EVALUATION OF CLOUD COMPUTING CLUSTER PERFORMANCE

Aleksandr O. Volkov,
Moscow Technical University of Communication and Informatics, Moscow, Russia,
aleksandr.o.volkov@phystech.edu

For cloud service providers, one of the most relevant tasks is to maintain the required quality of service (QoS) at an acceptable level for customers. This condition complicates the work of providers, since now they need to not only manage their resources, but also provide the expected level of QoS for customers. All these factors require an accurate and well-adapted mechanism for analyzing the performance of the service provided. For the reasons stated above, the development of a model and algorithms for estimation the required resource is an urgent task that plays a significant role in cloud systems performance evaluation. In cloud systems, there is a serious variance in the requirements for the provided resource, as well as there is a need to quickly process incoming requests and maintain the proper level of quality of service − all of these factors cause difficulties for cloud providers. The proposed analytical model for processing requests for a cloud computing system in the Processor Sharing (PS) service mode allows us to solve emerging problems. In this work, the flow of service requests is described by the Poisson model, which is a special case of the Engset model. The proposed model and the results of its analysis can be used to evaluate the main characteristics of the performance of cloud systems.

Information about author:
Aleksandr O. Volkov, PhD student, MTUCI, the chair of communication networks and commutation systems Moscow, Russia

INTRODUCTION

Cloud computing is a model that provides convenient, on-demand network access to a shared pool of computing resources that can be quickly provisioned and released with minimal operating costs. Typically, in a cloud computing environment, there are always three tiers: infrastructure providers, cloud service providers and customers. However, sometimes the infrastructure provider and the cloud provider are presented by the same entity. The infrastructure provider grants access to it's hardware. Three-tier cloud architecture is shown on the Figure 1.
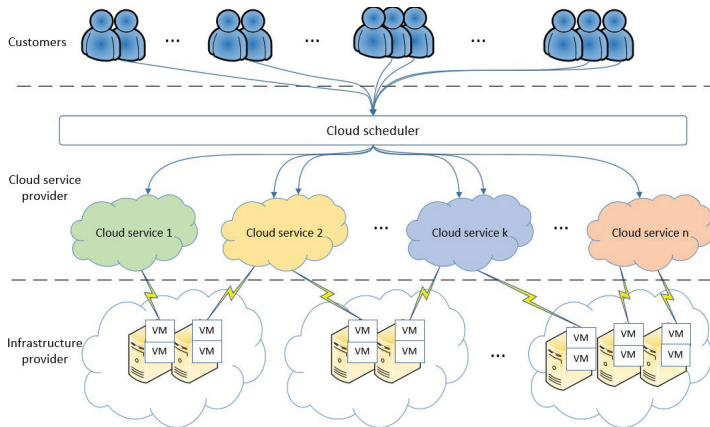

Fig. 1. Three-tier cloud architecture

The service provider gives resources leased from infrastructure providers and permission to use it's cloud services. In real-world cloud computing platforms (such as Amazon EC2, Microsoft Azure, IBM Blue Cloud) there are many work nodes managed by the cloud scheduler. The customer sends a service request to a cloud service provider who provides on-demand services. All requests from clients are placed in the cloud scheduler queue and then distributed among different server virtual machines, depending on the load level of each cluster. After that, the customer receives the requested service with a defined SLA (Service Level Agreement) from the service provider.

There are three main types of cloud computing by access level:

• *SaaS (Software as a Service).* In this case, the client is given access to ready-to-use applications that are deployed providers cloud and are fully served by them. Examples of this kind of systems are Salesforce.com, Google Apps and Google Mail.

• *PaaS (Platform as a Service)* allows clients to develop, launch and manage an application in the cloud in the development environment and programming languages supported by the cloud provider. At the same time, application developers are exempted from the task of in- stalling and maintaining an IDE (Integrated Development Environment) and can fully concentrate on developing the application.

• *IaaS (Infrastructure as a Service)* allows customers to control their own infrastructure without the need to physically maintain it. According to this model, the client is provided with access to data storage's, network resources, virtual servers, or dedicated hardware, via API (application programming interface) or control panel. IaaS is the most flexible agile cloud model, making it easier to manage computing resources and to scale. Typical examples of such services are Microsoft Azure and Amazon Web Services.

In this work we will consider the IaaS type of cloud computing.

RELATED WORKS

In teletraffic papers performance analysis of multiservice models with different modes of service was made in [1] – [5]. In [6] the survey of traffic models for communication networks was presented. In this survey key performance indicators like blocking probability and mean delay are independent of all traffic characteristics beyond the traffic intensity. In particular, a multi-rate model and multi-need model were described.

