K = Buffer size; Total number in system $\leq$ k+s

Lq =Average Customers in Queue

Wq= Average Waiting Time in the Queue

λ_e =Effective Arrival Rate

P_k = Blocking Probability (system is full)

α = Price of request completion resources consuming cost

β = Price for Job waiting cost for requests waiting in the buffer

t = Time in minutes

$$RC = \alpha\, s\, \mu + \beta\, Lq$$
$$= \alpha\, s\, \mu + \beta\, \lambda e Wq$$
$$= \alpha\, s\, \mu + \beta\, (1-P_K)\, \lambda\, Wq \quad (1)$$

Where $Lq = \lambda e Wq$ and $\lambda e = (1-P_K)\, \lambda$

The RC is a sum of two parts, they are cost incurred for the requests actual service completion by the VM's is Operational Cost (OC) and the number of requests waiting in the queue is the cost of waiting cost (WC).The OC is independent of λ and is dependent on S and μ which are elastic by virtue of this elastic character of Bean Stalk. The WC depends upon all the parameters λ, S, μ and K. So RC=OC+WC The mean response time is the total amount of time a request spends in both the queue and in service The objective is this paper is to study the running cost behaviour, the changes in parameter have bearing on the Running Cost, and the same is categorised in to four scenarios. They are λ varying scenarios, μ varying scenario and S varying scenario and K varying scenario. The analysis of these scenarios will help the cloud vendor for cost fore casting . The linear cost model in Eq (1) has been justified by the numerical illustration with different scenarios. For all the scenarios the price of request completion resources consuming cost α and the price for Job waiting cost for requests waiting in the buffer β are taken as 5 and 1 units respectively

λ increasing scenario: A website or a web-application can be opened by a plenty of users at any point of time. It becomes difficult for a web application to manage all these user requests at a stipulated time. Infrastructure must be scaled to handle these requests. Peaks in demand may only last for a short time, sometimes only milliseconds but they have a much effect. This impacts not only the web server and supporting systems but more importantly user response time and it leads to user frustration. The physical resources can be assigned and reassigned to the different virtual machines dynamically based on requests demand and available resources. These scenario gives ideas about how many VM's are scale up / down with optimum cost

μ increasing scenario: As the number of requests that are being served increases, the response time for the customer decreases and the running cost incurred for using the cloud resources decreases. The service rate also has fluctuating because of network problems, bandwidth and the load of the cloud cluster etc. There is a need to identify which service rate gives better performance of VM's for processing of requests with optimum cost

S increasing scenario: The cost incurred generally increases with increase in number of virtual machines running. To measure the elastic property of cloud computing there is a need to study the required VM's for various values of λ, μ and K. It will help to avoid the over provision of VM's as well as under provision of VM's.

K increasing scenario: The user may be blocked or leave the session is two-fold (i) A customer may leave the queue because the queue is very long (ii) he may be rejected when all VM's are busy or the occupancy in waiting queue is full. To avoid the requests blocking, to identify how much queue capacity be needed for varying values of λ, μ, S. This scenario suggests proper value for K with optimum cost.

For this study the following parameters are set up and conducted experiment on AWS, then evaluated the QOS measures like end-to-end response time, utilization of computing resources consumed, and blocking probability, average time spent in the system, average waiting time in the queue and estimated the cost for 4 scenarios.

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

The following Table1 depicted the different scenarios with different parameter values for conducting the experiment.

Table 1: Different parameter values for scenarios of Experiments

Name of the Scenarios	Fixed Parameter values	Varying Parameter
λ increases	S=2,K=10,μ=25,t= 20	(λ) Ranging from 20 to 100 with an interval of 5
μ increases	λ=50,S= 6,K=10,t= 20	(μ)Ranging from 5 to 80 with an interval of 5
S increases	λ= 20,K =50,μ=25,t= 20	(S) Ranging from 5 to 30 with an interval of 5
K increase	λ= 20,S= 2,μ= 25,t=20	(K) Ranging from 2 to 10 with an interval of 1

