Improving Scalability of Cloud System based on Routing Algorithms

Mohamed Eisa
Computer Science Department,
Port Said University, 42526
Port Said, Egypt

E. I. Esedimy
Computer Science Department,
Mansoura University,
Mansoura, Egypt

Alaa Halawani
Applied Physics & Electronics,
Umeå University,
Umeå, Sweden

ABSTRACT
Cloud computing provides end users with computing resources based on virtualization technologies at the data center. This allowed us to optimize data centers utilization by using techniques and algorithms that optimize the use of cloud computing resources. By taking advantage of some useful proprieties of routing algorithm proposed model is presented in the field of cloud computing that makes data centers more flexible and scalable. Our experimental results indicate that proposed model increases utilization of data centers resources and reduce waiting time.

Keywords
Cloud computing; Queuing models; Routing algorithms.

1. INTRODUCTION
Cloud computing services provide end users with services and resources as demand, services may be software resources such as Software as a Service (SaaS), Platform as a Service (PaaS), and Infrastructure as a Service (IaaS).

The exponential growth of computing resources in recent years has created the need for improving service quality and scalability of IT systems. Therefore, cloud computing presented a set of resources to end users as needed. This may be the network resources, storage, processing power, etc. [1]. Furthermore, resources in cloud computing systems are allocated in different locations and different platforms.

In cloud computing, the workload is the amount of processing given to the computer to do processing at some time. So, algorithms for balancing budget and load on the system to make efficient use of resources and improve the response time of the job was designed. Important things that must be considered during the development of this algorithm are: performance and dynamic loading.

Often traditional scheduling techniques and allocation strategies cannot be used in cloud computing, in which the number of end users requests increases and decreases over time in an unpredictable way. This leads to difficulties of analysis and discover of information from incoming requests to distribute the available resources according to user requirements and constraints of cloud provider. Similarly, unpredictable requests due to the increased costs of server load, maximum the total execution time of the task and the difficulty of making an optimal decision in the whole group of tasks [2].

Several approaches are used to distribute the load on the cloud computing system. In these traditional approaches, only a single server, called broker, serves all the entire end users so the overload on that single server increases and affects the system performance [2]. Therefore, dynamic heuristics algorithms are necessary to distribute the load on the cloud system. But some of these methods do not give adequate results when used and the allocation of resources in the system occurred with random method. Min-min algorithm is a type of dynamic heuristic algorithms, where the task with the minimum earliest completion time is scheduled and the procedure continues until all tasks are scheduled [3]. A new version of Min-min algorithm introduced by He. X et al [4] that schedules tasks depend on its bandwidth, where tasks with high bandwidth scheduled before the others. On the other side, Max-min schedule the task with the maximum earliest completion time and then assigned to the corresponding machine. Scheduling algorithm called Resource Awareness Scheduling Algorithm (RARA) proposed by saeed Parsa and Reza Entezari-Maleki to avoid the main drawbacks of the MAX-MIN and MIN-MIN [5]. R.F.Freund et al. [6] presented Min Completion Time (MCT) algorithm that allocates tasks arbitrary to be executed on a resource with minimum completion time. W. Chen [7] presented Heterogeneous – Earliest – Finish – Time algorithm (HEFT) that schedule task based on their priorities and each task is assigned to the resource that can complete the task at the earliest time.

Cui Lin [8] introduces scheduling algorithm that used particle swarm optimization (POS) for scheduling tasks on cloud computing. This algorithm uses computation cost and data transmission cost for scheduling application workflow. Y. Yang et al. [9] proposed an improved cost – based algorithm based on cloud computing systems. It considers resource cost and computation time. Shirazi et al [10] introduce survey of scheduling algorithms that distributed requests to back-end servers. Bryhin et al [11] analyzed and compare load balancing techniques for scalable web servers. Buyya et al. [12] have proposed scheduling policies to address minimum time and cost in the context of Grid computing. K.Mukherjee and G.Sahoo, [13] have given a mathematical model for market-Oriented Cloud Computing. They also have proposed a Bee and Ant colony system based scheduling policy. Qiang Li and Yike Guo [14] have proposed a model for resource scheduling in cloud computing based on stochastic integer programming technique, but none of these papers have considered the concept of server utilization, queue length, and the system response time.

