Research article

UDC 519.872:519.217
PACS 07.05.Tp, 02.60.Pn, 02.70.Bf
DOI: 10.22363/2658-4670-2021-29-2-158-172

To analysis of a two-buffer queuing system with cross-type service and additional penalties

Irina A. Kochetkova[1, 2], **Anastasia S. Vlaskina**[1], **Dmitriy V. Efrosinin**[1, 3], **Abdukodir A. Khakimov**[1], **Sofiya A. Burtseva**[1]

[1] *Department of Applied Probability and Informatics*
Peoples' Friendship University of Russia (RUDN University)
6, Miklukho-Maklaya St., Moscow, 117198, Russian Federation
[2] *Institute of Informatics Problems, FRC CSC RAS*
44-2, Vavilova St., Moscow, 119333, Russian Federation
[3] *Johannes Kepler Universität Linz*
69, Altenberger Straße, Linz, 4040, Austria

(received: May 20, 2021; accepted: May 25, 2021)

The concept of cloud computing was created to better preserve user privacy and data storage security. However, the resources allocated for processing this data must be optimally allocated. The problem of optimal resource management in the loud computing environment is described in many scientific publications. To solve the problems of optimality of the distribution of resources of systems, you can use the construction and analysis of QS. We conduct an analysis of two-buffer queuing system with cross-type service and additional penalties, based on the literature reviewed in the article. This allows us to assess how suitable the model presented in the article is for application to cloud computing. For a given system different options for selecting applications from queues are possible, queue numbers, therefore, the intensities of transitions between the states of the system will change. For this, the system has a choice policy that allows the system to decide how to behave depending on its state. There are four components of such selection management models, which is a stationary policy for selecting a queue number to service a ticket on a vacated virtual machine each time immediately before service ends. A simulation model was built for numerical analysis. The results obtained indicate that requests are practically not delayed in the queue of the presented QS, and therefore the policy for a given model can be considered optimal. Although Poisson flow is the simplest for simulation, it is quite acceptable for performance evaluation. In the future, it is planned to conduct several more experiments for different values of the intensity of requests and various types of incoming flows.

Key words and phrases: queuing system, cloud computing, Poisson flow, parallel queues, optimal policy

1. Introduction

In order to preserve and protect the users confidential data of computing resources, the concept of *Cloud Computing* was developed as a way to provide secure storage and processing of data for companies and individuals. Cloud Computing includes not only programs and applications delivered as services over the Internet, but also the hardware and system software in the data centers that provide those services. This technology has five main characteristics [1]: on-demand self-service, broad network access, resource pooling, rapid elasticity and measured service. In addition, Cloud Computing includes three main types of services: Infrastructure as a Service, Platform as a Services, and Software as a Service [2]. There are four different ways to use this technology: Public Cloud, Private Cloud, Community Cloud, and Hybrid Cloud.

Nowadays the Cloud Computing model has taken on an increasingly prominent role in a variety of IT-environments, where service providers seek to meet the needs of their customers and improve their competitive position. The increase in the number of users and the expansion of the services provided has led to the need for more storage space. As a result, service providers must work to increase the bandwidth of online data centers. Cloud Computing has become an integral part of maintaining high performance to improve competitiveness [3]. It is the fastest growing technology, and therefore, there are some challenges for developers and for those who use them. Let's consider some tasks:

— tasks of distribution and use of resources;
— model of calculations MapReduce (model of parallel computing over very large amounts of data) [4];
— protection of cloud infrastructure [5];
— ensuring the reliability of the work of many servers [6];
— homomorphic codes (a form of encryption);
— identification of spam pages [7];
— organization of information search.

The problem of optimal resource management in the Cloud Computing environment is described in many scientific publications. As known, one of the approaches to solving this problem is the construction of Queuing Systems (QS). To analyze the distribution of resources and develop an optimal method of performance management, in [8] a multiservice QS of a Cloud Computing model with the same type of tasks and identical servers is investigated. The optimization criterion is the minimization of the ratio of the average queue length to the number of lost tasks. It should be noted that the efficient operation of such a network presupposes the ability to flexibly respond to changes in the demand for computing power by turning on/off machines. Therefore, for a heterogeneous environment of Cloud Computing virtual machines the open Jackson queue network model was proposed in [9], which allows solving the problem of scaling the number of virtual machines. To solve this problem the architecture of an elastic system of dynamic resource management with several queues is presented in [10]. The model of an open system with message queues is presented in [11], where reliability is guaranteed due to the mechanism for optimizing the timeout duration, which does not allow the loss of a single message. The cloud architecture on e-health platforms in medical centers was studied in [12], where a model of two sequential $M/M/s$ queues is proposed: the first assumes receiving services for registration, data verification and consultation, and the second is for accessing the cloud database if the serving server is free. Here, it is possible for tasks to go into orbit, i.e. repeated calls if the service server is busy, and to leave the system due to impatience. The proposed model reduces the overall waiting time by 25% compared to the existing model, and also increases resource efficiency.