In [7], the authors tried to solve the common problem of resource provisioning in cloud computing. The problem of allocating resources between different clients is analyzed so that SLA for all types of clients is fulfilled. The cloud is presented as a $M/M/C/C$ system with different priority classes. In their analysis the main criterion of efficiency was the probability of refusal to provide services for different classes of clients, which is determined by analytical methods.

The authors of [8] have proposed a more sophisticated queuing model, consisting of two related subsystems, to evaluate the performance of heterogeneous data centers. Based on the proposed model, average response time, average latency and other key performance parameters were analyzed. Simulation experiments show that an analytic model is effective for accurate assessment of a heterogeneous data center performance.

In [9] modern cloud systems with large number of servers were analyzed. The cloud is modeled as a $M/G/m/m+r$ system, which contains a buffer of tasks of finite capacity, with the assumption that the time exponentially distributed between arrival and service. To evaluate the performance, the full probability distribution of the response time depending on the number of tasks in the system was obtained. The simulation results showed that their model provides accurate results for the average number of tasks in the system and the probabilities of blocking applications.

MODEL DESCRIPTION

A. General description of the model

In the presented model, the process of simultaneous processing of ordered services in the Processor Sharing mode is considered. Each customer can order one of the n cloud services, which means that the system has n flows. Let us denote by C the cloud performance (total resource amount), expressed in floating point operations (flop/s), and C_k is the maximum performance that can be allocated for the kth client. loud cluster model shown on the Figure 2.

Next, we will consider two possible tasks that will be solved in this work:

• **Task 1.** It is required to find such a C so that a resource in the amount of C_k is provided to service the k-th flow with a probability more than $(1-\varepsilon_k)$, where ε_k is the target system performance indicator. In practical terms, this allows to understand how much the performance of the current cloud should be increased to meet the required performance metrics ε_k.

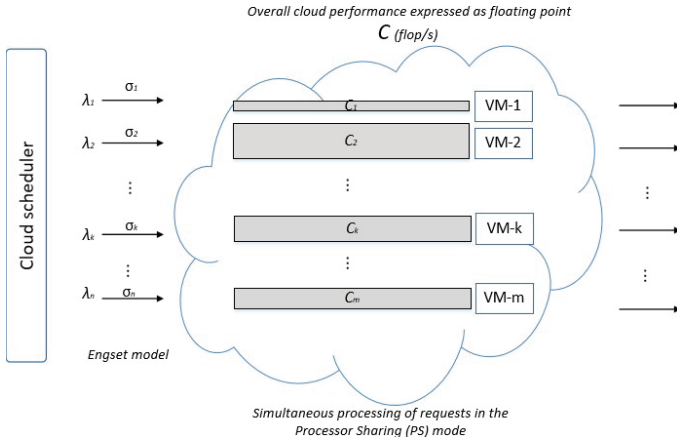*Overall cloud performance expressed as floating point*
C *(flop/s)*

Fig. 2. Cloud cluster model

• **Task 2.** For a given cloud performance, determine the quality of service indicators for requests, such as: the average number of requests being served; average ser- vice time for one request; the average cloud bandwidth provided to serve one request; the fraction of the time the cloud is in saturation. Analyzing QoS metrics helps you determine to what extent your current cloud infrastructure can handle the load it provides and what level of load will be critical for it.

B. Model functioning

The above model can be described using Engset's model. Consider a finite number of sources for k-th flow, n_k. Each source is either active, i.e., with an ongoing request, or idle. Service request durations are independent, exponentially distributed with mean $1/\mu_k$. Idle period durations are independent, exponentially distributed with mean $1/\mu_k$. For k-th flow we refer to $\alpha_k = \gamma_k / \mu_k$, the ratio of the mean call duration to the mean idle period duration, as the traffic intensity per idle source. We assume that $n > m$, i.e., the number of virtual machines is less than the amount of the requested resource, so that some requests may be blocked. A request whose service is blocked starts a new idle period as if the request were accepted and completed instantaneously. The system corresponds to a closed network of two queues an $./M/\infty$ queue and an $./M/m/m$ queue, with n customers that alternately visit both queues and jump over the $./M/m/m$ queue in case of blocking. The number of ongoing requests has the stationary distribution.

In further work, we will use a special case of the Engset model, which is the Poisson model.

We will assume that in the context of the k-th client, the amount of work required has an exponential distribution with the average value of σ_k represented in flop. Requests for the k-th flow arrive in the cloud according to the Poisson process with the intensity λ_k. Then $A_k = \lambda_k \sigma_k$ is the proposed arrival rate of requests from the k-th flow, expressed in flop/s, $\alpha_k = \sigma_k / C_k$ is the intensity of the offered traffic per k-th resource of size C_k, and $\rho_k = A_k / C$ is the coefficient of potential cloud load by serving the considered flow of requests.