On the other hand, FIFO (First – Come - First - Served) [15] algorithm schedule jobs according to its arrival time, where the earliest job on the waiting queue always executed first. To avoid jobs for waiting long time on the waiting queue, Fair algorithm assigns equal share of resources to all jobs. For real time application virtual machines can be used to schedule jobs based on cloud system as in schedule real-time applications, where virtual machines provide isolation among applications.

For example, Xen provides simplest EDF scheduler to enforce temporal isolation among the different virtual machines.

However, the previous work did not consider the distribution of resources that is able to scale up resources and scale down as demand change. The study takes into considerations load distribution and system utilization. The most important problem is how to build a model that can maximize server utilization and minimize waiting time in queuing models. Therefore, a mathematical model is proposed to deal with multiple tasks and resources based on the basis of maximizing the benefit of the cloud provider and decrease the response time of the system.

The remainder of this paper is organized as follows. Section 2 presents the preliminary and notations. Section 3 deals with the construction of our proposed model. Section 4 describes the experiments conducted by the discrete event simulation and displays the result. Section 5 concludes the paper.

2. PRELIMINARIES AND NOTATIONS

2.1 Cloud Computing

Cloud computing is a provider for services, software and other infrastructures according to customer needs at a specific time. Foster et al [16] presented Cloud Computing architecture that consists of multiple different layers as shown in Figure 1. Cloud computing architecture categories into four layers, which are fabric, unified resource, platform and application, first, the fabric layer is a layer of low-level architecture and has the raw material resources, such storage, processing power, etc. Second, a unified resource layer abstract resources through virtualization, this integrated resource model are subject to the upper layer and end users. Third, the platform layer is based on a uniform layer of resources, and includes an additional set of measurement tools, middleware and services to provide a favorable environment for application development 'and publication. Finally, applications running in the cloud on the application layer [16]. Furthermore, another study by Buyya et al [17] indicates that the cloud architecture is mainly composed of user- middleware, core- middleware and system level, as shown in Figure 2. Figure 2 shows the structure of cloud computing infrastructure based on virtualization that provide more scalability and reliability which add and release VMs based on the varying of cloud system workloads.

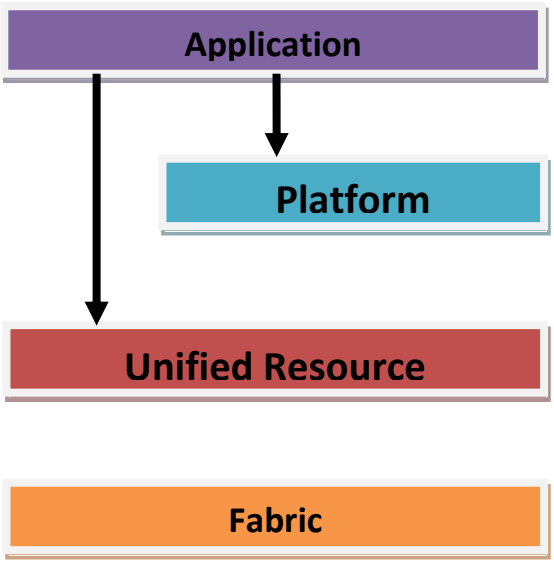

Fig. 1: Four-layer Cloud Architecture [16].

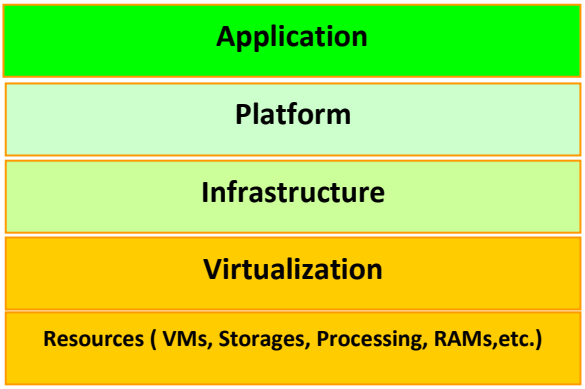

Fig. 2: Cloud Computing based on virtualization.

