Computer Science Journal of Moldova, vol.**30**, no.1(88), 2022

The response time analysis of queuing model in cloud computing environment*

Ali Madankan

Abstract

In this paper, the queuing theory is used to model cloud computing environment. The aim of this job is to analyze the response time as a measure of the Quality of Service (QoS) of computer services. In the cloud computing, multi resources need to be allocated simultaneously to multiple customers. When a customer requests for a service, if servers are busy, the requested job enters into the waiting line until a server completes its service. So, this may lead to a bottleneck in the network. By modeling cloud platforms by a queuing network, it can be easy to determine and to measure the QoS. The arrival rate and the service rate of processing servers are two main parameters that can affect the performance of the model; so, they are used to analyze the performance of the model. This paper proposes a queuing model which is applied at multiple servers in order to analyze the response time and also to improve the network performance and QoS effectively in a cloud computing environment.

Keywords: Cloud Architecture, Queuing System, Response Time, Quality of Service.

1 Introduction

Cloud computing has been an emerging technology for provisioning computing resource and providing infrastructure of web applications in recent years. Cloud computing greatly lowers the threshold for deploying and maintaining web applications since it provides infrastructure

* This work was supported by University of Zabol, Grant Ref. No. UOZ-GR-0065538536

as a service (IaaS) and platform as a service (PaaS) for web applications [1]. Consequently, a number of web applications, particularly the web applications of medium and small enterprises, have been built into a cloud environment. Meanwhile, leading IT companies have established public commercial clouds as a new kind of investment. For example, Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides resizable compute capacity in the cloud. It is designed to make webscale computing easier for developers [2]. Google App Engine enables enterprises to build and host web applications on the same systems that power Google applications. App Engine offers fast development and deployment; simple administration, with no need to worry about hardware, patches or backups; and effortless scalability [3]. The cloud providers usually offer high performance, scalability, security, and high availability services. In summary, both of the numbers of cloud applications and providers have kept gradually increasing for a couple of years. As a result, performance managing and guaranteeing the Quality of Service (QoS) have been ones of the most important aspects of clouding computing [8].

The response time is an important characteristics of the system performance, and response time metrics may be a part of service level agreement. This paper considers the response time for services as a measure to evaluate the QoS. The response time is the total time that takes for a job to enter into the system and depart from it. It includes the waiting time before getting the service and the service time. In the architecture of cloud computing systems, it is considered that there are infinite computing resources available on-demand, which allows the resources to be expanded as needed. Response time is an important characteristics of system performance, and response time metrics may be a part of service level agreement.

[4] studied a popular model of cloud computing that consists of services that are entered and delivered from a service center that is accessible from anywhere in the world. The two main components in architecture are the front-end (the gateway) and the back-end (servers). The front-end provides the software components and interfaces that a customer connects to the servers through it. The back-end, which is

the actual cloud architecture, includes a function to manage the jobs queue, the servers and their virtual machines (VMs), and the database system. To avoid database inconsistencies, it is usually considered only one storage (i.e., database) server.

Our focus is on the back-end (which we call cloud). This component of the architecture contains the processing servers and a data service. We considered an entry point for the cloud to contact customers to receive their requested jobs and to send the result through the (Internet) network. Customers' requests are transmitted to the webserver running a service application [5], associated with an SLA (Service Layer Agreement)

Queuing theory is used to model the architecture and to manage the response time for the services. The queued model lets the cloud to be optimally scaled to guarantee the quality of service for the response time. Also this model considers the proper deployment and removal of virtual machines according to the system load. Authors in [6] used queuing model to model the process of entering into the cloud and to schedule and to serve incoming jobs. In that paper, the main problem is to allocate resources in the queuing systems as a general optimization problem for controlled Markov process with finite state space. Authors in [7] studied performance management on cloud using multi-priority queuing systems. In that work, the web applications are modeled as queues, and the virtual machines are modeled as service centers; they could be distinguished into two groups of priority classes with each class having its own arrival and service rates. In addition, the author in [8] showed how to compute performance bounds for the stochastic control policy of Markov decision processes with average cost criteria. In other words, he found bounds on performance with respect to an optimal policy.

An open Jackson network [9], [10] of M/M/m and M/M/1 queues is considered to model the cloud. We have two types of server, processing and database servers. We are also interested in the modeling QoS performance by scaling cloud platforms, leaving aside other issues such as cloud availability [11], energy consumption [12], variability [13] or reliability [14].