We will analyze a two-buffer queuing system with cross-type service and additional penalties. System model describing the trajectory of movement of customers is described in Section 2. Part 3 presents a multidimensional Markov chain. Experimental evaluation of the model is presented in Section 4.

2. System model

2.1. Overview

Let us consider the case presented in Figure 1.

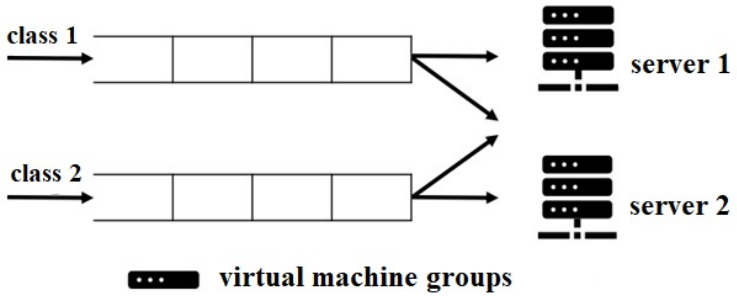

Figure 1. The architecture of two-class multi-server queuing system with a controllable cross-connectivity

In this scenario, two web applications (web apps), classes 1 and 2 in figure, are deployed over the cloud and parallelized on two different services. Each server hosts groups of virtual machines (VM) and each group is assigned to its own class. If the server is not able to provide access to the web apps, the cloud can adapt to network load conditions and another server will provide the necessary resources on demand. In other words, in the system of applied cross-type service, when customers can connect both to the virtual machines of selected server type and to an alternative if there are free virtual machines. If the cloud is not able to provide services, i.e. all virtual machines are busy, the customer will wait for answer. Note that the duration of a service is not related to the type of web apps (class), but is related to a number of server. This scenario of service imposes additional penalties, when customer service on the server assigned to this class of application is cheaper than providing additional resources on an alternative server. It would be logical to assume that it would be more profitable for the cloud provider to leave the customer to wait for the release of resources on his group of virtual machines. But customer waiting also imposes cost losses, downtime of resources, as well as long processing of a request on a low-performance server will be unprofitable factors. As a result, the main idea is to optimally distribute customers between the two servers by calculating the optimal scheme for queue selecting.

2.2. Admission control

In this and the next section consider in more detail how access to virtual machines is performed. With this configuration (as mentioned earlier), the following processes are possible in the operation of the system: receipt of a request from a customer to connect to the cloud, providing access to the first server and providing access to the second server. For the case of receipt of the request:

1. If there are no waiting customers and virtual machines corresponding to this type of request are free, then access to the service can be initiated.
2. If there are no waiting customers, virtual machines corresponding to this type of request are busy, but the alternatives are free, then access to the service can be initiated on alternative server.
3. If all servers are busy, the customer will wait for cloud access in its queue.

2.3. Selection control

The considered system takes into account the cost of providing access to a particular server, therefore it is important to describe the processes that occur when providing a web app on both servers. For the case of providing access to the first server:

1. If there are no waiting customers, then the virtual machines of first server will be idle.
2. If there are first-class waiting customers and there are no second-class waiting customers, then first-class waiting customer will be given access to the first server.
3. If there are no first-class waiting customers and there are second-class waiting customers, then second-class waiting customer will be given access to the first server.
4. If there are all-classes waiting customers, it is necessary to select who will be given access to the first server.

And for the case of providing access to the second server:

1. If there are no waiting customers, then the virtual machines of second server will be idle.
2. If there are second-class waiting customers and there are no first-class waiting customers, then second-class waiting customer will be given access to the second server.
3. If there are no second-class waiting customers and there are first-class waiting customers, then first-class waiting customer will be given access to the second server.
4. If there are all-classes waiting customers, it is necessary to select who will be given access to the first server.

Item 3 reflects the condition of queuing only when all virtual machines are busy. To make the functioning of the system clearer, let's limit the number of virtual machines on each server to one. An example of such system is shown in the Figure 5. In the figure groups of virtual machines (in our case, one

VM on each server) are shown with circles, the different classes are indicated by a filled circle and a shaded circle. A circle within a rectangle represents a waiting customer.