The layers of cloud computing contain applications that end users can access directly and applications can be made at this layer. In addition, the intermediate layer has a framework that helps developers to create an environment for applications to be developed, deployed and applied at the cloud. The platform layer has services that define the runtime environment based middleware layer for holding and monitoring services at user-level application. Finally, the system level layer, where physical resources such as servers and these resources are managed by the virtualization services set above this layer [17].

2.2 Queuing theory

Queuing theory has become a mathematical tool to deal with different types of queues [18]. Waiting queues are the abstract representation, which aims to identify factors that affect the system's ability to respond to service requests that occurrence at random periods. In general, the models are determined by simple queues in terms of arrival process, service mechanism and waiting queue discipline. Arrival process determines the structure of the probabilistic way service requests occur over time, and the service mechanism describes the number of servers and the potential of the infrastructure over a period of time required to serve the user.

The ultimate goal of the analysis of queues is expected to understand the behavior of the model as a basis for informed and intelligent decisions can be made by management. Thus, the mathematical analysis of the production of models and metrics that used by cloud system, such as the waiting time, the use of the average server, and productivity, as well as the possibility of overflow buffering capacity, and the allocation of time waiting period in server activity, etc.

Queuing system was defined as, $QS = (S, R)$ where S is a set of servers $\{S = S_1, S_2, S_3, \ldots, S_n\}$, R is a finite set of requests $\{R = R_1, R_2, R_3, \ldots, R_n\}$, we assume that the types of requests and Sevres's queue are random, independent, identically distributed and adapted according to their order in the sequence of on a First-Come-First-Served (FCFS).

Also, we assume that the type of request and sever is random, independent and distributed according to (FCFS).The maximum processing time of the cloud server's queue provider can be calculated using the following parameters:

R = Processing time rate,

D = Total demand rate,

c_o = Cost of processing unit, and

n = Number of waiting jobs.

2.2.1 Single- server -queue (M / M / 1 Model)

M / M / 1 model is represented a server with a single queue that has unlimited queue capacity, infinite applications and arrivals are Poisson or random distribution. A is defined as a system queue, Q = (S, R), where S is a set of servers S = {S_1, S_2, S_3, ..., S_N}, R is a limited set of applications {R = R_1, R_2, R_3, ..., R_n}, where the types of applications and server queue system is random, independent and distributed, which has been adapted to users and servers based on their classification in the sequence, on a first-come-first-served(FCFS) basis.

Using mathematical method to calculate formulas for the single waiting queue model with Possion arrivals and exponential service time based on knowledge of the arrival rate and the service rate, which

μ = the mean number of services per time period (the service rate),

λ = the mean number of arrivals per time period (the arrival rate).

The average number of jobs in the waiting queue can be calculated as the following

$$Lq=\frac{\lambda^2}{\mu(\mu-\lambda)} \quad (1)$$

Also, we can compute the probability that an arriving unit has to wait for service

$$P_w=\frac{\lambda}{\mu} \quad (2)$$

and P_w is called server utilization factor or traffic intensity.

The probability that no jobs are waiting in the system

$$P_o=1-\frac{\lambda}{\mu} \quad (3)$$

From the above formulas (2, 3), we can generalize formula to compute the probability of n waiting jobs in the queue by:

$$P_n=(\frac{\lambda}{\mu})^n P_o \quad (4)$$

3. MODEL CONSTRUCTION

Cloud using a variety of techniques for load balancing among servers via cloud broker that accepts requests from end users and distributed it. In general, the proposed model presented a group of routing algorithms that are responsible for receiving all incoming requests from cloud broker and distributed across multiple servers queue. This model is shown in Figure 3 and consists of four modules: multiple queues for incoming requests, routing algorithms based cloud broker, local queues for each local scheduler. Applications (R_1, R_2 ... R_n) from different locations are shown in global queue (GQ), then each application of cloud agent accepts requests from end users and distributed among different servers through a network of local contacts. There are many algorithms for load balancing in the field of cloud computing. We will study the performance and scalability of the scheduling algorithms depending on the cloud computing. We will study the performance and scalability of the scheduling algorithms depending on the corridor of the cloud, as shown in Figure 3. According to the analysis of network behavior of cloud computing with multiple servers and applications service, we can be considered a set of cloud computing features. Routing of the request in the cloud network, i.e., the path followed by the requests among the resources, can be described either probabilistically or according to the following strategies: FCFS, Round robin, Least Connection algorithm and Least Loaded algorithm. These strategies can be further described as follows.