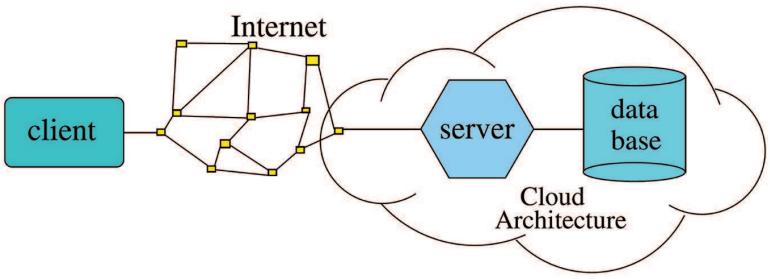


Figure 1. Cloud Computing Environment

2 Model

Let us consider a sequence of M/M/1 and M/M/N, a queuing system model which represents a cloud platform like the model in [6]. The first queue is the entry point that manages incoming jobs and sends them into a server of the second queueing system. Let us call it Input Server (IS). The IS is actually a load balancer. This server is a M/M/1 queue with an exponential arrival and service rates λ and L, respectively. To have steady state, we consider $\lambda < L$. Its aim is to find a server to deliver the job. In order to keep the system in the optimal situation, it uses a distribution algorithm depending on the averaged workload in each server.

The second queueing system, the M/M/N queueing system, has N individual servers, where these processing servers (PS_i), for $i = 1, ..., N$, can serve any incoming jobs. Let consider that each server can serve all accepted job by IS. Also let consider that each server, PS_i, has the same service rate μ, this is $\mu = \mu_i$, $i = 1, ..., m$.

Let us consider there is a database server (DS) in the model, where all PS_i are connected to it and can send their requests to it during the service in the cloud. This kind of access to the databases and directories between virtual machines (VMs) is very important. To avoid data inconsonance, let us consider to have only one database server in this model. Processing servers access to the database server with a

probability σ. DS has all files, dictionaries, databases, and secondary memories and provides these facilities for other servers in the cloud architecture. DS, itself, puts requests into an M/M/1 queue, where it assumes exponential arrival and service rates of $\sigma\gamma$ and D, respectively.

There is another server in this model that is called Outputer Server (OS). In the cloud architecture, jobs come from Client Server (CS); so, the cloud manager has to deliver the completed job to the CS over the network (say internet). The OS receives the responses from the PS_i and sends the responses to CS with an exponential parameter λ.

In the model, the response time (T), which is a very important measure in the cloud architecture, is calculated as:

$$T = T_{IS} + T_{PS} + T_{DS} + T_{OS} + T_{CS}, \tag{1}$$

where each of these terms are the total time that each servers spends to complete its service in this model. In the following, the terms of Eq.1 are calculated.

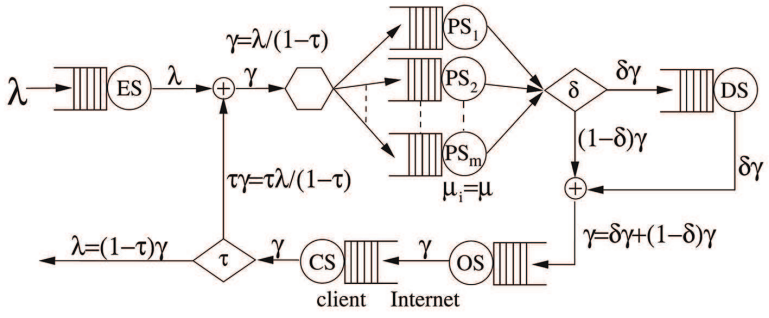


Figure 2. Queuing Model of Cloud Computing

2.1 Inputting Service Time T_{IS}

The Input Server is a load balancer; so, the parameter T_{IS} shows the spent time by Input Server to distribute incoming jobs. Since we modeled the input server (IS) as a M/M/1 queueing system, thus, the

response time for a M/M/1 queue is [16]:

$$T_{IS} = \frac{1}{L - \lambda}, \tag{2}$$

where λ and L are the arrival rate and the service rate of the IS, respectively.

