Journal of Engineering Science and Technology
Special Issue on ATITES2021, April (2021) 62 - 71

A HEURISTIC ANALYSIS TO MINIMIZE AVERAGE TASK UNIT COMPLETION TIME AND AVERAGE WAITING TIME IN CLOUD COMPUTING ENVIRONMENT

SOUVIK PAL[1,*], MOHAMMED HAYDER KADHIM[2], FARHAD MARDUKHI[3], BIKRAMJIT SARKAR[4], MEDHA RAY[5]

[1]Department of Computer Science and Engineering, Global Institute of Management and Technology, India
[3,4]Department of Computer Engineering and Information Technology, Razi University, Kermanshah, Iran
[4,5] Department of Computer Science and Engineering JIS College of Engineering, India
*Corresponding Author: souvikpal22@gmail.com

Abstract

Cloud computing is a business model or an infrastructure consisting of a pool of physical resources that can be arranged on-demand basis. As the end users are concerned about getting better services from the service providers, this manuscript is using scheduling strategies which in turn facilitate the users with minimization of both task unit completion time (Refers to the time taken by each task unit to complete its task), and the average waiting time (Refers to the average waiting time of the cloud customers). In this paper, we have analysed the simulation results and compared the average task unit completion time. We have also evaluated and compared the performance parameters by means of queuing model.

Keywords: Average waiting time, Cloud computing, Completion time, Queuing model, Virtual machine.

1. Introduction

Rapid usage of internet all over the globe, cloud computing has already been headed in the IT industry [1, 2]. Cloud computing is transforming the computing landscape adapting with instantaneous requirements [3, 4]. Cloud concept and its computing process is the emerging topic in the internet-centric and IT-market oriented business place. When jobs come in, the resources are provisioned. The key problem in cloud computing is job scheduling and waiting time minimization. [5, 6]. In short, we can say that the waiting time refers to the time for which a process waits from its submission to completion, and the completion time refers to the time taken by each task unit to complete its task. This manuscript aims to compare the average completion time of the task units between different scheduling policies and to minimize waiting time using CloudSim simulator and queuing model respectively.

Objective of the manuscript

The CloudSim toolkit, scheduling strategy, queuing model, and average waiting time have all been addressed in this paper. CloudSim 3.0 was used to calculate the average task completion time as a function of the number of task units. In this relation, we have taken a batch of a finite number of task units to be scheduled instead of an infinite number of task units. That has been implemented using M/M/c/K queuing model to reduce the average waiting time which is our primary focus of our manuscript. We chose the M/M/c/K model because service requests and processing times are modeled using a Poisson distribution rate and an exponential distribution (M: Markov Chain) with a finite number of servers (c) and a maximum of K customers in the system for each time case. Therefore, this paper has compared two different queuing system to analyze the average waiting times inside the queue and also into system.

2. Literature Review

In this section, we have discussed about the literature survey of the related study. Mell and Grance [7], have discussed cloud computing definition, characteristics, deployment models and service models. Zhang and Zhou [8] have presented Cloud Computing Open Architecture (CCOA) in their paper. Cloud computing comes into light with the advantage of both Service-Oriented-Architecture (SOA) and virtualization.

Buyya et al. [9] have presented computing technique including IT paradigms as an utility-based service. They have taken the initiative to build up cloudbus, a software system "Aneka" including Software Development Kit, cloud brokerage service capable of deploying prompt services, and energy-aware resource allocation procedure. Buyya et al. [10] have proposed a simulation toolkit CloudSim which helps the cloud developer for simulation and modelling of cloud computing environments.

Calheiros et al. [11] have figured out CloudSim toolkit for evaluation and performance of cloud application. CloudSim toolkit helps developers for modeling different components like virtual machines, data centers and resource allocation techniques. Khazaei et al. [12] have focused on performance analysis of cloud data centers using queuing model.

Garg and Buyya [13] have introduced the Network-key CloudSim's components and their features, as well as extending it with a scalable network for cloud infrastructure efficiency optimization. Bessai, et al. [14] have presented bi-criteria approaches over distributed resources and have taken account of the cost estimation and execution time. They have summarized in their paper as formalization of workflow model for scheduling, minimization of cost execution, monitoring overall execution time.

Prasad and Rao [15] have discussed resource procurement in cloud infrastructure. They have planned a procurement section which may be appropriate for cloud broker to implement cloud optimal algorithm for resource procurement. Church and Goscinski [16] have made a survey on computing solutions which are concerned to mammalian genomics. Their survey is linked to bioinformatics within cloud software services.