3.1 First Come First Served (FCFS)

This algorithm is simple and fast which jobs are available in the queue as they come. FCFS algorithm schedule jobs according to its arrival time, where the earliest job on the waiting queue always executed first. The implementation of the FCFS policy is easily and managed with FCFS queue.

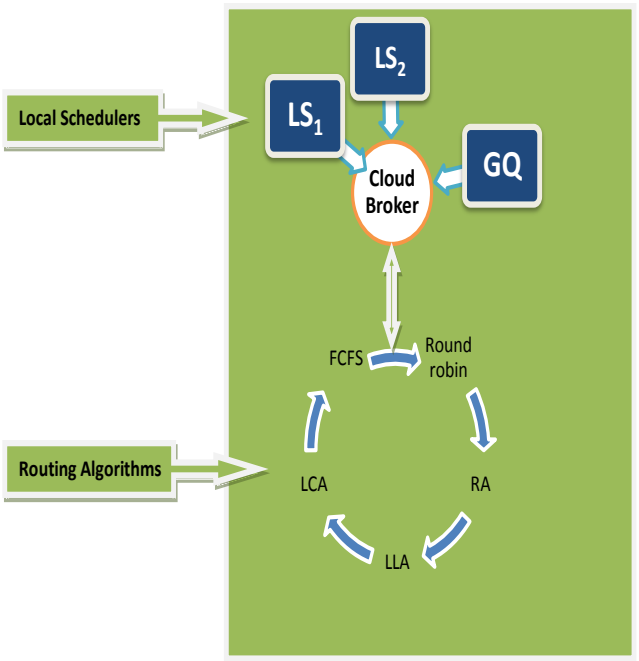

Fig. 3: Proposed Model

3.2 Round Robin algorithm (RR)

This algorithm running according to a circular routing where the first job is sent to the top station, the second job is sent to the central station, and the third job is sent to the bottom station and so on. In planning the round robin, the operations are given a little time and a slice of processor time that is called a time–slice or a quantum. Also, Round Robin algorithm divided processes equally among all processors. In addition, the Round Robin algorithm divided processes among all processors equally [19].

3.3 Least Connection algorithm (LC)

The least-connection scheduling algorithm directs requests to the node with less established connections. This is dynamic routing algorithms that modify routing paths according to traffic loads and network topology change. In addition, you can run dynamic routing algorithm, either periodically or in direct response to changes in the topology or link cost where a group of nodes with similar performance. Contact provide at least it is good to ensure uniform distribution of pregnancy when the applications are very different, as all requests for time did not have the opportunity to address a node [20].

3.4 Least Loaded algorithm (LL)

Routing requests to servers with minimum loaded, or with minimum queue-length, or with the shortest response time. In addition, the algorithm calculates the least loaded the cheapest path between the source and destination using the routing algorithm using global central network in which the algorithm takes the connectivity between all nodes and all link costs as inputs.

3.5 Global VM

Global VM algorithm is able to compute lower and upper waiting time from waiting queues using definitions in 2.2.1. This algorithm have three steps, in the first step it assigns each one of this requests to Local Scheduler(LS) that work based on queuing models methodology. The second step is the core of algorithm where we compute the lower and upper waiting time for each queue. The last step start by mapping the waiting job to routing algorithm. The computation time of Global VM algorithm is $O(kn^2)$ for n waiting queues and k number of requests. The goal of this algorithm is to reduce waiting time and increase server utilization at cloud server by calculating lower and upper waiting time for each queue as presented in Figure 4. The input of the algorithm is a finite set of requests and the output mapping requests based on routing algorithms. Suppose we are given a collection of n jobs that must be executed. To execute the jobs we have m identical machines, $M_1, \ldots, M_m$, available. Supposing that there is a server set $S = \{S_0, S_1, ..., S_{n-1}\}$, W(Si) is the weight of server Si ,N(Si) is the number of server Si connected to cloud broker ,The formal procedure of routing algorithm is as follows:

INPUT: R is a finite set of requests $\{R=R_1,R_2,R_3,\ldots,R_n\}$.