2.2 Processing Server Time T_{PS}

The parameter T_{PS} shows the time spending by Process Servicing to process the job. Since there are N processing servers (PS) PS in the cloud architecture, and these PSs are modeled as an M/M/N queue, then the spent time by this kind of queue is as follows [15]:

$$T_{PS} = \frac{1}{\mu} + \frac{C(N, \rho)}{N\mu - \gamma}, \tag{3}$$

where γ is arrival rate and $\mu = \mu_i$, $i = 1, ..., N$, is service rates of each processing servers. In the (3), the term $C(m, \rho)$ gives the probability of a new coming job to the M/M/N queue. The formula of calculating of $C(m, \rho)$ is defined as [16]:

$$C(m, \rho) = \frac{\left(\frac{(mp)^m}{m!}\right)\left(\frac{1}{\rho}\right)}{\sum_{k=0}^{m-1}\left(\frac{(mp)^k}{k!}\right) + \left(\frac{(mp)^m}{m!}\right)\left(\frac{1}{\rho}\right)}, \tag{4}$$

where $\rho = \gamma/\mu$.

2.3 Database Server Time T_{DS}

In this model, there is a database server that with the probability of σ jobs needs some files from the database server. So, $\sigma\mu$ is the arrival rate to the DS. Also, if the parameter T_{DS} is the total time that Database Server spends to respond, and since the DS is modeled as an M/M/1 queue, so:

$$T_{DS} = \frac{1}{D - \sigma\mu}. \tag{5}$$

2.4 Output Server Time T_{OS}

When process of a job is done by processing servers, Output Server sends the job to the customer. This function takes time that we use the parameter T_{OS} to show the spent time by Output Server (OS). The OS is considered as an M/M/1 queue too. Since this server is connected to the customer over the internet, if F is the average size of jobs and O is the average bandwidth speed of the OS, then the service rate of this part of architecture is O/F. So, the response time of this server is:

$$T_{OS} = \frac{F}{O - \gamma F}. \tag{6}$$

2.5 Client Server Time T_{CS}

The parameter T_{CS} shows the spent time by Client Server to respond. The OS sends the data to CS through Internet and CS receives it. The CS is modeled as an M/M/1 queue, where if C is the average bandwidth speed of the CS, then the service rate is defined as C/F. As a consequence, the response time of CS is:

$$T_{CS} = \frac{F}{C - \gamma F}. \tag{7}$$

3 Result

Regarding the presented model, it is obvious that several parameters are involved in the response time. So, in this section, an analysis of how the response time varies by changing in some of these parameters in the model, is presented. The aim of this paper is to study the expected behaviour of the model when system configurations are modified. In the following subsection, we simulated the model by using the Sage mathematical software to obtain the results.

3.1 Performance Test

To evaluate the presented methodology, Sage mathematical software is used to run this model. We purpose to show the effects of the parameters on total response time (T) by modifying the parameters. We compare the results with real data from a real cloud system with regard to the following parameters:

λ is the arrival rate. Customers send their job requests to the system; so, λ is the average number of incoming jobs to the system. In other meaning, $1/\lambda$ is the average time between consequently incoming jobs.

μ is the service rate, which means $1/\mu$ is the mean service time. For ease of computation, we considered the same service rate for all PS_i servers. In order to analyze the system, we consider the system is not the steady-state, which means the total service rate of the system must always be greater than the incoming rate λ; and, as a result, the system is stable. Also we consider the same service rate for all servers (ES, PS_i, $i = 1, ..., N$, DS, OS, and CS. So, we have $\mu = L = D = O/F = C/F$.

δ is the database access probability. As the model is a web-based system, some incoming jobs need to access the DS and some do not. So, we consider this probability that an incoming job needs to access the DS.

N is the number of PSs that serve requests. Varying in this parameter shows how adding or removing servers affects the system.

F is the average file size. The system sends the result to the CS by files. Files have different sizes which depend on the block size of the web application. This parameter is considered to be the mean size of the files which is not greater than 1MB in most cases.