Calero and Aguado [17] have developed a service platform for adaptive monitoring purpose. Xia et al. [18] suggested an empirical model for assessing IaaS consistency. They focused on the device overhead rate, rejection likelihood, and request completion time as key quality metrics.

Pal and Pattnaik [19] have adapted Johnson job scheduling algorithm. Resource migration techniques [20], Different scheduling policies [21, 22], techniques for minimizing the waiting time, finding profit optimization methods [23, 24], and server utilization policies in terms of queuing model have been discussed in recent times.

3. Simulation Platform

3.1. Architecture of functional model

The principal components of CloudSim those are associated with our experimental analysis are shown in the Fig. 1. We have presented the correlation between the entities of CloudSim. Users submit their tasks to the broker and broker acts like a dispatcher between data center and user. Data center is correlated with Cloud Information Service (CIS) that registers each data center entry and discovers the resources. A data center encapsulates a set of hosts on which numbers of Virtual Machines (VM) are scheduled. Host models a physical server.

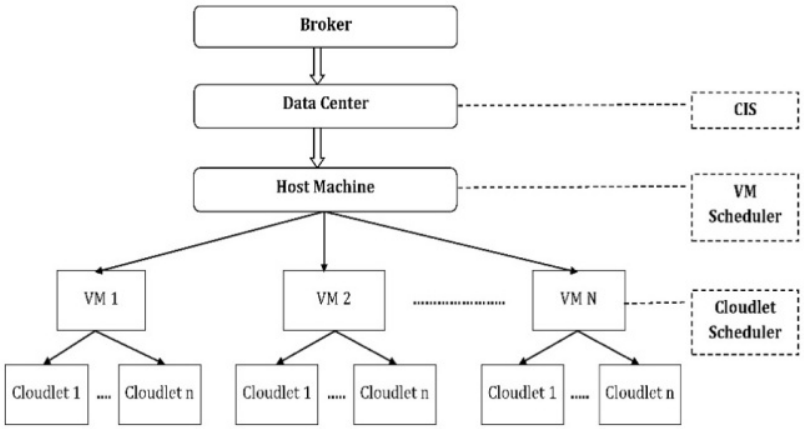

Fig. 1. Structure of Cloudsim functional model.

The host's VM Scheduler simulates the scheduling policies needed for allocating processing cores to virtual machines. Cloudlets, which model cloud-based application services, are executed on VMs. Cloudlet Scheduler assigns various cloudlets to different VMs for processing. Each application service contains a pre-assigned instruction length, and Cloudlet Scheduler implements this instruction for VM [10, 11].

3.2. Scheduling policy

We have focused to find out the minimum average completion time of the submitted task units which result in the performance enhancement of the cloud system. There are two scheduling strategies defined in this section i.e. Time-shared scheduling policy and Space-shared scheduling policy. These scheduling strategies can be applied to both VMs and task units. CloudSim allows two level VM provisioning: firstly, at the host level, and secondly, at the VM level. In first case, it determines the amount of processing power of each processing core required for each VM. And at the second level, each task unit is assigned a fixed amount of processing power by the VM. At each level, space-shared and time-shared policies are implemented in CloudSim. These two policies differ with the performance of the task units or application services. In this section, VM provisioning is presented in Figs. 2 and 3. In this scenario, a host (having two processing cores) receives request for hosting two VMs in such a way that one VM needs two cores and executes four task units. VM1 is assigned to the task units namely T1, T2, T3and T4, and the second one (VM 2) is to host T5, T6, T7 and T8. Tn refers to Task unit, where n= 1 to 8.

SPACE-SHARED POLICY:	
Step 1:	*Accepted task units are arranged in the ready queue.*
Step 2:	*Find out if there are any free processing cores available or not.*
Step 3:	*If available, then task units are put in the execution queue.*
Step 4:	*First task unit is assigned to the hosting VM.*
Step 5:	*After completion of the first task, the next task is assigned.*
Step 6:	*If the queue is empty, it checks for the new task.*
Step 7:	*Then it repeats from the step 1.*
Step 8:	*End*
Note 1:	As each VM requires two cores, only one VM can run at a given instance of time. Only after completion of VM1, VM2 can be assigned the cores.
Note 2:	Each task unit requires only one core.