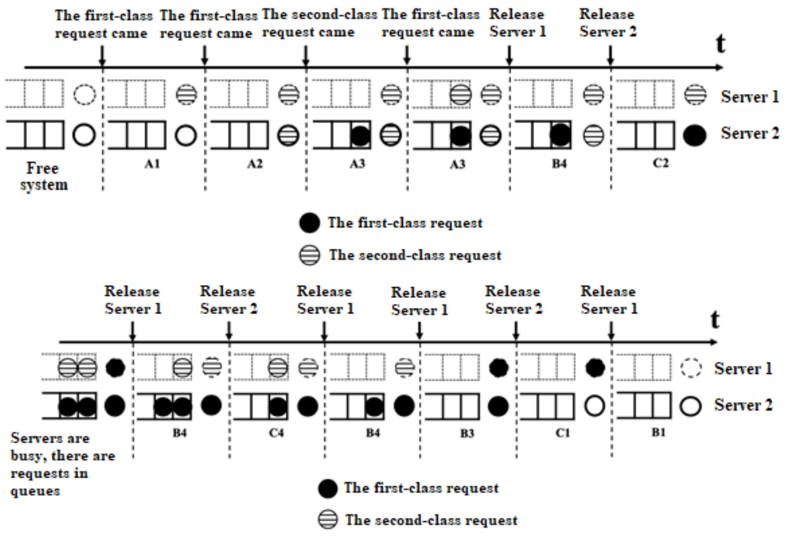

Figure 2. Example of system behavior with one virtual machine on each server

2.4. Problem of finding a routing policy

The step of describing how to access the servers is led to the problem of selection the class of customers when both servers are busy and the system has both types of waiting customers (items 4 in cases of providing access to the servers in previous section are just responsible for this problem). Hence, the question arises - how to define customers who will serviced when the virtual machine is released, i.e. what type of request will be granted access to the cloud. This customer's choice will call *a routing policy.*

There are several methods for organizing and processing queues. We can use the fixed principle or, for example, exhaustive, when we take requests from one queue until it becomes empty then from another until this one becomes empty then we go back to the first, etc. Also we can choose a random principle: from the first, then from the second. Hence, how to choose the routing policy with maximum efficiency?

Therefore, the first thing is to understand the criterion by which to choose the best, i.e. optimal routing policy. The most common problem for models with additional penalties, i.e. which take into account the costs of waiting in queues and servicing on "own" or "alternative" virtual machines, is the problem of minimizing average losses per unit of time. This problem covers special cases of minimizing the average number of requests or the average time in the system. Let's choose the first option as a criterion for routing policy for our study.

And the second thing is how to find this optimal policy: either by brute force, or using a controlled queuing system, the analysis of which allows to find for a given criterion this optimal policy. This algorithm is dynamic and will depend on the state of the system. Those, for each state its own optimal routing policy (some control matrix) can be chosen. This approach allows for a more narrowly configure the system.

3. Queuing model

3.1. Description

The queuing network depicted in Figure 3 models the application of Figure 1.

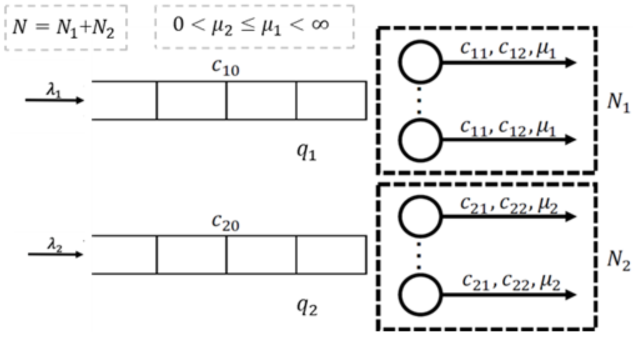

Figure 3. Model in the form of a queuing system

The main parameters of the system are reflected in Table 1. It is composed of two buffers and cross-type service. The system has N_1 and N_2 ($N_1+N_2 = N$, N_k, $k \in \{1, 2\}$) groups of devices, as well as storage units with infinite capacity for the first and second class of customers.Two classes of arrivals are assumed to be generated according a Poisson process with parameters λ_1 and λ_2. The service time is distributed exponentially with intensities μ_1 and μ_2, in such a way that $0 < \mu_2 \leqslant \mu_1 < \infty$. It is also taken into account that one group of devices is more powerful than another. If all groups of devices are busy, the customers arrive in the infinite buffer of its type. The service cost we denote by c_{kj}, where $k \in \{1, 2\}$ is the class of customers and $j \in \{1, 2\}$ is the indicator of our and alternative devices, ($c_{k2} > c_{k1}$): 1 — servicing on our devices, 2 — servicing on an alternative. In other words, c_{11}/c_{21} — cost of servicing on our first/second group devices; c_{12}/c_{22} — cost of servicing on an alternative first/second group devices; c_{k0} ($c_{k0} > 0$) — cost of waiting for service in the k-buffer, $k \in \{1, 2\}$.

