Maximizing Profit in Cloud Computing Using M/G/c/k Queuing Model

Assia Outamazirt, Kamel Barkaoui, Djamil Aissani

Maximizing Profit in Cloud Computing Using M/G/c/k Queuing Model

Assia Outamazirt, Kamel Barkaoui, Djamil Aissani

Maximizing Profit in Cloud Computing Using M/G/c/k Queuing Model

Assia Outamazirt
Research Unit LaMOS University of Bejaia
Bejaia, Algeria
Email: outamazirt.assia@gmail.com

Kamel Barkaoui
CEDRIC, CNAM
Paris, France
Email: kamel.barkaoui@cnam.fr

Djamil Aïssani
Research Unit LaMOS University of Bejaia
Bejaia, Algeria
Email: lamos_bejaia@hotmail.com

***Abstract*—The economics of cloud computing is more and more a crucial issue. To maximize the profit, a service provider should understand both service charges and business costs, and how they are determined by the characteristics of the applications and the configuration of a multiserver system. In this work, we treat a multiserver system as an $M/G/c/k$ queuing model with impatient customers, such that our optimization problem can be formulated and solved. There are no analytical formulas to calculate performance measures of $M/G/c/k$ queue. Therefore, we provide new analytical formulas to compute the mean waiting time of new service request arrival to the system and the delay probability, i.e., the probability that a newly submitted service request must wait because all servers are busy. We focus on the calculation of these latter, because our performance metric in this work is the task waiting time. Then, expected revenue of a service provider is calculated. Numerical calculations of the optimal server size and the optimal server speed are demonstrated. Finally, the expected profit in one unit of time for a cloud provider is obtained.**

***Keywords*—Cloud computing; multiserver configuration; $M/G/c/k$ queuing model; mean waiting time; delay probability; profit maximization.**

I. Introduction

The economics of cloud computing is more and more a crucial issue. To maximize the profit, a service provider should understand both service charges and business costs, and how they are determined by the characteristics of the applications and the configuration of a multiserver system.

The problem of optimal multiserver configuration for profit maximization in a cloud computing environment has be analyzed in previous researches (see [1], [2], [3], [4]). In [1], a multiserver system is treated as an $M/M/c$ queuing model, such that the optimization problem can be formulated and solved analytically. Their pricing model took into considerations the amount of a service, the workload of an application environment, the configuration of a multiserver system, the service-level agreement, the satisfaction of a consumer and so forth. They also considered two types of server speed and power consumption models.
In [2], the cloud data center is considered as a queuing system with finite capacity, interarrival and service times were both assumed to be exponentially distributed. The authors developed a profit function, in which both the system blocking loss and the user abandonment loss are evaluated in total revenue. Their main goal is to maximize the profit of service providers under a loss probability guarantee.
In [3], a multiserver system is considered as an $M/M/c+D$ queuing model for the profit maximization problem in a homogeneous cloud environment. Both inter-arrival and service times of cloud service request are assumed to be exponentially distributed, and the system has a varying size. In order to guarantee the quality of service requests and maximize the profit of service providers, the authors proposed a novel double-quality-guaranteed renting scheme for service providers. Then, they formulated an optimal configuration problem of profit maximization in which many factors are taken into considerations, such as the market demand, the workload of requests, the server-level agreement, and so forth.

The multiserver system is considered as an $M/M/c$ queuing model, because this queuing model is the only model that accommodates an analytical and closed form expression of the probability density function of the time of a new arrival service request to the system [1]. However, the assumption of the service time being exponentially distributed is inappropriate to substantiate the actual service time of the cloud service request [5]. Therefore, we assume general service time for service requests. Also, as the number of task request arrivals is relatively high while the probability that a single task request arrive at the system is relatively small, then the arrival process can be considered as a Poisson process. Consequently, we treat the problem of optimal multiserver configuration for profit maximization in a cloud computing as an $M/G/c/k$ queuing model.