The experiment is conducted with the geographical origin as United States of America East on Amazon AWS. The web application source bundle in the form of .war file along with queueing model is uploaded in the Elastic Beanstalk. Then Elastic Beanstalk prompts the user to create an environment for running application. This cloud setup was initialised with system perimeters like servers, buffer capacity, capacity of the virtual server and simulation are presented in the table 1. In the first scenario λ fixed input rate is 20 requests per second and variable input ranging from 20 to 100 requests per second representing the rate of the traffic. Second scenario μ fixed with 5 and variable from 5 to 80 representing the service rate of the instances. Similarly third and fourth scenarios S and K are fixed with 5 and 2 and varying from 5 to 30 and 2 to 10. The experiment was started with 5 running instances and 1000 requests per second. In this live process client requests are randomly changed for time t for every 20 minutes.

6. RESULTS AND OBSERVATIONS

The analytical results for the linear cost model of Eq (1) are adopted from our earlier work [23] for different values of λ, μ, S and K. The experiments are conducted for calculation of various performance measures and finally calculated RC as per the given values in Table1.The results are compared with the experimental results conducted on Amazon EC2. The level of significance in difference of these results is shown below for various scenarios.

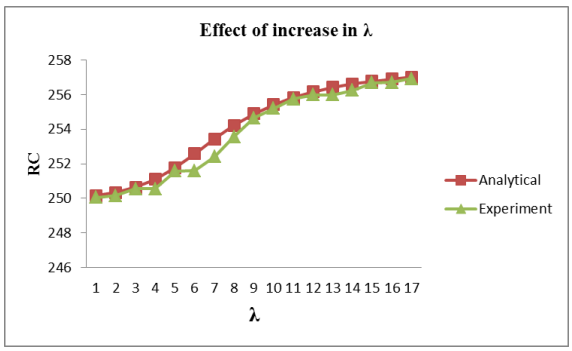

Figure 4: Running cost versus λ

From the above graph it can be observed that for fixed values of μ, S, K as λ increases the running cost also increases because the Wq also increases .In this situation WC increases whereas OC reaches to constant value after S reaches the maximum VM's. The experimental results are very closely associated with the analytical results

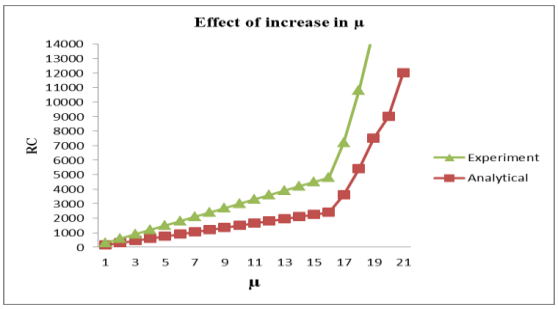

Figure 5: Running Cost verses μ

Figure 5 shows that the Running Cost increases as μ increases because Wq decreases and tends to zero .P_k also tends to zero so OC increases. The results of experimental analysis are slightly above the analytical results when one looked at the graph.

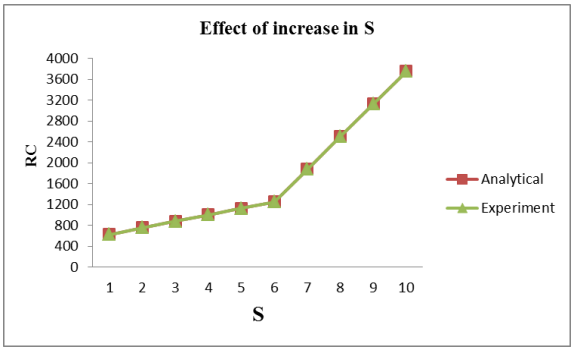

Figure 6: Running Cost verses S

Figure 6 shows that the Running Cost increases as the S increases. As the number of servers increases the waiting time in the queue is negligible and the WC becomes zero. So the Running Cost depends on the OC which increases with the increase of number of instances of virtual instances. The experimental results are very close to the analytical results.