TIME-SHARED POLICY:	
Step 1:	*Accepted task units are arranged in the ready queue.*
Step 2:	*All the task units are simultaneously assigned to VM. These task units are dynamically context switched during their life cycle.*
Step 3:	*When the queue is empty, it checks for the new task.*
Step 4:	*Then it repeats from the step 1.*
Step 5:	*End*

Note: Each VM gets a time slice on every individual processing core and the time slices are distributed among the task units.

In Fig. 2, VMs are allocated on the basis of space-shared policy and within a VM, task units are allocated to the processing cores on time-shared basis. As mentioned before, since each VM requires two cores, only one VM can run at a given instance of time. VM 1 and VM 2 can't be processed simultaneously. VM 1 will complete its execution first, and then only VM 2 can start its execution. VM 2 can be assigned the cores which are space-shared basis and cloudlets are in Time-shared basis. As a result, during the life cycle of a VM, all of the allocated task units are dynamically context swapped (Time-shared).

In Fig. 3, VMs are allocated in the basis of time-shared policy and within a VM, task units are allocated to the processing cores in space-shared basis. Each VM gets a time slice on every individual processing core and the time slices are distributed among the task units (Time-shared). As the processing cores are shared, VMs are getting fewer amounts of processing power. But, at any given instance of time, only one task can be allocated to each processing core (Space-shared).

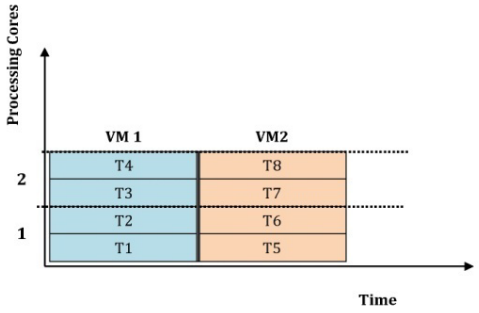

Fig. 2. VM in space-shared and Cloudlet in time-shared.

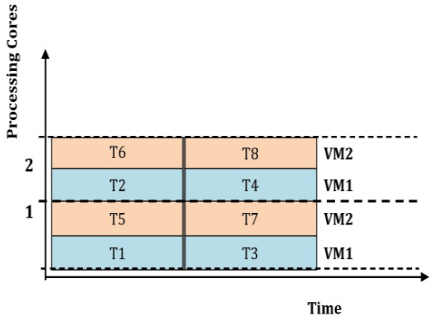

Fig. 3. VM in time-shared and Cloudlet in space-shared.

4. Result Analysis

4.1. Simulation-based result analysis

The key objective of our experiment is to compare the average completion time of the said scheduling scenario and to realize the minimum average completion time using simulation environment [25]. In first case, VMs are allocated in space-shared

policy and task units are in time-shared policy. Secondly, VMs are in time-shared policy and task units are in space-shared policy. After the creation of VMs, the task units are submitted, and VMs and task units are arranged accordingly. During simulation, a data center is designed with features such as x86 architecture, Linux as an OS, and Xen as a VMM.

The simulation environment is made up of two hosts, each with 1000 MIPS (Millions of Instruction per Second), 2 GB of RAM memory, and 1 TB of storage. VM runs inside a host, sharing the host list (the number of hosts present in the simulation) with other VMs. It processes the cloudlets. This processing happens according to a policy (time-shared or space shared), defined by the cloudlet scheduler. The host can submit cloudlets to the VM to be executed. In our simulation consideration, VM ID is initially 0 (Auto incremented depending on the number of VMs), Host IDs are 0 and 1 as the number of hosts is 2. Number of VMs is 20; each of the VM has 1000 MIPS capacity, 512 MB RAM, and 1024 MB storage capacity.

In each VM, Cloudlet scheduler is modeled with Time-shared and Space-shared scheduling strategy. Cloudlet ID is initially 0 (Auto incremented depending on the number of Cloudlets). Length of each Cloudlet is 4, 00000 (the length or size (in MI) of this cloudlet to be executed) (MI: Millions of Instruction). The file size and output size of each cloudlet are 300 (in byte) and also 300 (in byte) respectively. The Number of Cloudlets varies from 20 to 80.

Table 1. Comparison analysis of average task unit completion time.