3.2. Stochastic process

According the above system description, denote as $Q_k(t)$ — number of customers in the k-buffer at time t and $D_{kj}(t)$ — number of j-customers on k-server at time t. In other words, at some arbitrary time: d_{11} — number of customers of the 1st type on virtual machines of the 1st server, d_{12} — number of customers of the 2nd type on virtual machines of the 1st server, d_{21} — number of customers of the 1st type on virtual machines of the 2nd server, d_{22} — number of customers of the 2nd type on virtual machines of the 2nd server.

So, this system may be modeled by a multidimensional Markov chain with continuous time $\vec{X}(t) = \{Q_1(t), Q_2(t), D_{11}(t), D_{12}(t), D_{21}(t), D_{22}(t)\}$ – the number of customers in the system at t, $t \geqslant 0$ on a state space X:

$$X = \{(\vec{x} = (q_1, q_2, d_{11}, d_{12}, d_{21}, d_{22})) : d_{kj} \geqslant 0, q_k \geqslant 0, k, j = 1, 2; \quad (1)$$

$$(0, 0, d_{11}, d_{12}, d_{21}, d_{22}) : d_{k1} + d_{k2} \leqslant N_k; \quad (2)$$

$$q_1 + q_2 > 0,\ d_{k1} + d_{k2} = N_k, k = 1, 2\}. \quad (3)$$

Table 1

System parameters

Parameters	**Description**
Server	
k	server, $k = \{1; 2\}$
N_k	number of devices (virtual machines) of k-server
μ_k	service intensity of k-server virtual machines (exponential distribution)
Customer and queue	
j	type of arrivals (class of customer), $j = \{1; 2\}$
λ_j	k-th incoming flow rate
Cost	
c_{k0}	cost of waiting in k-customer queue
c_{k1}	cost of k-customer servicing on our devices
c_{k2}	cost of k-customer servicing on alternative devices

3.3. Policy

It is clear that if there are different options for selecting customers from queues then the transition intensity between the states of the system will change. Transition rate matrix can be described in accordance with the rules from in 2.2, 2.3 when $q_1 + q_2 = 0$. And if two queues are occupied, $d_{kj} > 0, q_1 + q_2 \geqslant 1$ (p.4 of 2.3), then in accordance with the queue selection function in a fixed state $\vec{x}$ when servicing j-type customers on k-service:

$$f_{kj}(\vec{x}) \in \{1, 2\}, \quad \vec{x} \in X : q_1 + q_2 > 0. \quad (4)$$

Those the elements are the numbers of the queue from which the customer for the freed device will be taken. Based on the given rules if $q_2 \geqslant 1, q_1 = 0$

then $f_{kj}(\vec{x}) = 2$, if $q_1 \geqslant 1, q_2 = 0$ then $f_{kj}(\vec{x}) = 1$. At $q_1 \geqslant 1, q_2 \geqslant 1$ $f_{kj}(\vec{x})$ is not defined. Besides, this function will depend not only on the current state of the system, but also on the server on which the customer was served. Denote as $\vec{f}(\vec{x}) = (f_{11}(\vec{x}), f_{12}(\vec{x}), f_{21}(\vec{x}), f_{22}(\vec{x}))$ – vector of politics at different values of k, j. We will call the *routing policy* a vector

$$f = \vec{f} = (\vec{f}(\vec{x}), \vec{x} \in X : q_1 + q_2 > 0) \tag{5}$$

of the four components of such selection management models, which is a stationary policy for selecting a queue number to service a customer on a vacated device each time immediately before service ends. It will depend on the server on which the customer was served and what class of customer it is.

Thus, if define a fixed strategy f we can write out the corresponding equilibrium equations system and find the probability distribution. This probability distribution will be denoted as:

$$\pi^f(\vec{x}) = \mathrm{P}[^f(t) = \vec{x}]. \tag{6}$$

3.4. Minimizing cost as optimal policy

Now based on reduction of delays as the selected optimization criterion, it is necessary to compute the service cost for our and alternative devices taking into account waiting in the queue. In some state x it can be represented by:

$$c(\vec{x}) = \sum_{k=1}^{2} (c_{k0} q_k + c_{k1} d_{k1} + c_{k2} d_{k2}). \tag{7}$$

Then the average cost of operating the system for a fixed policy can be described as:

$$g^f = \sum_{\vec{x} \in X} c(\vec{x}) \pi^f(\vec{x}). \tag{8}$$

Finally, we will consider the optimal routing policy f^* to be the one that minimizes these values:

$$g^* = \min_f g^f. \tag{9}$$

There is an iterative routing policy algorithm [13] that allows, based on the initial, fixed policy, to construct a sequence of improved policies until the optimal average cost is reached. As a result of performing the described steps, we get matrix of queue selection for each state of the system taking into account the minimization of costs.