OUTPUT: Mapping Requests based on queue models
//Constructing sampling waiting queues

Create set Q_i by sampling Q/N // N is the number of waiting queues

//Determine equivalence queues based on queue models

Let $S=\{s_1,s_2,.....,s_n\}$ are the available servers

/ /compute the weight of server W(Si)

Read $M(servers)$; $N(requests)$

For (m = 0; m < n; m++)

For (i = m+1; i < n; i++){

if (N(Si) ≤ 0)

continue;

if (N(Si) < C(Sm))

m = i; }

return Sm;}

return NULL;

}

4. EXPERIMENTAL RESULTS

4.1 Experimental *1*

In this section we use simulation studies to investigate the effectiveness of different routing algorithms. We suppose a model for many users who submit requests for execution from large number of sites. At each site, we have developed two elements: Cloud Broker (CB), which determine where to send the jobs sent to this site, and the local scheduler (LS), is responsible for determining the order in which the work is done in this particular site, as shown in Figure 4. We will simulate our based on discrete event simulation model [21, 22]. Requests enter the system and form separate queues randomly for each cloud server. Poisson input flow of customers, and the service time distribution of cloud servers is a second class Erlang. The center consists of 8 performance issue separate cloud server that can accept a limited number of concurrent requests in the execution, that the limited capacity of the region is distributed with the most number of applications. The Global VM to distribute the requests between servers, according to the FCFS algorithm. Our simulation based on eight servers and the previous parameters to illustrate the performance of the cloud system. We first compare the performance between the proposed optimal model, in which the jobs for schedule and computation are allocated optimally by the different routing algorithms, in which the jobs scheduled and allocated equally.

4.2 Experimental 2

The aim of this experiment is to determine the average waiting time and utilization for different servers to access different rates. Average waiting time and the use of a server are important measures to discuss plans for load balancing. In this experiment, we use a standard RUBIS tool [23] to generate the workload for the multi-tier architecture we used. RUBIS simulates eBay, the online auction shopping on the web site in which people and businesses buy and sell a wide range of goods and services all over the world. Thus, clients perform read-only interactions with the site, and the interactions reading and writing interactions that modify the database. We consider a web server with four processors with the workload ranging 200-1200 requests / second. We run RUBIS tool using four different scheduling algorithms, a RR, LC, LL and FCFS. According to the results shown in figure 5.

4.3 The discussion of the results

In this section, we perform simulations to evaluate our proposed model based on different routing algorithms. Table1 shows the queue length, residence time, utilization and throughput for each model based on the same constraints. We first compare the performance between the different routing algorithms, in which the waiting time and utilization for waiting queues and servers are computed by proposed RR algorithm, where the arrival rates and service time for proposed model are allocated equally. Comparison of the queue length, residence time, utilization and throughput between the proposed model and the routing algorithms is shown in Table 1. From table1, we can see that the round robin achieves much lower queue length compared to the least connection and FCFS under the same constraints. Also, the RA allocates a large number of waiting jobs in the computing servers, thus leading to a higher utilization. We next evaluate the system throughput between the routing algorithms in the cloud system. Our goal to study the effective of different workload and services times for routing algorithm in cloud system using cloud broker as shown on table 2. We observe that when increasing of workload and mean service time at servers the round robin is the best over the other routing algorithms where round robin achieve high Utilization , Response time , throughput. On the other hand, when both work load and mean service time also decreased the least connection algorithm is the best where reduce cost of time service and power consuming. In this section, we conduct simulations to evaluate our proposal on the basis of different models of the model queue