No. of Cloudlets	**Average task unit completion time (s)**	
	VM in space-shared and Cloudlet in time-shared	***VM in time-shared and Cloudlet in space-shared***
20	720	560
40	1360	880
60	2000	1200
80	2680	1540

We have performed the test experiment according to the above simulation scenario and the comparison of average completion time of the task units is presented in Table 1. Figure 4 shows the comparison analysis of average completion time of task units using the said strategies, i.e., space-shared and time-shared policies. Initially VMs are assigned to the hosts and the incoming cloudlets are scheduled aforesaid policies. The number of cloudlets is generally higher than that of the VMs. Therefore, the number of cloudlets and the scheduling policies of cloudlets within the VM are the considerable issues in our simulation.

According to the graph, if the cloudlets or task units are scheduled in space-shared policy, it will show better outcome in comparison to time-shared policy. While increasing the number of cloudlets in both the policies, the average completion time of the task units is going high in time-shared policy in comparison with space-shared. When the time span of the cloudlet completes and the cloudlet is still executing on the VMs, the scheduler compulsorily pre-empts the cloudlets on the VM and allocates the VMs to the next cloudlet [context switch]. Therefore, increasing number of cloudlets leads too much context switch [for Time-shared policy] and this phenomenon increase the average completion time causing an overall degradation of system performance**.**

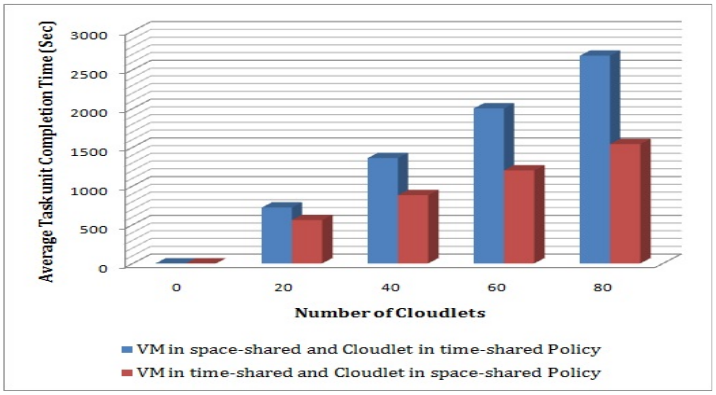

Fig. 4. Comparison analysis of average task unit completion time.

4.2. Queuing model-based numerical analysis

When the number of services requested exceeds the capacity of the service provider, waiting lines or queues will be created. The basic queuing model includes the arrival and service process, as well as the number of servers and the system's maximum capacity [26, 27]. The M/M/c and M/M/c/K queuing models were the subject of this segment. The service request and process time are expected to follow a Poisson propagation rate and an exponential distribution, respectively (M: Markov Chain). The notation used by Kendal [28] denotes the average arrival rate and average service rate, respectively. The number of servers is denoted by c, and the number of customers is limited to K. So, since K is the system's maximum capacity, (K-c) is the system's queue capacity. For both queue models, we found two numbers of servers and eight places of maximum availability. To ensure system stability, an equilibrium state involving the utilization factor must be considered like $\rho = \frac{\lambda}{\mu} \leq 1$. As shown in Table 2, we started with the average arrival rate and the average service rate.

Table 2. Initial parameter (Average arrival rate and service rate [29]).

Λ	μ
20	40
60	70
120	122

According to queuing model, it is denoted as *Ls*, *Lq*, *Ws*, and *Wq*, respectively. Tables 3 and 4 show the comparison study using different queuing model.

Table 3. M/M/c queuing model.

	Lq	*Ls*	*Wq*	*Ws*
λ = 20	0.033	0.533	0.0016	0.0267
λ = 60	0.192	1.05	0.0033	0.0176
λ = 120	0.313	1.297	0.0026	0.0108

Table 4. M/M/c/K queuing model.

	Lq	*Ls*	*Wq*	*Ws*
λ = **20**	0.033	0.533	0.0016	0.0266
λ = **60**	0.186	1.04	0.0031	0.0174
λ = **120**	0.296	1.277	0.0025	0.0107

According to the findings in the tables above, the M/M/c/K queuing model produces better results than the M/M/c model.

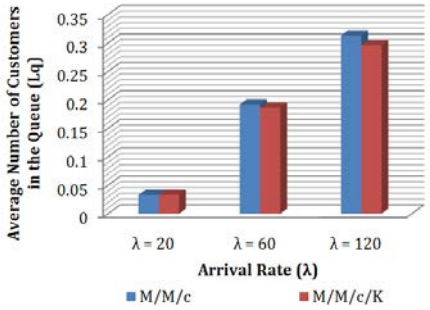

Fig. 5. Graph analyzing in (*Lq*).