Let us introduce the total flow parameters. Let's denote by A is the total intensity of the offered traffic, and by ρ is the coefficient of potential cloud load. The described characteristics are given by the ratios:

$$A = \sum_{k=1}^{n} A_k; \qquad \rho = \sum_{k=1}^{n} \rho_k.$$

Let $(i_1, i_2, ..., i_n)$ is the state of the model, where i_k is the number of requests of the k-th flow that are currently serviced in the cloud, $i_k \in [0, +\infty)$, $k = \overline{1, n}$. For the request of the k-th flow, a performance of C_k flop/s is allocated, if the total amount of performance allocated to all requests, including the current one, does not exceed C. If this condition is not met, then all serviced applications share the entire common resource among themselves. Let's denote by $\upsilon(i_1, i_2, ..., i_n)$ is the cloud performance given to serve the kth flow in the state $(i_1, i_2, ..., i_n)$.

$$\sum_{k=1}^{n} \upsilon(i_1, i_2, ..., i_n) \leq C; \qquad (1)$$
$$\upsilon(i_1, i_2, ..., i_n) \leq i_k C_k, \quad k = \overline{1, n}.$$

C. Target service requests

The above model is described by the Markov process:

$$r(t) = (i_1(t), i_2(t), ..., i_n(t)),$$

where $i_k(t)$ is the number of serviced claims of the k-th flow at the time instant t, $k = \overline{1, n}$.

Let $P(i_1, i_1, ..., i_n)$ is unnormalized probabilities of $r(t)$. For the existence of stationarity of the service mode, it is necessary:

$$P(i_1, i_2, ..., i_n) = P(0, 0, ..., 0)\Phi(i_1, i_2, ..., i_n)\prod_{k=1}^{n} A_k^{i_k}. \qquad (2)$$

Where $\Phi(i_1, i_2, ..., i_n)$ is balance function. In this case, $\Phi(0, 0, ..., 0) = 1$, and for negative values of the state $(i_1, i_2, ..., i_n)$ it is equal to zero. The resource allocation function can be obtained from the balance condition [6]. Formula (2) is equivalent to the following system:

$$P(i_1, ..., i_n) = \begin{cases} P(0, 0, ..., 0)\prod_{k=1}^{n} \dfrac{\alpha_k^{i_k}}{i_k!}, & \text{for } i \leq C; \\ \dfrac{C}{i}\sum_{k=1}^{n} \rho_k P(i_1, ..., i_k - 1, ..., i_n), & \text{for } i > C. \end{cases} \qquad (3)$$

By normalizing the equation (3), the stationary probabilities of the model can be calculated. Let us express through these probabilities G is the probability of congestion (the share of the cloud being in a saturation state), that is, when it is not possible to provide the maximum performance for all received requests.

$$G = \sum_{\{(i_1, ..., i_n)|i_1 C_1 + ... + i_n C_n > C\}} p(i_1, ..., i_n). \qquad (4)$$

Let $m_k(i)$ be the total performance flow of class k, expressed in units of peak allowable performance C_k, that is, for $m_k(i)$ it is true:

$$\begin{cases} m_k(i) = i_k, & for\ i \leq C; \\ \sum_{k=1}^{n} C_k m_k(i) = C, & for\ i > C. \end{cases} \tag{5}$$

Let us determine the main QoS indicators for cloud computing: T_k is the average service time for one request; L_k is the average number of serviced requests in the system; γ_k is the average throughput for servicing one applications. Assuming also that each flow gets maximum performance; we get the performance loss factor for the kth flow W_k.

$$L_k = \sum_{(i_1,...,i_n)} p(i_1,...,i_n)i_k; \quad T_k = \frac{L_k}{\lambda_k}; \quad \gamma_k = \frac{\sigma_k}{T_k} = \frac{A_k}{L_k};$$

$$W_k = \frac{\sum_{(i_1,...,i_n)} p(i_1,...,i_n)(i_k - m_k(i))C_k}{\sum_{(i_1,...,i_n)} p(i_1,...,i_n)i_k C_k}; \quad k = \overline{1,n}. \tag{6}$$

D. Recursive algorithm

Let's calculate the value $i = i_1 C_1 + ... + i_n C_n$ for an arbitrary state of the $(i_1, i_2, ..., i_n)$. We will single out two boundary cases for i: $i \leq C$ and $i > C$. In the first case, all requests receive the maximum possible performance, but if the second inequality is true, then the value of the parameter i can be interpreted as a potential performance requirement for servicing all requests.