4. Simulation model and numerical analysis

4.1. Performance measures

We have studied the properties of a two-buffer queuing system with cross-type service and additional penalties. Performance parameters of this system can be easily found. *The average number of customers of each type in the* *queue*, which is obtained by summing the number of customers in the queue for a fixed routing policy over all system states:

$$\overline{Q}_k = \sum_{\vec{x}\in X} q_k \pi^f(\vec{x}) = \\ = \sum_{q_1=0}^{\infty} \sum_{q_2=0}^{\infty} \sum_{d_{11}+d_{12}=N_1} \sum_{d_{21}+d_{22}=N_2} q_k \pi^f(q_1, q_2, d_{11}, d_{12}, d_{21}, d_{22}). \quad (10)$$

The average number of devices servicing 1 *and* 2 *classes of customers*, which is calculated by summing the number of customers serviced over all system states:

$$\overline{C}_j = \sum_{\vec{x}\in X} (d_{11} + d_{12} + d_{21} + d_{22}) \pi^f(\vec{x}) = \\ = \sum_{q_1+q2=0} \sum_{d_{11}+d_{12}=0}^{N_1} \sum_{d_{21}+d_{22}=0}^{N_2} \sum_{k=0}^{2} (d_{k1} + d_{k2}) \pi^f(q_1, q_2, d_{11}, d_{12}, d_{21}, d_{22}) + \\ + \sum_{q_1+q2>0} \sum_{d_{11}+d_{12}=N_1} \sum_{d_{21}+d_{22}=N_2} (N_1 + N_2) \pi^f(q_1, q_2, d_{11}, d_{12}, d_{21}, d_{22}). \quad (11)$$

And *the average number of customers in the system*: $\overline{N} = \sum_{j=1}^{2} (\overline{Q}_j + \overline{C}_j)$.

4.2. Simulation model

It has already been shown that the considered routing policy depends on the states of the system. Therefore, it is dynamic, which means that the question arises how this policy (i.e. transition rate matrix) is sensitive to changes in input data. Another issue is the study of the behavior of the model in other distribution laws. If the first question can be solved by a mathematical model, then the second cannot. Therefore, to analyze the studied model we build a simulation model in the *Anylogic* environment.

To simulate the proposed model, we settled on the *AnyLogic* software tool. *AnyLogic* system is based on the use of the object-oriented Java language. This determines the principles for creating, debugging, and deploying simulation models. One of the features of this tool is the ability to flexibly integrate with external programs, in our case, it is msSQL.

Since the selection policy is a fairly large array of data, we load it into an external Database Management System (DBMS) and each time, according to the degree of fullness of the queues, *AnyLogic* sends a request to msSQL. Table 2 lists the elements and their values of the simulator.

Formulated the table of rules is as follows: $(q_1, q_2, d_{11}, d_{12}, d_{21}, d_{22}, f_{11}, f_{12}, f_{21}, f_{22})$ where q_1 and q_2 is current queue status. d_{11} is number of order from their queue d_{12} is number of orders from someone else's queue. d_{21} and d_{22} are similarly. Indicators $(f_{11}, f_{12}, f_{21}, f_{22})$ reflect which queue the request is taken from, where it is denoted by binary values 0 and 1. SQL requests s are written on *selectOutput* elements of the AnyLogic simulator.