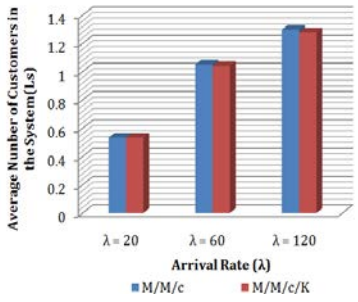

Fig. 6. Graph analyzing in (*Ls*).

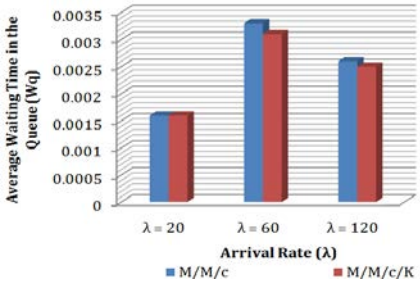

Fig. 7. Graph analyzing in (*Wq*).

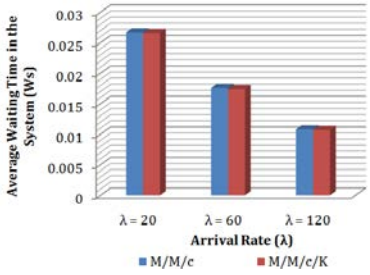

Fig. 8. Graph analyzing in (*Ws*).

The comparative findings are seen in Figs. 5-8, which examine the total number of customers and average waiting time. This M/M/c/K queuing model is more effective than the M/M/c model. These findings allow us to use queuing theory to evaluate and compare performance parameters such as *Ls*, *Lq*, *Ws*, and *Wq*. The first one shows enhanced result in comparison with the later one.

5. Concussion

In this manuscript, we have discussed simulation-based approaches and queuing model to do performance analysis of the service policies. This paper discusses simulation-based methods for ensuring improved service quality by reducing average task unit completion times and reducing average waiting times using the aforementioned queuing model.. In terms of total number of customers and average waiting time, the M/M/c/K queuing model outperforms the M/M/c model. Future research may help validate and refine simulation scenarios to determine the cost per memory unit, bandwidth, storage unit, and other factors.

References

1. Buyya, R.; Yeo, C.S.; Venugopal, S.; Broberg, J.; and Brandic, I. (2009). Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility. *Future Generation Computer Systems*, ACM, 25(6), 599-616.
2. Shaikh, F.; and Haider, S. (2012). Security threats in cloud computing. *6th International IEEE Conference on Internet Technology and Secured Transaction*, 11-14 December, 214-219.
3. Pal, S.; and Pattnaik, P.K. (2012). Efficient architectural framework of cloud computing, *International Journal of Cloud Computing and Services Science*, 1(2), 66-73.
4. Sarathy, V.; Narayan, P.; and Mikkilineni, R. (2010). Next generation cloud computing architecture-enabling real-time dynamism for shared distributed physical infrastructure. *19th IEEE International Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprises (WETICE'10)*. Larissa, Greece, 28-30 June, 48-53.
5. Saha, S.; Pal, S.; and Pattnaik, P.K. (2015). A novel scheduling algorithm for cloud computing environment. *Computational Intelligence in Data Mining*. Volume 1, Advances in Intelligent Systems and Computing, Springer India, 410, 387-398.
6. Pal, S.; and Pattnaik, P.K. (2012). A simulation-based approach to optimize the execution time and minimization of average waiting time using queuing model in cloud computing environment. *International Journal of Electrical and Computer Engineering*, 6(2), 743-750.
7. Mell, P.; and Grance, T. (2009). The NIST definition of cloud computing. *National Institute of Standards and Technology*, 800-145, 1-7.
8. Zhang, L.J.; and Zhou, Q. (2009). CCOA: Cloud computing open architecture. *IEEE International Conference on Web Services*, 607-616.
9. Buyya, R.; Pandey, S.; and Vecchiola, C. (2009). Cloudbus toolkit for market-oriented cloud computing. *International Conference on Cloud Computing (CloudCom)*, LNCS, Springer, 5931, 24-44.
10. Buyya, R.; Ranjan, R.; and Calheiros, R.N. (2009). Modeling and simulation of scalable cloud computing environments and the cloudSim toolkit: challenges and opportunities. *International Conference on High Performance Computing and Simulation* (HPCS '09), 1-11.
11. Calheiros, R.N.; Ranjan, R.; Beloglazov, A.; De Rose, C.A.; and Buyya, R. (2011). CloudSim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *Software: Practice and Experience*, 41(1), 23-50.
12. Khazaei, H.; Misic, J.; and Misic, V.B. (2011). Modeling of cloud computing centers using M/G/m queues. *IEEE International Conference on Distributed Computing Systems Workshops* (ICDCSW), 87-92.
13. Garg, S.K.; and Buyya, R. (2011). NetworkCloudSim: Modeling parallel applications in cloud simulations. *IEEE International Conference on Utility and Cloud Computing (UCC)*, 5-8 December, 105-113.
14. Bessai, K.; Youcef, S.; Oulamara, A.; Godart, C.; and Nurcan, S. (2012). Bi-criteria workflow tasks allocation and scheduling in cloud computing