Suppose that $i \leq C$ is satisfied. Let's introduce the variables $P(i)$ and $Y_k(i)$:

$$P(i) = \sum_{\{(i_1,...,i_n)|i_1 C_1 + ... + i_n C_n = i\}} P(i_1, i_2, ..., i_n);$$

$$Y_k(i) = \sum_{\{(i_1,...,i_n)|i_1 C_1 + ... + i_n C_n = i\}} P(i_1, i_2, ..., i_n)i_k, \quad k = \overline{1,n}. \tag{7}$$

From (3) and (7) we get expressions for $P(i)$ and $Y_k(i)$:

$$P(i) = \sum_{\{(i_1,...,i_n)|i_1 C_1 + ... + i_n C_n = i\}} P(i_1, i_2, ..., i_n);$$

$$Y_k(i) = \sum_{\{(i_1,...,i_n)|i_1 C_1 + ... + i_n C_n = i\}} P(i_1, i_2, ..., i_n)i_k, \quad k = \overline{1,n}. \tag{8}$$

Next, we obtain a recursive formula for $P(i)$, given that $P(0) = 1$ and $P(i) = 0$ for $i < 0$:

$$P(i) = \sum_{\{(i_1,...,i_n)|i_1 C_1 + ... + i_n C_n = i\}} P(0) \prod_{k=1}^{n} \frac{\alpha_k^{i_k}}{i_k!} =$$

$$= \sum_{\{(i_1,...,i_n)|i_1 C_1 + ... + i_n C_n = i\}} \frac{i}{i} \prod_{k=1}^{n} \frac{\alpha_k^{i_k}}{i_k!} = \tag{9}$$

$$= \sum_{\{(i_1,...,i_n)|i_1 C_1 + ... + i_n C_n = i\}} \frac{1}{i} \alpha_k C_k \frac{\alpha_1^{i_1}}{i_1!} ... \frac{\alpha_k^{i_k-1}}{(i_k-1)!} ... \frac{\alpha_n^{i_n}}{i_n!} I(0 \leq i_k - 1) =$$

$$= \sum_{k=1}^{n} \frac{1}{i} A_k P(i - C_k) I(0 \leq i - C_k), \quad i = \overline{1, C}.$$

In (9) $I(\cdot)$ is an indicator function that takes the value 1, when the inner expression is true, and 0 otherwise. The presented formula is called the Kaufman-Roberts recursion [6]. Similarly, we obtain the formula for $Y_k(i)$:

$$Y_k(i) = \sum_{\{i_1 C_1 + ... + i_n C_n = i\}} P(0) \prod_{k=1}^{n} \frac{\alpha_k^{i_k}}{i_k!} I(i_k = \alpha_k) P(i - C_k). \tag{10}$$

Let us calculate the auxiliary characteristics for the case $i > C$:

$$\overline{P} = \sum_{\{i_1 C_1 + ... + i_n C_n > C\}} P(i_1, i_2, ..., i_n);$$

$$\overline{Y_k} = \sum_{\{i_1 C_1 + ... + i_n C_n > C\}} P(i_1, i_2, ..., i_n)i_k, \quad k = \overline{1,n}, \tag{11}$$

and for the case $i \leq C$:

$$P_k = \sum_{i=C-C_k+1}^{C} P(i), \quad k = \overline{1,n};$$

$$Y_{k,j} = \sum_{i=C-C_j+1}^{C} Y_k(i), \quad j = \overline{1,n}; \quad k = \overline{1,n}. \tag{12}$$

Let's transform (11) using the relation (3):

$$\overline{P} = \sum_{i>C} P(i_1, i_2, ..., i_n) =$$

$$= \sum_{i>C} \sum_{k=1}^{n} P(i_1, ..., i_k - 1, ..., i_n) = \sum_{k=1}^{n} \rho_k(P_k + \overline{P}). \tag{13}$$

Finally, from (13) we get:

$$\overline{P} = \frac{1}{1-\rho} \sum_{k=1}^{n} \rho_k P_k. \tag{14}$$

Similarly, we get the expression for $Y_{k,j}$, using (11) and the following transformations:

$$\overline{Y_k} = \sum_{i>C} P(i_1, i_2, ..., i_n)i_k = \sum_{i>C} \sum_{j=1}^{n} \rho_j P(i_1, ..., i_k - 1, ..., i_n)i_k =$$

$$= \rho_k(P_k + \overline{P}) + \sum_{j=1}^{n} \rho_j(Y_{k,j} + \overline{Y_k}). \tag{15}$$

The final expression will look like this:

$$\overline{Y}_k = \frac{1}{1-\rho}\left[\rho_k(P_k + \overline{P}) + \sum_{j=1}^{n}\rho_j Y_{k,j}\right]. \qquad (16)$$

Finally, let's define a few more characteristics of the model's service quality indicators: G_k is a probability of congestion for the k-th flow, that is, the share of the cloud being in a saturation state for the k-th flow. And ε_k is a target indicator of resource availability C_k, it can actually be interpreted as the probability of blocking an application if it is impossible to provide the required amount of the resource C_k.

$$G_k = \frac{\overline{Y}_k}{Y_k(0) + Y_k(1) + ... + Y_k(n)}, \quad k = \overline{1,n}; \qquad (17)$$

$$\varepsilon_k = \frac{P_k}{\sum_{i=1}^{C} P(i)}, \quad k = \overline{1,n}. \qquad (18)$$

Let us formalize a recursive algorithm to estimate the service characteristics of a model:

1) Let's set the initial value $P(0) = 1$. We obtain an expression for the unnormalized probabilities $P(i)$, where $i = \overline{1,C}$ in terms of $P(0)$, using the relation:

$$P(i) = \frac{1}{i}\sum_{k=1}^{n} A_k P(i - C_k)I(0 \le i - C_k), \qquad (19)$$

derived from (9).