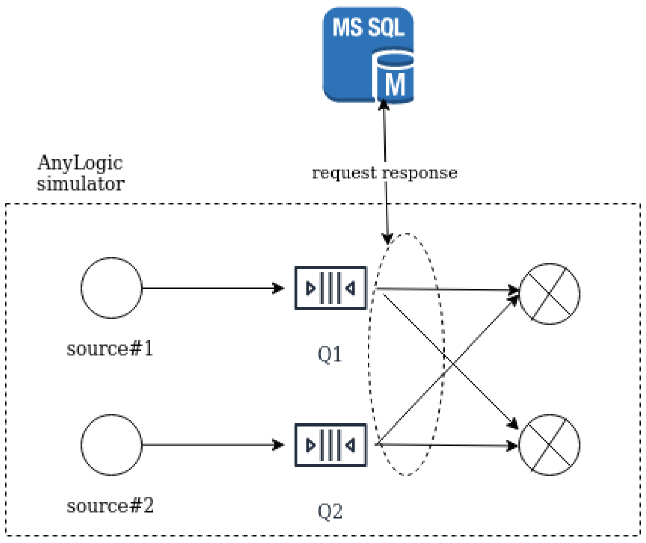

Figure 4. Basic architecture of the simulator

Table 2

Simulation components

Elements	**Value**
source1	input flow
source1	input flow
TS1,TS2,TS,TS3	elements for marking orders with a temporary labels
queue1,queue2	two classes of queues
TE1,TE2	elements for reading labels
selectOutput1,selectOutput2	elements that distribute to instrument groups according to SQL requests
dalay1,delay2	Group of devices
auxiliary variables	
delA	busy state of the first device group
delS	busy state of the second device group
qu1	current state of first queue size
qu2	current state of second queue size

4.3. Numerical example

We start considering requests arriving at rate system with $N_1 = N_2 = 5$ virtual machines on each server. Assume that the incoming flow rates of the 1st and 2nd classes are identical and equal $\lambda_1 = \lambda_2 = 30$. Since the first server is faster then the service intensity in the devices of the first server is $\mu_1 = 20$ and in the second server is $\mu_2 = 5$. The system also has two buffers of infinite capacity. The results of simulation (under exponential assumptions) in the Anylogic environment (Figure 5) are presented in the Table 3. Because the input data for this model is a fixed routing policy then only a fixed case is considered here. Also, the behavior of the system was studied under normal distribution assumptions.

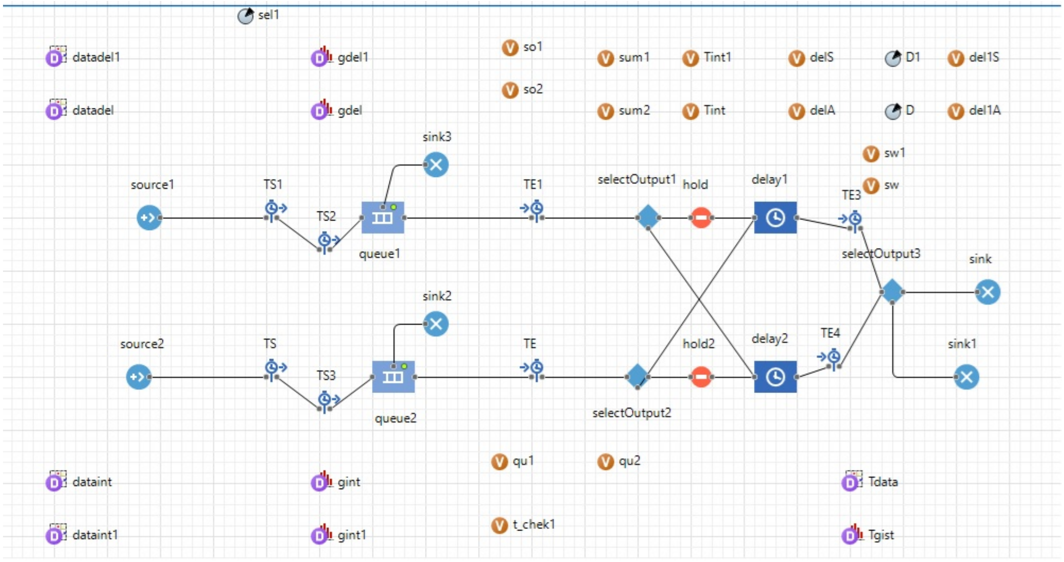

Figure 5. Model schema in Anylogic

The results obtained indicate that requests are practically not delayed in the queue, and therefore the policy for a given model can be considered optimal. Although Poisson flow is the simplest for simulation, it is quite acceptable for performance evaluation.

5. Conclusions

In this paper we analyze the queuing system with two parallel buffers supplied with two groups of servers. A queuing system and a simulation model have been constructed. Initial data were set and the results of the simulation model were obtained. The results obtained indicate that requests are practically not delayed in the queue of the presented QS, and therefore the policy for a given model can be considered optimal. Although Poisson flow is the simplest for simulation, it is quite acceptable for performance evaluation. In the future, it is planned to conduct several more experiments for different values of the intensity of requests and various types of incoming flows.