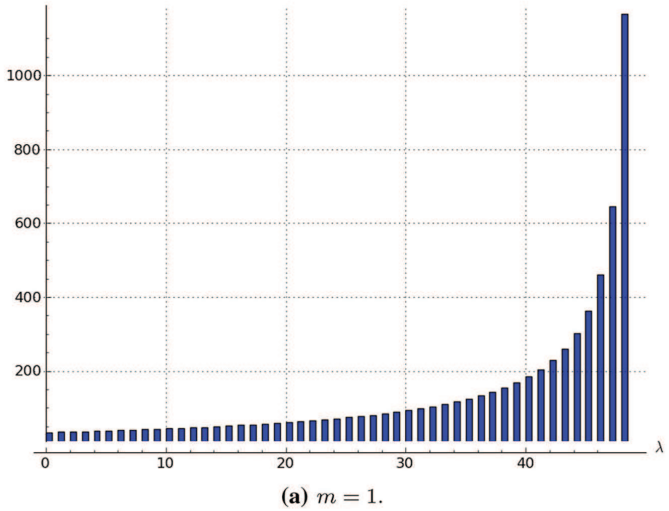
O is the server bandwidth.

C is the client bandwidth. Each client usually has different speed of receiving data from the system and it is usually out of our control.

3.1.1 Response time

First, we tested the response time variation regarding the number of processing servers. Here, in Fig.3, you can see the difference between

the existing one or two processing servers. In Fig.3, it is obvious that increasing the number of processing server has a great effect on decreasing the total response time. By our considerations, the response time becomes fix when the number of servers increases and there is no significant differences when we add more than 5 processing servers. To explain why it happened, we use utilization of the PS_i server, defined as $\rho = \gamma/\mu$, where $\gamma = \lambda/(1-\tau)$. When we add a server to the processing servers, then the utilization rate of the M/M/m queue decreases; and, as a result, the response time becomes equal to the service time. So, this is why we have no improvement in the response time.

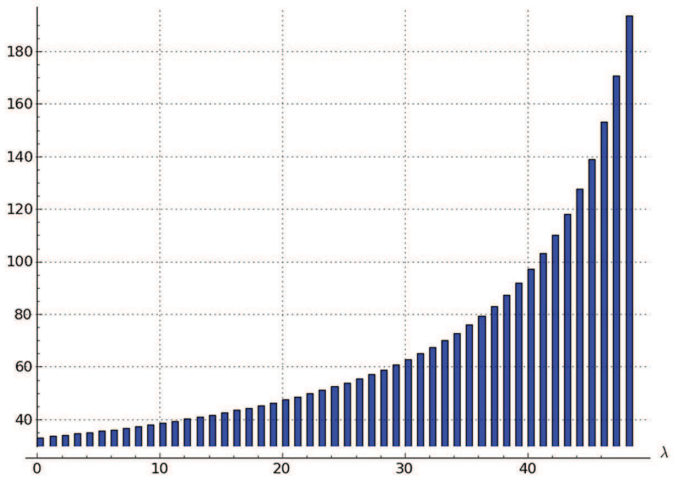


(a) $m = 1$.

Figure 3. The response time with respect to λ

Fig. 4 shows that the response time growth has a direct relation to the utilization of the server: it increases when the server is more utilized. When the utilization is less than 1 and close to 0, it is equal to the service time. When the utilization passes one, the rate of grow of the response time becomes exponential and it increases exponentially with respect to increases of utilization.

The relation between the Total Response Time regarding the mean



(b) $m = 2$.

Figure 4. The response time with respect to λ

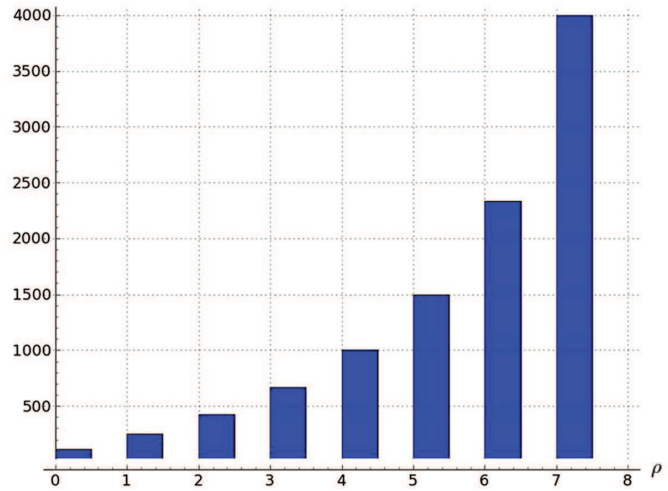


Figure 5. The response time with respect to ρ

file size (F), arrival rate (λ) and the output server bandwidth (O) is shown in Fig.5. It is obvious that the total response time has direct effect from the mean file size (F) and increases exponentially when the mean file size increases. Also it has inverse effect regarding the server bandwidth (O) and decreases when the server bandwidth (O) increases. It is because the OS needs a bigger bandwidth to send the large files to clients over the network. It means files are transferred at low speed when the server bandwidth decreases; and, as a result, the total response time (T) increases.