2) Find the unnormalized values of the function $Y_k(i)$ for $i = \overline{C_k, C}$ and $k = \overline{1,n}$ using the formula

$$Y_k(i) = \alpha_k P(i - C_k), \qquad (20)$$

which follows from (10).

3) Referring to (12), we find auxiliary characteristics:

$$P_k = \sum_{i=C-C_k+1}^{C} P(i), \quad k = \overline{1,n};$$
$$Y_{k,j} = \sum_{i=C-C_j+1}^{C} Y_k(i), \quad j = \overline{1,n}; \quad k = \overline{1,n}. \qquad (21)$$

Using (13), (14), (15) and (16), we find auxiliary characteristics that determine the behavior of the model when $i > C$, as a result we get:

$$\overline{Y}_k = \frac{1}{1-\rho}\left[\rho_k(P_k + \overline{P}) + \sum_{j=1}^{n}\rho_j Y_{k,j}\right]; \qquad (22)$$

$$\overline{P} = \frac{1}{1-\rho}\sum_{k=1}^{n}\rho_k P_k. \qquad (23)$$

4) Let us calculate the normalization constant:

$$N = P(0) + P(1) + ... + P(C) + \overline{P}.$$

5) Let's calculate the values of QoS indicators of requests using (4), (6) and (17):

$$L_k = \frac{1}{N}(\sum_{i=1}^{C} Y_k(i) + \overline{Y}_k); \quad T_k = \frac{L_k}{\lambda_k};$$

$$G_k = \frac{\overline{Y}_k}{Y_k(0) + Y_k(1) + ... + Y_k(n) + \overline{Y}_k}; \quad G = \frac{\overline{P}}{N}; \qquad (24)$$

$$\gamma_k = \frac{\sigma_k}{T_k} = \frac{A_k}{L_k}; \quad W_k = 1 - \frac{\gamma_k}{C_k}; \quad k = \overline{1,n}.$$

E. Algorithm for evaluating cloud performance C

To evaluate the performance of the C cloud, let's use points 1 and 3 in the above algorithm:

1) Let's set the initial value $P(0) = 1$. We get $P(i)$, for $i = \overline{1,C}$ and $C = \max(C_1, ..., C_n)$ using the relation (19):

$$P(i) = \frac{1}{i}\sum_{k=1}^{n} A_k P(i - C_k)I(0 \le i - C_k).$$

2) Using (21) we find:

$$P_k = \sum_{i=C-C_k+1}^{C} P(i), \quad k = \overline{1,n}.$$

3) Using (19), calculate the performance targets $\widehat{\varepsilon}_k$ for a given C and find the smallest of them:

$$\widehat{\varepsilon}_k = \frac{P_k}{\sum_{i=1}^{C} P(i)}, \quad k = \overline{1,n}.$$

4) To stop the iterative cycle, repeat steps 1-3, iteratively increasing C, as long as the sum of difference in square between the assigned performance indicators and calculated in the previous paragraph should tend to zero ($\sum_{k=1}^{n}(\widehat{\varepsilon}_k - \varepsilon_k)^2 \to 0$).

EVALUATION OF CLOUD COMPUTING PERFORMANCE

A program in the Python programming language was developed that implements the recursive algorithms for evaluating the performance and QoS indicators described above. The source of the initial data was Amazon Web Services, in particular it's computing cloud Amazon Elastic Compute Cloud (Amazon EC2). The choice of this source is due to the fact that at the moment AWS is one of the leaders among the cloud computing providers.