Estimating the performance measures of the multiserver queuing systems with inter-arrival time and/or service time is generally distributed remains a complex and challenging problem. For $M/G/c$ queues, it has been well known that the delay probability in the $M/M/c$ queue, i.e., the Erlang delay formula, is usually a good approximation for other service time distributions [6]. In [7], based on the exact solution for finite capacity $M/M/c/k$ queues, a different approximation for the blocking probability in $M/G/c/k$ queue has been proposed. In [8], the authors proposed an approximation for the average queuing delay in $M/G/c/k$ queue based on the relationship of joint distribution of remaining service time to the equilibrium service distribution. In [6], by using an excellent approximation for the mean waiting time in the $M/G/c$ queue, the author provided more accurate approximation of the delay probability in the $M/G/c$ queue for small values of the number of servers. But the proliferation of cloud computing has resulted in the establishment of systems multiserveur with a large number of servers, which makes this approximation is not suitable for

performance evaluation of cloud computing. Therefore, we provide an analytical formula to compute the delay probability in $M/G/c/k$ queue regardless of the number of servers considered, and when the capacity of queue tends to infinity, the delay probability in the $M/G/c$ queue can be computed. Also, using this formula, we can compute the mean waiting time of each new arrival to the system. Furthermore, we focus on the computing of this performance measure because the waiting time is the source of customer dissatisfaction and it is the one of the factor which can determine the service charge to a service request.

As we know, when service providers fail to deliver the pre-agreed quality, a penalty or compensation is required. In this paper, the service level is reflected by the waiting time of requests. Hence, we define a deadline as the maximum mean waiting time for customers and if the mean waiting time exceeds this limit, a service will be entirely free with no charge.

To sum up, we treat a multiserver system as an $M/G/c/k$ queuing model with impatient customers, such that our optimization problem can be formulated and solved. As we said before, we provide an analytical formula for computing the delay probability and we also propose an analytical formula for computing the mean waiting time for each new service request arrival to the system. We calculate the expected revenue per unit time with taking into the revenue losses resulting by customers abandonment and the revenue losses resulting by lost customers due to finite capacity of system. Then, we develop a profit function by also taking the cost of infrastructure renting and the cost of energy consumption. Finally, we solve our optimization problem.

The remainder of the paper is organized as follows. In Section II, we present our model. Sections III and IV are devoted to present our proposed analytical formulas for computing mean waiting time of the new service request arrival and delay probability, respectively. In Section V, a profit function is developed. We solve the problem of optimal multiserver configuration for profit maximization in a cloud computing environment and we present our results in Section VI. Finally, Section VII concludes the work.

II. Multiserver model

In this paper, a multiserver system is treated as an $M/G/c/k$ queuing system which is elaborated as follows. There are c servers that run at an identical speed s_p (measured by the number of instructions that can be executed in one unit of time). Stream of service requests follow a Poisson process, which means that task inter-arrival time A is exponentially distributed with a rate of λ (> 0). We denote its Cumulative Distribution Function (CDF) as $A(x) = P[X \leq x]$ and its Probability Density Function (PDF) as $a(x) = \lambda e^{-\lambda x}$. Laplace Stieltjes Transform (LST) of inter-arrival time is

$$A^*(s) = \int_0^\infty e^{-sx} a(x)dx = \frac{\lambda}{\lambda + s}.$$

A multiserver system maintains a queue with finite capacity. The First-Come-First-Served (FCFS) queuing discipline is adopted, and when an incoming service request finds the system full, it will be lost. The task execution requirements (measured by the number of instructions to be executed) are i.i.d. general random variables r with mean E(r). Since the server execution speed is s_p, the service times of the requests are also i.i.d. general random variables $y = \frac{r}{s_p}$ with a CDF $H(y) = P[Y \leq y]$ having finite mean $E(H) = \frac{E(r)}{s_p}$ and independent of the arrival process. The PDF is $h(y)$ and the LST of service time is

$$H^*(s) = \int_0^\infty e^{-sy} h(y)dy.$$

Let cv^2 be the squared coefficient of variation (variance divided by the square of the mean) of H, and let $\rho = \frac{\lambda E(H)}{c}$ be the traffic intensity.

A. Blocking probability

Let $\pi = (\pi_0, \pi_1, ..., \pi_k)$ the equilibrium probability distribution of the number of service requests present at the arrival instants [9], where $\pi_i = \lim_{n \to \infty} P(X_n = i)$ with $0 \leq i \leq k$, and X_n represents the stochastic process of the number of service requests present in the system at the arrival instants. From this equilibrium probability distribution, the fraction of rejection of the arrival service requests (blocking probability), P_k, can be calculated as:

$$P_k = \pi_k,$$

and the effective arrival rate can be obtained as:

$$\lambda' = \lambda(1 - P_k).$$

B. Abandonment probability

Customers decide to abandon the cloud service with probability P_a or accept this cloud service with probability $1 - P_a$. Then, the expected abandonment rate λ_a can be obtained as:

$$\lambda_a = \lambda' \times P_a,$$

where P_a is equal to product of mean waiting time, $E(W)$, and potential abandonment index d, with $d \in [0, 1]$. The mean final arrival rate, λ_f, that really wants to receive service can be defined as:

$$\lambda_f = \lambda' - \lambda_a.$$

Example 1: $M/G/20/52$ Queue
In this example, we calculate the abandonment probability of customer according to the number of tasks in the system.