4 Conclusions and future work

In this paper, we considered a queuing model to design cloud computing architectures to guarantee the quality of service. We used the open Jackson's networks to model the cloud architecture which is the best fit model. Then we used analysis of the Jackson's networks to study the system performance and to analyze the response time.

First, in order to study QoS requirements, a sequence of two queuing systems, M/M/1 and M/M/N, is considered to model the cloud platform. This study showed us, regarding to have good QoS (have proper response time), where a bottleneck can occur in the system and which parameter can solve the problem. This model is an applying model and very useful to tune up the service performance and thus to guarantee the SLA contract between the client and the service provider.

References

[1] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "Above the Clouds: A Berkeley View of Cloud Computing," Electrical Engineering and Computer Sciences University of California at Berkeley, Technical Report No. UCB/EECS-2009-28, February 10, 2009. [Online]. Available:

http://www.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-28.html.

[2] Amazon, "Amazon Elastic Compute Cloud (Amazon EC2)," 2010. [Online]. Available: http://aws.amazon.com/ec2/.

[3] Google, "Google App Engine," 2010. [Online]. Available: http://code.google.com/intl/en/appengine/.

[4] K. Xiong and H. Perros, "Service performance and analysis in cloud computing," in *Proceedings of IEEE World Conference Services*, 2009, pp. 693–700.

[5] J. Martin and A. Nilsson, "On service level agreements for IP networks," in *Proceedings.Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies*, 2002, vol. 2, pp. 855–863. DOI: 10.1109/INFCOM.2002.1019332.

[6] A. Madankan, A. Delavarkhalafi, S.M. Karbassi, and F. Adibnia, "Resource Allocation in Cloud Computing Via Optimal Control to Queuing Systems," *Vestnik YuUrGU. Ser. Mat. Model. Progr.*, vol. 12, no. 4, pp. 67–81, 2019. DOI: 10.14529/mmp190405.

[7] A. Madankan and A. Delavar Khalfi, "Performance Management on Cloud Using Multi-Priority Queuing Systems," in *Advanced Research on Cloud Computing Design and Applications*, 2015, ch. 15, pp. 229–244. DOI: 10.4018/978-1-4666-8676-2.ch015.

[8] A. Madankan, "Performance Bounds and Suboptimal Policies for Multi-Class queue," *Vestnik YuUrGU. Ser. Mat. Model. Progr.*, vol. 12, no. 1, pp. 44–54, 2019. DOI: https://doi.org/10.14529/mmp190104.

[9] J.R. Jackson, "Networks of waiting lines," *Operations Research*, vol. 5, no. 4, pp. 518–521, 1957.

[10] J.R. Jackson, "Jobshop-like queueing systems," *Management Science*, vol. 10, no. 1, pp. 131–142, 1963.

[11] M. Martinello, M. Kaâniche, and K. Kanoun, "Web service availability: impact of error recovery and traffic model," *J Reliab Eng Syst Saf*, vol. 89, no. 1, pp. 6–16, 2005.

[12] A. Beloglazov and R. Buyya, "Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in Cloud data centers," *Concurr Comput Pract Exp*, vol. 24, no. 13, pp. 1397–1420, 2012.

[13] A. Iosup, N. Yigitbasi, and D. Epema, "On the performance variability of production cloud services," in *11th IEEE/ACM international symposium on cluster, cloud and grid, computing (CCGrid'2011)*, 2011, pp. 104–113.

[14] K.V. Vishwanath and N. Nagappan, "Characterizing cloud computing hardware reliability," in *Proceedings of the 1st ACM symposium on Cloud computing (SoCC '10)*, 2010, pp. 193–204.

[15] M. Barbeau and E. Kranakis, *Principles of ad-hoc networking*, New York: Wiley, 2007.

[16] L. Kleinrock, *Queueing systems: theory*, vol. 1, New York: Wiley-Interscience, 1975.

Ali Madankan,

University of Zabol
Zabol, Iran
E–mail: Amadankan@uoz.ac.ir

Received April 16, 2021
Accepted November 21, 2021