Let us analyze the dependence of the cloud load factor on the total traffic intensity A for different sets of ε_k. Consider a model with n = 3 service classes, with the corresponding computing power: C_1 = 50 Gflop/s, C_2 = 100 Gflop/s and C_3 = 500 Gflop/s, with targets performance $\varepsilon_1 = 1\%$, $\varepsilon_2 = 2\%$, $\varepsilon_3 = 5\%$

for the first case, $\varepsilon_1 = 3\%$, $\varepsilon_2 = 5\%$, $\varepsilon_3 = 8\%$ for the second and $\varepsilon_1 = 5\%$, $\varepsilon_2 = 7\%$, $\varepsilon_3 = 9\%$ for the third, respectively. Let's build a graph of the dependence of ρ on A, the graph is shown in Figure 3.
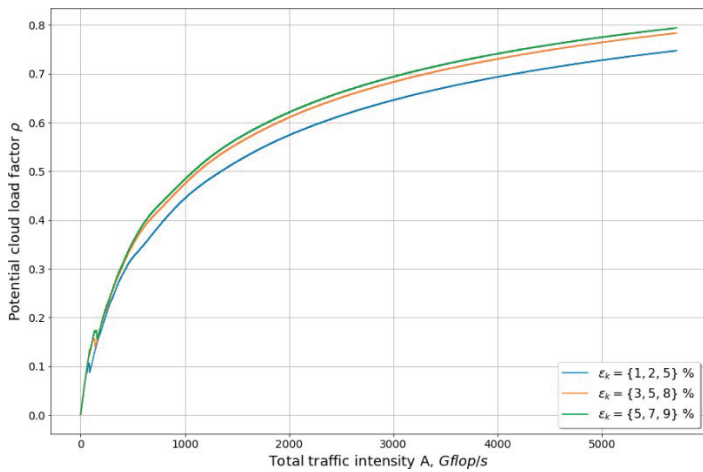

Fig. 3. Dependence of the coefficient of potential cloud load ρ on the total traffic intensity A, Gflop/s

The graph shows that the smaller the ε_k, the slower the potential load. This reflects the following logic: to provide small ε_k, more C is required, which means $\rho = A/C$ will increase at a slower rate. Note also that it is typical for all cases that for a low traffic intensity ρ grows rather quickly, however, when $A > \sum C_k$, the growth of the load factor becomes smoother. In addition, near the maximum value of the transmitted resource, the graphics begin to diverge significantly.

A. Dependence of QoS metrics on the characteristics of the cloud computing model

Let's analyze the second problem posed earlier. In order to understand what the QoS level will be for different types of virtual machines, let's simulate a cloud node with 7 different virtual machines, as an example, take the following Amazon EC2 instances: 3 general purpose instances (M4: m4.2xlarge, m4.4xlarge, m4.10xlarge) and 4 instances optimized for computational tasks (C4: c4.8xlarge; C5: c5.9xlarge, c5.12xlarge, c5d.12xlarge). As a result, we get the following input parameters:

- We consider a cloud with a total performance value $C = 2$ Tflop/s and 7 Poisson streams, with intensity $\lambda_k = A_k / \sigma_k$ each;
- The amount of work required for each flow is $\sigma = 100$ Tflop;
- It is assumed that the contribution of orders from the kth flow is exactly the same as from other flows, that is, $\rho_k = \rho / n$, $A_k = \rho_k C$;
- Each request has a resource of C_k available within the kth flow with a pool of values: 100 (m4.2xlarge), 200 (m4.4xlarge), 400 (m4.10xlarge), 600 (c4.8xlarge), 1000 (c5.9xlarge), 1500 (c5.12xlarge), 2000 (c5d.12xlarge, unlimited access) respectively for $k = \overline{1,7}$.

The Figure 4 shows the relationship between the performance loss factor for the k-th flow and the load factor ρ. If the value of W_k is zero, it means that the performance of C_k is fully available to the user. As you can see from the graph, the smaller the size of the required resource, the higher the level of utilization is required for there to be a loss in the provided performance. Here, again, the acceptable load level depends on the value of C_k and is determined by the value of ρ, at which W_k starts to take on a nonzero value.
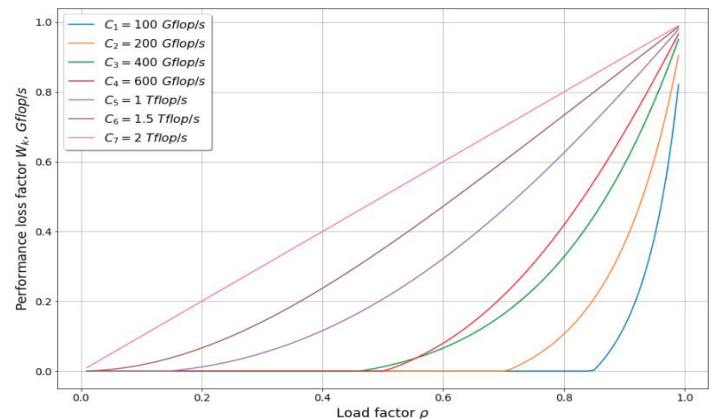

Fig. 4. Dependence of the productivity loss factor for the *k*-th flow on ρ for different C_k