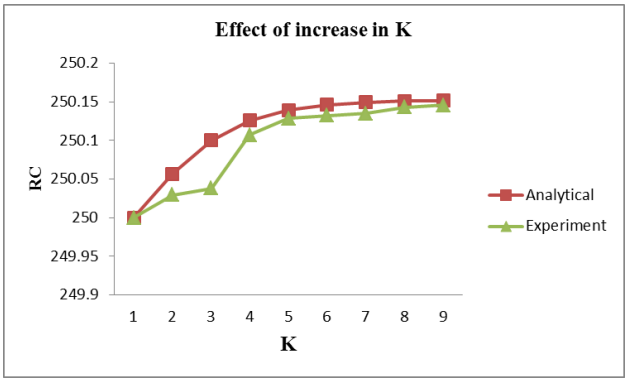

Figure 7: Running Cost estimation verses K

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

From the above graph it is observed that the Running cost increases very nominally and stabilizes when K increases. From this observation it is to conclude that the influence of K is almost minimal. The analytical results are slightly more in some cases when compared to experimental results.

7. CONCLUSION

This paper provides a design of experimental setup for conducting experiments and analysing the Running Cost for hosting Web application on the Amazon cloud .This experimental analysis is useful for cloud vendors to estimate the Running Cost based on four parameters λ, μ, S and K and derived performance metrics which were computed analytically in the in earlier work .The experimental results are on par with the analytical results. In λ, μ, S increasing scenarios the operating cost increases because the power consumption and bandwidth utilization are high, so the Running Cost increases. Where as in K increasing scenario for highest values of K the increase of Running Cost is negligible when compared with other scenarios. From this experimental it is to conclude that the effect of buffer size is negligible and depends up on number of user requests, number of virtual machines in operation and requests completion time. There is a need to study how to derive optimum running cost for various values of these parameters as a future work.