Table 3

Simulation results

Key Performance Indicators	**Exp (30)**	**Norm (30, 0.001)**	**Norm (30, 0.01)**
Average 1-queue length	0.206	0.289	0.163
Average 2-queue length	0.153	0.168	0.121
Average number of customers serviced on the 1st group of virtual machines	3.85	3.87	3.32
Average number of customers serviced on the 2nd group of virtual machines	3.91	3.94	3.41
Average number of 1-class customers in the system	4.11	4.18	3.68
Average number of 2-class customers in the system	04.08	4.10	3.531
Average time of 1-class of waiting customers	0.007	0.008	0.005
Average time of 2-class of waiting customers	0.005	0.004	0.004
Average time in the system of customers serviced on the 1st group of virtual machines	0.0503	0.0503	0.0402
Average time in the system of customers serviced on the 2nd group of virtual machines	0.199	0.199	0.169
Average time in the system of 1-class customers	0.068	0.070	0.059
Average time in the system of 2-class customers	0.133	0.135	0.124

Acknowledgments

This paper has been supported by the RUDN University Strategic Academic Leadership Program (recipients Irina Kochetkova, Anastasia Vlaskina, simulation model). The reported study was funded by RFBR, project number 20-37-70079 (recipients Irina Kochetkova, Anastasia Vlaskina, Sofia Burtseva, mathematical model).

References

[1] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, and M. Zaharia, "A view of cloud computing," *Communications of the ACM*, vol. 4, no. 53, pp. 50–58, 2010. DOI: 10.1145/1721654.1721672.

[2] N. Taleb and E. A. Mohamed, "Cloud computing trends: a literature review," *Academic Journal of Interdisciplinary Studies*, vol. 1, no. 9, pp. 91–104, 2020. DOI: 10.36941/ajis-2020-0008.

[3] I. Baldini, P. Castro, K. Chang, P. Cheng, S. Fink, V. Ishakian, and P. Suter, "Serverless computing: current trends and open problems," *Research Advances in Cloud Computing*, pp. 1–20, 2017. DOI: 10.1007/978-981-10-5026-8_1.

[4] V. Sontakke and R. B. Dayanand, "Optimization of Hadoop MapReduce Model in cloud Computing Environment," Proceedings of the 2nd International Conference on Smart Systems and Inventive Technology, ICSSIT 8987823, 2019, pp. 510–515. DOI: 10.1109/ICSSIT46314.2019.8987823.

[5] A. Yashwanth Reddy and R. P. Singh, "Design and development of multi tenancy in cloud: security issues," *International Journal of Scientific and Technology Research*, vol. 3, no. 9, pp. 694–697, 2020.

[6] B. Kalyani and K. Rao, "Assessment of physical server reliability in multi cloud computing system," AIP Conference Proceedings 1952,020045, 2018. DOI: 10.1063/1.5032007.

[7] Y. Li, Y. Xu, and J. Chen, "Research on spam pages identification in search service based on cloud computing," *Huazhong Keji Daxue Xuebao (Ziran Kexue Ban)/Journal of Huazhong University of Science and Technology (Natural Science Edition)*, vol. 1, no. 40, pp. 249–253, 2012.

[8] A. Madankan, A. Delavarkhalafi, S. M. Karbassi, and F. Adibnia, "Resource allocation in cloud computing via optimal control to queuing system," *Bulletin of the South Ural State University, Series: Mathematical Modelling, Programming and Computer Software*, vol. 4, no. 12, pp. 67–81, 2019. DOI: 10.14529/mmp190405.

[9] C. N. Khac, K. B. Thanh, H. H. Dac, S. N. Hong, V. P. Tran, and H. T. Cong, "An Open Jackson Network Model for Heterogeneous Infrastructure as a Service on Cloud Computing," *International Journal of Computer Networks and Communications*, vol. 1, no. 11, pp. 63–80, 2019. DOI: 10.5121/ijcnc.2019.11104.

[10] Z. Cheng, H. Li, Q. Huang, Y. Cheng, and G. Chen, "Research on elastic resource management for multi-queue under cloud computing environment," *Journal of Physics: Conference Series*, vol. 9, no. 898, 2017. DOI: 10.1088/1742-6596/898/9/092003.

[11] L. Jing, C. Yidong, and M. Yan, "Modeling Message Queueing Services with Reliability Guarantee in Cloud Computing Environment Using Colored Petri Net," *Mathematical Problems in Engineering*, 2015. DOI: 10.1155/2015/383846.

[12] S. Kannan and S. Ramakrishnan, "Performance Analysis of Cloud Computing in Healthcare System Using Tandem Queues," *International Journal of Intelligent Engineering and Systems*, vol. 4, no. 10, pp. 256–264, 2017. DOI: 10.22266/ijies2017.0831.27.