The dependence of the service time of one customer on ρ is shown in the Figure 5. Similarly to the previous obtained dependence for small ρ, the service execution time is determined by C_k and reaches the values σ / C_k at ρ is close to zero, however, as the cloud load tends to unity, the Tk tends to $T_k = \rho_k / \lambda_k (1-\rho) = \sigma / C (1-\rho)$ (Little's formula). The dependence of the probability of congestion C_k on ρ is illustrated by the Figure 6. Interestingly, low C_k is characterized by a smoother growth of C_k, while large C_k is characterized by a sharp change in the rate of congestion.
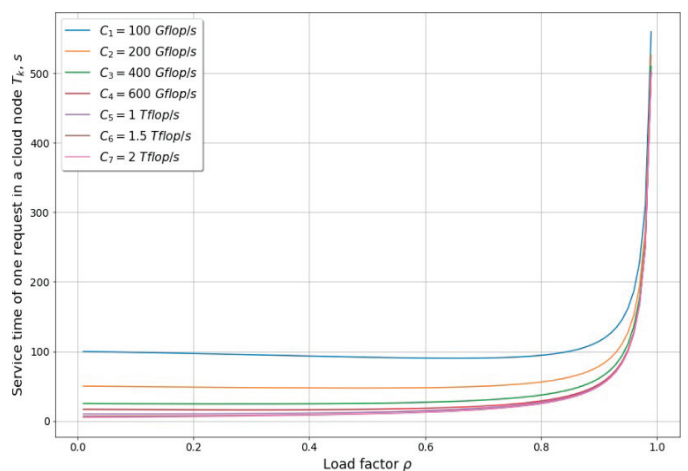

Fig. 5. Dependence of service time for one request in a cloud node on for different C_k
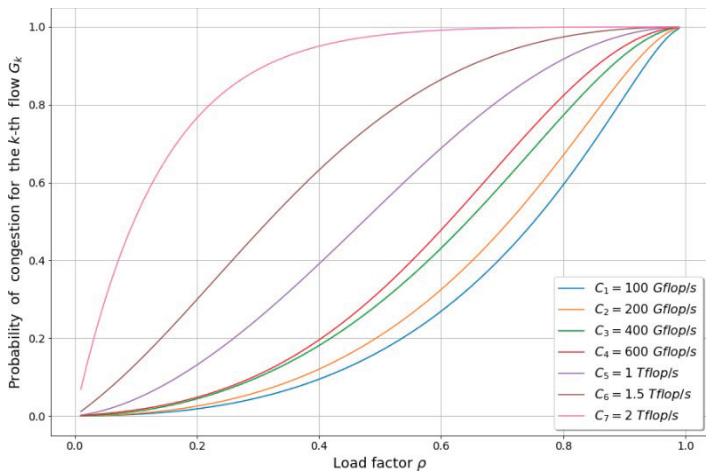
Fig. 6. Dependence of the probability of congestion
for the k-th flow on ρ for different C_k

Conclusion and recommendations
for using the results

The results of the work and the above analysis of the tasks
can be useful in the following cases:

• To evaluate cloud performance. In order to comply with a
SLA, the required performance must be determined. This can be
useful when planning future infrastructure or expanding the cur-
rent one.

• Analysis of such quality of service indicators as the aver-
age number of claims in service, the average service time of one
claim, the average cloud bandwidth provided to service one
claim, the percentage of time the cloud is in a saturation state
helps to determine the state of the current cloud computing infra-
structure, and then how well it handles incoming traffic. In addi-
tion, it can help identify critical load levels and identify bottle-
necks in the system.

References

1. S.N. Stepav, M.S. Stepanov (2017). Planning transmission re-
source at joint servicing of the multiservice real time and elastic data
traffics. *Automation and Remote Control*, vol. 78. no. 11, pp. 2004-
2015.

2. S.N. Stanov, M.S. Stepanov (2018). Planning the resource of
information transmission for connection lines of multiservice hierar-
chical access networks. *Automation and Remote Control*, vol. 79, no. 8,
pp. 1422-1433.

3. S.N. Stanov, M.S. Stepanov (2018). The Model and Algorithms
for Estimation the Performance Measures of Access Node Serving the
Mixture of Real Time and Elastic Data. In: Vishnevskiy V., Kozyrev D.
(eds) Distributed Computer and Communication Networks. DCCN
2018. *Communications in Computer and Information Science (CCIS),*
vol. 919, pp. 264-275. Springer, Cham.

4. S.N. Stanov, M.S. Stepanov (2019). Efficient Algorithm for
Evaluating the Required Volume of Resource in Wireless Communica-
tion Systems under Joint Servicing of Heterogeneous Traffic for the
Internet of Things. *Automation and Remote Control*, vol. 80, no.11,
pp. 1970-1985.

5. . Bonald, J. Virtamo (2005). A recursive formula for multirate
systems with elastic traffic. *IEEE Communications Letters*, vol. 9,
pp. 753-755.