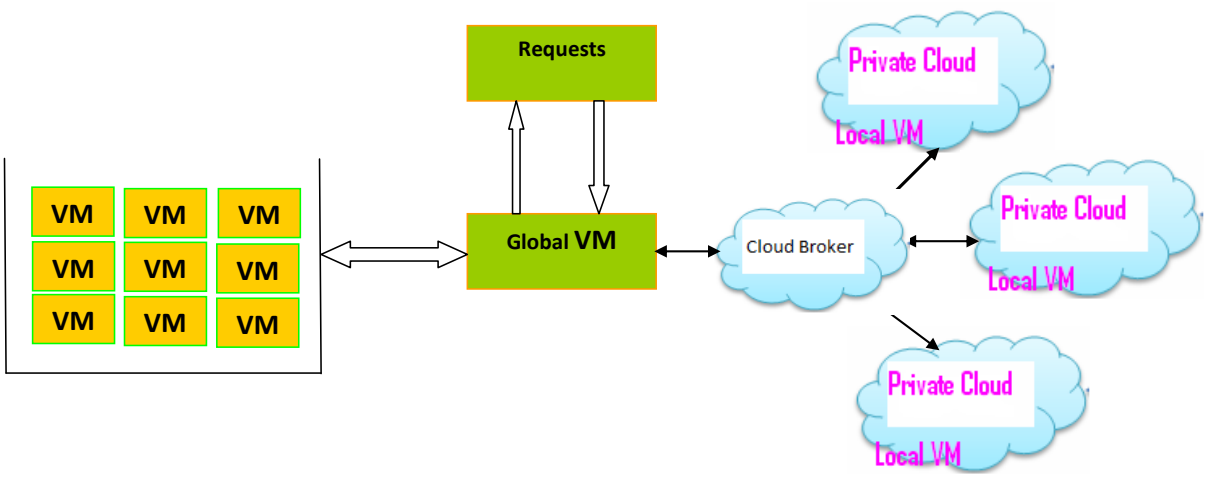

Fig. 4: Global VM based queuing models

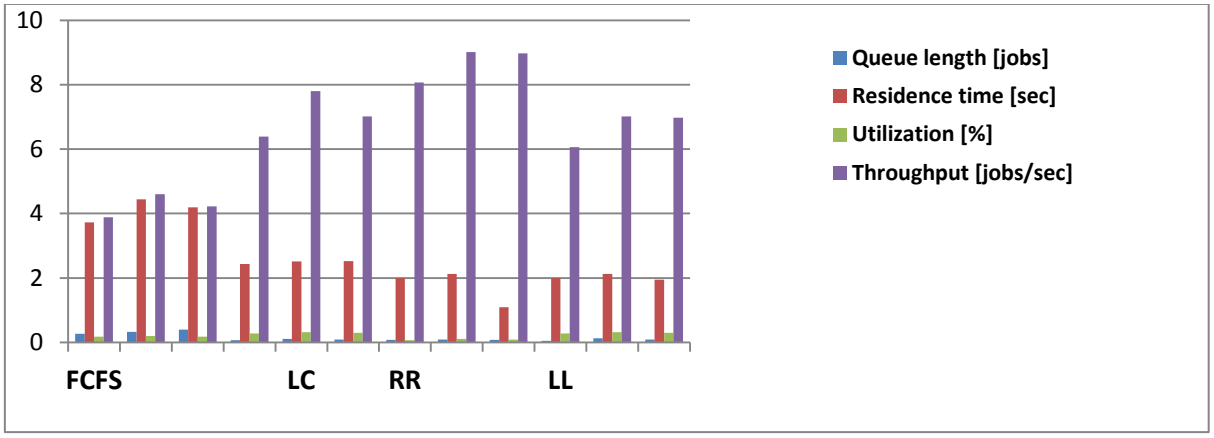

Fig. 5: Average response time for different combinations of scheduling and host utilization using queuing models

Table 1. Comparison between proposed model and queuing models

Algorithm		Queue length [jobs]	Residence time [sec]	Utilization [%]	Throughput [jobs/sec]
FCFS	**Min**	0.263	3.726	0.180	3.886
	Max	0.324	4.437	0.198	4.603
	Avg	0.392	4.192	0.182	4.221
RR	**Min**	0.071	2.436	0.276	6.386
	Max	0.106	2.515	0.318	7.806
	Avg	0.089	2.520	0.297	7.013
LC	**Min**	0.080	1.992	0.071	8.067
	Max	0.084	2.129	0.106	9.014
	Avg	0.076	1.095	0.089	8.973
LL	**Min**	0.053	1.992	0.276	6.067
	Max	0.132	2.129	0.318	7.014
	Avg	0.088	1.950	0.297	6.973