[13] D. Efrosinin, I. Gudkova, and N. Stepanova, "Algorithmic analysis of a two-class multi-server heterogeneous queuing system with a controllable cross-connectivity," *Lecture Notes in Computer Science*, vol. 12023, 2020. DOI: 10.1007/978-3-030-62885-7_1.

For citation:
I. A. Kochetkova, A. S. Vlaskina, D. V. Efrosinin, A. A. Khakimov, S. A. Burtseva, To analysis of a two-buffer queuing system with cross-type service and additional penalties, Discrete and Continuous Models and Applied Computational Science 29 (2) (2021) 158–172. DOI: 10.22363/2658-4670-2021-29-2-158-172.

Information about the authors:
Kochetkova, Irina A. — Candidate of Physical and Mathematical Sciences, assistant professor of Department of Applied Probability and Informatics of Peoples' Friendship University of Russia (RUDN University); Senior Researcher of Institute of Informatics Problems of Federal Research Center "Computer Science and Control" Russian Academy of Sciences (e-mail: gudkova-ia@rudn.ru, phone: +7(495)9550927, ORCID: https://orcid.org/0000-0002-1594-427X, ResearcherID: E-3806-2014, Scopus Author ID: 35332169400)

Vlaskina, Anastasia S. — PHD student of Department of Applied Probability and Informatics of Peoples' Friendship University of Russia (RUDN University) (e-mail: vlaskina.anastasia@yandex.ru, ORCID: https://orcid.org/0000-0001-6453-814X, Scopus Author ID: 57204395118)

Efrosinin, Dmitriy V. — Doctor of Science in physics and mathematics, associate professor of Johannes Kepler Universitaet Linz; associate professor of Peoples' Friendship University of Russia (RUDN University) (e-mail: dmitry.efrosinin@jku.at, ORCID: https://orcid.org/0000-0002-0902-6640)

Khakimov, Abdukodir A. — Junior researcher of Department of Applied Probability and Informatics of Peoples' Friendship University of Russia (RUDN University) (e-mail: khakimov-aa@rudn.ru, ORCID: https://orcid.org/0000-0003-2362-3270)

Burtseva, Sofiya A. — Master student of Department of Applied Probability and Informatics of Peoples' Friendship University of Russia (RUDN University) (e-mail: sofiya_burceva@inbox.ru, ORCID: https://orcid.org/0000-0003-4305-7050)

УДК 519.872:519.217
PACS 07.05.Tp, 02.60.Pn, 02.70.Bf
DOI: 10.22363/2658-4670-2021-29-2-158-172

К анализу двухбуферной системы массового обслуживания с кросс-типом обслуживания и дополнительными штрафами

И. А. Кочеткова[1,2], А. С. Власкина[1], Д. В. Ефросинин[1,3], А. А. Хакимов[1], С. А. Бурцева[1]

[1] *Кафедра прикладной информатики и теории вероятностей*
Российский университет дружбы народов
ул. Миклухо-Маклая, д. 6, Москва, 117198, Россия
[2] *Институт проблем информатики*
Федеральный исследовательский центр «Информатика и управление» РАН
ул. Вавилова, д. 44, корп. 2, Москва, 119333, Россия
[3] *Линцский университет*
Альтенбергерштрассе, д. 69, Линц, Австрия, 4040

Концепция облачных вычислений была создана для улучшения конфиденциальности пользователей и безопасности хранения данных. Однако ресурсы, выделяемые для обработки этих данных, должны быть правильно распределены. Проблема оптимального управления ресурсами в среде облачных вычислений описана во многих научных публикациях. Для решения задач оптимальности распределения ресурсов систем можно использовать построение и анализ характеристик СМО. Авторами проведён анализ системы массового обслуживания с двумя очередями с кросс-типом обслуживания и дополнительными штрафами, который основывается на литературных источниках, рассмотренных в статье. Это позволяет нам оценить, насколько модель, представленная в статье, подходит для применения в облачных вычислениях. Данная система предполагает разные варианты выбора заявок из очередей, номеров очередей, следовательно, интенсивности переходов между состояниями системы будут меняться. Для этого предлагается политика выбора, которая позволяет системе решать, как себя вести в зависимости от своего состояния. Используются четыре компоненты модели управления выбором, которые представляют собой стационарную политику для определения номера очереди, из которой будет взята заявка на обслуживание. Данный выбор происходит каждый раз непосредственно перед окончанием обслуживания. Для численного анализа построена имитационная модель.

Ключевые слова: система массового обслуживания, облачные вычисления, пуассоновский поток, параллельные очереди, оптимальная политика