6. T. Bona (2007). Insensitive Traffic Models for Communication
Networks. *Phil. Trans. Roy. Soc.* London, vol. A247, pp. 529-551, June
2007.

7. W. Ellens M. Zivkovic and J. Akkerboom, R. Litjens, H. den
Berg (2012). Performance of cloud computing centers with
multiplepriority classes. *Proceedings of the 5th IEEE International Con-
ference on Cloud Computing*, pp. 245-252.

8. J. W.ai, J. Xi, J.-X. Zhu, Sre-W. Huang (2015). Performance
analysis of heterogeneous data centers in cloud computing using a com-
plex queuing model. *Mathematical Problems in Engineering*, pp. 1-15.

9. Khazaei, J. Misic, V. Misic. Performance Analysis of Cloud
Computing Centers Using M/G/m/m+r Queuing Systems. *IEEE Trans-
actions on Parallel and Distributed Systems*, pp. 936-943.

ОЦЕНКА ПРОИЗВОДИТЕЛЬНОСТИ КЛАСТЕРА ОБЛАЧНЫХ ВЫЧИСЛЕНИЙ

Волков Александр Олегович, *Московский технический университет связи и информатики, Москва, Россия*
aleksandr.o.volkov@phystech.edu

Аннотация

Для поставщиков облачных услуг одной из наиболее актуальных задач является поддержание требуемого качества обслуживания (QoS) на приемлемом для клиентов уровне. Это условие усложняет работу провайдеров, так как теперь им необходимо не только управлять своими ресурсами, но и обеспечивать ожидаемый уровень QoS для клиентов. Все эти факторы требуют точного и хорошо адаптированного механизма анализа эффективности предоставляемой услуги. По указанным выше причинам разработка модели и алгоритмов оценки необходимого ресурса является актуальной задачей, которая играет значительную роль в оценке производительности облачных систем. В облачных системах существуют серьезные различия в требованиях к предоставляемому ресурсу, а также есть необходимость в быстрой обработке входящих запросов и поддержании должного уровня качества обслуживания – все эти факторы создают трудности для облачных провайдеров. Предлагаемая аналитическая модель обработки запросов к системе облачных вычислений в сервисном режиме Processor Sharing (PS) позволяет решать возникающие проблемы. В этой работе поток запросов на обслуживание описывается моделью Пуассона, которая является частным случаем модели Энгсета. Предложенная модель и результаты ее анализа могут быть использованы для оценки основных характеристик производительности облачных систем.

Ключевые слова: облачные вычисления, оценка производительности, мультисервисные модели, совместное использование процессоров (PS).

Литература

1. *Stepanov S.N., Stepanov M.S.* Planning transmission resource at joint servicing of the multiservice real time and elastic data traffics // Automation and Remote Control, 2017, vol. 78. no. 11, pp. 2004-2015.

2. *Stepanov S.N., Stepanov M.S.* Planning the resource of information transmission for connection lines of multiservice hierarchical access networks // Automation and Remote Control, 2018, vol. 79, no. 8, pp. 1422-1433.

3. *Stepanov S.N., Stepanov M.S.* The Model and Algorithms for Estimation the Performance Measures of Access Node Serving the Mixture of Real Time and Elastic Data. In: Vishnevskiy V., Kozyrev D. (eds) Distributed Computer and Communication Networks. DCCN 2018. Communications in Computer and Information Science (CCIS), vol. 919. pp. 264-275. Springer, Cham.

4. *Stepanov S.N., Stepanov M.S.* Eficient Algorithm for Evaluating the Required Volume of Resource in Wireless Communication Systems under Joint Servicing of Heterogeneous Traffic for the Internet of Things // Automation and Remote Control, 2019, vol.80, no.11, pp. 1970-1985.

5. *Bonald T., Virtamo J.* A recursive formula for multirate systems with elastic traffic // IEEE Communications Letters, 2005, vol. 9, pp. 753- 755.

6. *Bonald T..* Insensitive Traf?c Models for Communication Networks // Phil. Trans. Roy. Soc. London, vol. A247, pp. 529-551, June 2007.

7. *Ellens W., Zivkovic M. and Akkerboom J., Litjens R., den Berg H.* Performance of cloud computing centers with multiplepriority classes // Proceedings of the 5th IEEE International Conference on Cloud Com- puting, 2012, pp. 245-252.

8. *Bai J.W., Xi J., Zhu J.-X., Huang Sre-W.* Performance analysis of heterogeneous data centers in cloud computing using a complex queuing model // Mathematical Problems in Engineering, 2015, pp.1-15.

9. *Khazaei H., Misic J., Misic V.* Performance Analysis of Cloud Com- puting Centers Using M/G/m/m+r Queuing Systems // IEEE Transactions on Parallel and Distributed Systems, pp. 936-943.