Table 1 shows the queue length, residence time, and the use and productivity for each model based on the same restrictions. You can determine the size of the work specified as a criterion in the evaluation of a computer system in terms of performance (the ease with which the computer handles the workload), which itself generally divided into response time (the time between the user request and in response to a request from the system) and productivity (the amount of work done in time).First, the results between the proposed model and other queuing models were compared to the expected .The use of servers and queues are calculated, where the distribution of rates and access times may serve both the proposed model and routing algorithms. The comparison showed the queue length, residence time, and the use and productivity proposal based on the models of the queuing models and different algorithms in Table 1. We run our simulations are based on eight servers and the previous configuration to illustrate the performance of the cloud system. First, the optimal performance of the

proposed model was compared, jobs are assigned to the area and have optimally by the different models of the queue, where jobs are assigned programming and account for both. Figure 5 shows the comparison between the response times of service limit of a single server and used it in configuring our simulator and investigated the performance of four load scheduling algorithms, namely FCFS, least loaded, least connection, and round robin.

Our investigation shows that:

• Round Robin algorithm leads better, but they need information about the time requirements of each service request. This information is usually not available in realistic simulation, and therefore it is difficult to employ an algorithm basis .Round Robin performs much worse than the other two algorithms to decrease the workload on average. However, when the rate of access to mass increase, and the performance of the three algorithms begin to converge. Performance at the lowest contact approaches the basic algorithm in a much faster speed of the round robin.

• The least connection algorithm contact at least easy to implement and it performs well in the medium and high workloads. However, when the workload is very low, and less time waiting to schedule the connection is much higher than the basic algorithm (2-4 times higher). But, for such workloads absolute gap between low these times of waiting two weeks is very low, and therefore it may still achieve a response time (or deadline) required by the end user.

Table 2. The response time and throughput of proposed system model

	#of jobs	System Response Time [sec]	System Throughput [job/sec]
Min	**1.842**	**19.401**	**0.123**
Max	**3.023**	**20.201**	**0.133**
Avg	**2.801**	**19.851**	**0.114**

From Figure 6, we can see that the proposed model takes less response time than the different queuing models under same constraints

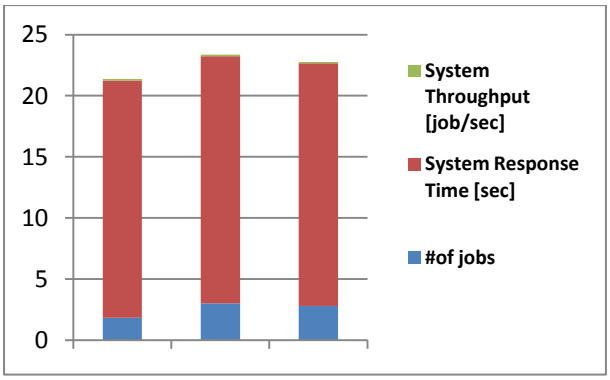

Fig. 6: Average response time and system throughput for proposed model under same constraints

4. CONCLUSION

In this paper, the model presented in the basis of the model queue. Routing incoming requests to the queue with the least amount of work to reduce the workload, response time and the average length of the queue. These results indicate that our model is to increase the use of comprehensive planning and reduce waiting time. The experimental results indicate that the reduction of the proposed model in the world to wait in the field of cloud architecture.

6. REFERENCES

[1] Raytheon UK Targeted in Cloud-Based Attack. Available online: http://www.zdnet.co.uk/news/security-threats/2011/10/12/raytheon-uk-targeted-in-cloud-basedattack- 40094173/ (accessed on: Feb 1, 2014).

[2] Mohamed Eisa, E. I. Esedimy and M. Z. Rashad, Enhancing Cloud Computing Scheduling based on Queuing Models, International Journal of Computer Applications (0975 – 8887) Volume 85 – No 2, January 2014.

[3] T. Kokilavani, Dr. D.I. George Amalarethinam, "Load Balanced Min-Min Algorithm for Static Meta-Task Scheduling in Grid Computing", "International Journal of Computer Applications", vol. 20, no. 2, April 2012, pp. 43-49.

[4] He. X, X-He Sun, and Laszewski. G.V, "QoS Guided Min-min Heuristic for Grid Task Scheduling," Journal of Computer Science and Technology, vol. 18, 2003, pp. 442-451.