REFERENCES

1. Raj Kumar Buyya and KarthikSukumar, Platforms for Building and Deploying applications for Cloud Computing, CSI Communications, May 2011, Vol. 35. No. 1. pp. 6-11
2. Aaron Orendorff, "Global Ecommerce: Statistics and International Growth Trends", (online) https://www.shopify.com/enterprise/global-ecommerce-statistics, 2017.
3. Amazon S3: https://aws.amazon.com/s3/
4. Hadoop, http://hadoop.apache.org/core/
5. Cloud Store, http://kosmosfs.sourceforge.net/
6. YunhongGu, Robert Grossman, Sector and Sphere: The Design and Implementation of a High Performance Data Cloud, Theme Issue of the Philosophical Transactions of the Royal Society A: Crossing Boundaries: Computational Science, E-Science and Global E-Infrastructure, 28 June 2009 vol. 367 no. 1897 2429-2445.
7. L.Wang, G. vonLaszewski, A. Younge et al., "Cloud computing: a perspective study," New Generation Computing, vol. 28, no. 2, pp. 137–146, 2010
8. L. Kleinrock, Queueing Systems: Theory, vol. 1, Wiley-Interscience, New York, NY, USA, 1975
9. S. L. Garfinkel. An Evaluation of Amazon's Grid Computing Services: EC2, S3 and SQS. Technical Report TR-08-07, Harvard University, August 2007.
10. SwapnaAddamani, AnirbanBasu,"Performance Analysis of Web Applications on IaaS Cloud Computing Platform", International Journal of Computer Applications (0975 – 8887), Volume 64– No.15, February 2013
11. AlexandruIosup, Simon Ostermann, M. NezihYigitbasi, RaduProdan, Thomas Fahringer, and Dick H.J. Epema, "Performance Analysis of Cloud Computing Services for Many-Tasks Scientific Computing", IEEE Trans. On Parallel and Distributed Systems, Vol. 22, No.6. June 2011, pp. 931-945.
12. Keith R. Jackson, LavanyaRamakrishnan, Krishna Muriki, Shane Canon, ShreyasCholia, John Shaif, Harvey Wasserman and Nicholas Wright, " Performance Analysis of High Performance Computing Applications on the Amazon Web Services Cloud". Proc. IEEE Second International Conference on Cloud Computing Technologies Science, 2010
13. VladmirStantchev, "Performance Evaluation of Cloud Computing offerings". Proc. 2009 Third International Conference on Advanced Engineering Computing and Application in Sciences, pp. 187-192
14. Raj Kumar Buyya, Rajiv Ranjan and Rodrigo N. Calheiros1,"Modeling and Simulation of Scalable Cloud Computing Environments and the CloudSimToolkit: Challenges and Opportunities", Proc. of International Conference on High Performance Computing & Simulation, 2009, June 2009. pp. 1-11
15. E. Deelman, G. Singh, M. Livny, B. Berriman, and J. Good, "The cost of doing science on the cloud: the montage example," in Proceedings of the 2008 ACM/IEEE conference on Supercomputing. IEEE Press, 2008, pp. 1–12.
16. Paul Brebner and Anna Liu,"Performance and Cost Assessment of Cloud Services", Proc of Int. Conference on Service Oriented Computing, 2010, pp. 39-50
17. Ani Brown Mary, N and Saravanan, K,"Performance factors of Cloud computing data centres using, [(M/G/1): (∞/GDMODEL)] Queuing system", International Journal of Grid Computing & Applications (IJGCA) Vol.4, No.1, March 2013
18. Satyanarayana A, Dr.P. Suresh Varma, Dr.M.V.RamaSundari, Dr. P SaradaVarma," Performance Analysis of Cloud Computing under Non Homogeneous Conditions", International Journal of Advanced Research in Computer Science and Software Engineering, Volume 3, Issue 5, May 2013 ISSN: 2277 128X.
19. LizhengGuo, Tao Yan, Shuguang Zhao, Changyuan Jiang," Dynamic Performance Optimization for Cloud Computing Using M/M/m Queueing System", Journal of Applied Mathematics, Volume 2014, Article ID 756592, 8 pages
20. Murugesan, R, Elango C, and Kannan S, " Resource Allocation in Cloud Computing with M/G/s– Queuing Model", Volume 4, Issue 9, September 2014, ISSN: 2277 128X, International Journal of Advanced Research in Computer Science and Software Engineering, PP 443-447.
21. Murugesan R, Elango C, and Kannan S," Cloud Computing Networks with Poisson Arrival Process-Dynamic Resource Allocation", IOSR Journal of Computer Engineering (IOSR-JCE) e- ISSN: 2278- 0661, p-ISSN: 2278-8727, Volume 16, Issue 5, Ver. IV (Sep – Oct. 2014), PP 124-129
22. Murugesan R, Elango C, and Kannan S, "Resource Allocation in Cloud Computing with General Classification Time and Exponential Service (G/M/s)", International Journal of Engineering And Computer Science ISSN: 2319-7242, Volume 3, Issue 10 October, 2014 Page No. 8905-8910.
23. N.Neelima, B.BasaveswarRao, K.GangadharaRao, K.Chandan , "Performance Analysis of web Application deployment on cloud using M/M/S/K Queueing model" ,International Journal of Applied Engineering Research ISSN 0973-4562 Volume 13, Number 11 (2018) pp. 9485-9492
24. https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/awseb-dg.pdf#Welcome
25. https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/GettingStarted.html
26. Amazon, Amazon Load Balancer Service. https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/using-features.managing.elb.html
27. AWS Elastic Beanstalk,https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/using-features.managing.ec2.html
28. https://docs.aws.amazon.com/AmazonS3/latest/dev/Introduction.htm
29. Amazon Rote 53: https://aws.amazon.com/route53/
30. Suneetha Bulla, B. BasaveswaraRao, K. GangadharaRao, K. Chandan, "An experimental evaluation of the impact of the EDoS attacks against cloud computing services using AWS" , International Journal of Engineering & Technology7 (1.5) (2018) 202-208
31. F. Al-Haidari, et al. (2015). Evaluation of the Impact of EDoS Attacks against Cloud Computing Services, (Arab J Sci Eng.), Springer
32. Amazon EC2: https://aws.amazon.com/ec2/
33. L. Wu and R. Buyya, "Service Level Agreement (SLA) in Utility Computing Systems", Technical Report, pp. 1-27, 2010
34. Hao-peng CHEN, Shao-chong L1"A Queueing-based Model for Performance Management on Cloud", https://www.researchgate.net/publication/224219108

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)