environments. *IEEE International Conference on Cloud Computing* (CLOUD), 24-29 June, 638-645.

15. Prasad, A.S.; and Rao, S. (2014). A mechanism design approach to resource procurement in cloud computing. *IEEE Transactions on Computers*, 63(1), 17-30.
16. Church, P.C.; and Goscinski, A.M. (2014). A survey of cloud-based service computing solutions for mammalian genomics. *IEEE Transactions on Services Computing*,7(4), 726-740.
17. Calero, J.M.A.; and Aguado, J.G. (2015). MonPaaS: an adaptive monitoring platform as a service for cloud computing infrastructures and services. *IEEE Transactions on Services Computing*, 8(1), 65-78.
18. Xia, Y.; Zhou, M.; Luo, X.; Zhu, Q.; Li, J.; and Huang, Y. (2015). Stochastic modeling and quality evaluation of infrastructure-as-a-service clouds. *IEEE Transactions on Automation Science and Engineering*, 12(1), 162-170.
19. Pal, S.; and Pattnaik, P.K. (2016). Adaptation of Johnson sequencing algorithm for job scheduling to minimise the average waiting time in cloud computing environment, *Journal of Engineering Science and Technology* (*JESTEC*), 11(9), 1282-1295.
20. Pal, S.; Kumar, R.; Son, L.H.; Saravanan, K.; Abdel-Basset, M.; Manogaran, G.; and Thong, P.H. (2019). Novel probabilistic resource migration algorithm for cross-cloud live migration of virtual machines in public cloud. *The Journal of Supercomputing*, 75, 5848-5865.
21. Kheirollahpour, R. Jazayeriy, H.; and Rabiei, M. (2019). A heuristic-based task scheduling method for reducing waiting time in cloud environment. *Iranian Conference on Electrical Engineering*, Yazd, Iran, 1884-1888.
22. Chen, C.-L.; Chiang, M.-L.; and Lin, C.-B. (2020). The High performance of a task scheduling algorithm using reference queues for cloud-computing data centers. *Electronics*, 9(2), 371.
23. Jeyalaksshmi, S.; Nidhya, M.S.; Suseendran, S.P.; and Akila, D. (2021). Developing mapping and allotment in volunteer cloud systems using reliability profile algorithms in a virtual machine, 2nd *International IEEE Conference on Computation, Automation and Knowledge Management* (ICCAKM), Dubai, United Arab Emirates, 97-101.
24. Tsai, J.-F.; Huang, C.-H.; and Lin, M.-H. (2021). An optimal task assignment strategy in cloud-fog computing environment. *Applied Sciences*, 11(4), 1909.
25. Shannon, R.E. (1998). Introduction to the art and science of simulation. *Winter Simulation Conference*, 7-14.
26. Tadj, L. (1996). Waiting in line. *Potential, IEEE*, 14(5), 11-13.
27. Cheng, C.; Li, J.; and Wang, Y. (2015). An energy-saving task scheduling strategy based on vacation queuing theory in cloud computing. *Tsinghua Science and Technology*, 20(1), 28-39.
28. Kendall, D.G. (1951). Some problems in theory of queues. *Journal of the Royal Statistical Society: Series B*, 13(2), 151-185.
29. Li, L. (2009). An optimistic differentiated service job scheduling system for cloud computing service users and providers. *IEEE International Conference on Multimedia and Ubiquitous Engineering*, 295-299.