[5] Saeed Parsa, Reza Entezari-Maleki, "RASA: A New Grid Task Scheduling Algorithm", "International Journal of Digital Content Technology and its Applications", vol. 3, no. 4, December 2009, pp. 91-99.

[6] R. F. Freund, M. Gherrity, S. Ambrosius, M. Campbell, M. Halderman, D. Hensgen, E. Keith,T.Kidd, M. Kussow, J. D. Lima, F. Mirabile, L. Moore, B. Rust, and H. J. Siegel, "Scheduling resources in multi-user, heterogeneous, computing environments with Smart Net", "7th IEEE Heterogeneous Computing Workshop (HCW '98)", 1998, pp. 184-199.

[7] W. Chen, J. Zhang, "An Ant Colony Optimization Approach to a Grid Workflow Scheduling Problem With Various QoS Requirements", "IEEE Transactions on Systems, Man, and Cybernetics - Part C: Applications and Reviews", vol. 39, no. 1, January 2009.

[8] Cui Lin, Shiyong Lu," Scheduling Scientific Work flows Elastically for Cloud Computing", 4th International Conf. IEEE on Cloud Computing, 2011.

[9] Y. Yang, K. Liu, J. Chen, X. Liu, D. Yuan and H. Jin, "An Algorithm in SwinDeW-C for Scheduling Transaction-Intensive Cost - Constrained Cloud Workflows", 4th IEEE International Conference on e-Science, 374-375, Indianapolis, USA, December 2008.

[10] Shirazi, B. A., K. Krishna, and H. Ali. 1995. Scheduling and Load Balancing in Parallel and Distributed Systems. Wiley-IEEE Computer Society Press. Voas, J., and J. Zhang. 2009. Cloud Computing: New Wine or Just a New Bottle? IT Professional 11:15- 17.

[11] Bryhni, H., E. Klovning, and O. Kure. 2000. A Comparison of Load Balancing Techniques for Scalable Web Servers. IEEE NETWORK 14: 58-64.

[12] R. Buyya, M.M. Murshed, D. Abramson, and S. Venugopal. Scheduling parameters weep applications on global grids: a deadline and budget constrained cost-time

optimization algorithm. Software Practice and Experience, 35(5): 491-512, 2005.

[13] K.Mukherjee, G.Sahoo, "Development of Mathematical Model for Market-Oriented Cloud Computing", International Journal of Computer Applications (0975 – 8887), Volume 9– No.11, November 2010.

[14] Qiang Li, Yike Guo. "Optimization of Resource Scheduling in Cloud Computing", 12th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing, 978-0-7695-4324-6/10© IEEE, DOI 10.1109/SYNASC.2010.8.

[15] Poonam Devi, "Implementation of Cloud Computing By Using Short Job scheduling", International Journal of Advanced Research in Computer Science and Software Engineering, July – 2013, ISSN: 2277-128X

[16] Foster, I. et al (2008) Cloud Computing and Grid Computing 360-Degree Compared. Grid Computing Environment Workshop, GCE '0. 12-16 November, pp. 1-10.

[17] Buyya, R., Ranjan, R., and Calheiros, R. (2009) Modelling and Simulation of Scalable Cloud Computing Environment and the Cloud Sim Toolkit: Challenges and Opportunities. International Conference on High Performance Computing and Simulation, HPCS '09. 21-24 June, pp. 1-11.

[18] An Introduction to Queueing Theory - L. Breuer, D. Baum – Springer Verlag 2005.

[19] Rakesh Mohanty, H. S. Beheram Khusbu Patwarim Monisha Dash, M. Lakshmi Prasanna , "Priority Based Dynamic Round Robin (PBDRR) Algorithm with Intelligent Time Slice for Soft Real Time Systems", (IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 2, No. 2, February 2011.

[20] Radojevic, B. & Zagar, M. (2011). Analysis of issues with load balancing algorithms in hosted (cloud) environments. In proceedings of 34th International Convention on MIPRO, IEEE.

[21] Integrating MATLAB, Simulink and State flow Components in a Sim Events odel:www.mathworks.com/wbnr15638

[22] Averill M. Law, W. David Kelton, McGraw-Hill 2000 Simulation Modeling and Analysis (3rd Edition).

[23] RUBiS. Rubis: Rice University bidding system. http://rubis.ow2.org/, 2010.