Number of tasks found in the system L_n	P_a
21	0.004
25	0.10
32	0.33
35	0.44
44	0.74
49	0.91
52	1.00

As we see in this table, the abandonment probability increases with the increase of the number of customers in the system. In other words, when the number of customers increases in the system, the mean waiting time also increases; therefore, some customers may feel that waiting time is too long to endure and they will abandon the system without obtaining service.

III. Mean waiting time of the new service request arrival

In this section, we propose an analytical formula to compute the mean waiting time of a new service request arrival to the system. To do this, we assume that all c servers were beginning service when the nth service request arrived to the system. Let L_n the number of service requests found in the system when the nth service request arrived. For computing the mean waiting time of this nth service request, we first consider two random variables W and V which represent the actual and the virtual waiting times, respectively.

Given that $W > 0$, the virtual waiting time V can be decomposed as:

$$V = V_R + V_q, \tag{1}$$

where:

- V_R is the smallest of the remaining service times of c service requests in service,
- V_q is the time until all service requests waiting in the buffer enter the service facility.

And $V_R = V_q = 0$, when $W = 0$.

According to the PASTA (Poisson Arrivals See Time Averages) property, it is obvious that

$$E(V) = E(W) = E(V_R) + E(V_q). \tag{2}$$

We assume that the time required to empty the system with c servers is c times smaller than with a single server when all c servers are busy. Then, we can used this approximation:

$$E(V_q) \simeq \frac{E(H)}{c}(\lambda' E(W)). \tag{3}$$

Using (1), (2) and (3), we obtain

$$E(W) \simeq P(W > 0)E(V_R|W > 0) + \frac{E(H)}{c}(\lambda' E(W)). \tag{4}$$

And hence,

$$E(W) \simeq \frac{P(W > 0)E(V_R|W > 0)}{1 - \frac{E(H)}{c}\lambda'}. \tag{5}$$

To compute $E(V_R/W > 0)$, we will show some asymptotic properties that are useful for approximating this quantity. However, to compute $P(W > 0)$, we will provide an analytical formula in the next section.

Let H_e be the stationary-excess CDF associated with the service-time CDF H, where

$$H_e(y) = \frac{1}{E(H)} \int_0^y \big(1 - H(u)\big)du, \quad y \geq 0,$$

and let

$$I_H(c) = \int_0^\infty \big(1 - H_e(y)\big)^c dy, \quad c \geq 1.$$

Then, we have the following lemma:

Lemma 1: [10]
For $c \geq 1$,

$$\lim_{\rho \to 0} E(V_R|W > 0) \simeq I_H(c) \tag{6}$$

and

$$\lim_{\rho \to 1} E(V_R|W > 0) \simeq \frac{(1 + cv^2)}{2c}E(H). \tag{7}$$

Proof 1: The light and heavy traffic properties (6) and (7) are direct consequences of the limit theorems of [11] and [12], respectively.

We suppose that a cloud provider tries to keep the traffic intensity up as much as possible in order to make the most of prepared installed infrastructure, hence, we assume that the traffic intensity is rather high, $\rho \to 1$. Consequently, we consider the property (7) in the computation of the mean waiting time of the nth service request arrival. Thus, we obtain the following formula:

$$E(W) \simeq \frac{P(W > 0)}{2(\frac{c}{E(H)} - \lambda')}(1 + cv^2). \tag{8}$$

IV. Delay probability

Let's now compute the $P(W > 0)$. For this, we consider the indicator function $U(n)$:

$$U(n) = \begin{cases} 1, & if \quad L_n \geq c; \\ 0, & if \quad L_n < c. \end{cases} \tag{9}$$

We have:

$$\begin{aligned} E(U(n)) &= 1P(L_n \geq c) + 0P(L_n < c) \\ &= P(L_n \geq c). \end{aligned} \tag{10}$$

As the arrivals follow a Poisson process, the nth service request sees upon arrival L_n other service requests in the system, i.e. Poisson arrivals see time averages. Hence,

$$P(W > 0) = P(L_n \geq c). \tag{11}$$

Also, due to PASTA property, the delay probability, π_W, (i.e. the probability that a newly submitted service request must wait because all servers are busy) is equal to

$$\pi_W = P(W > 0). \tag{12}$$

Thus, it is enough to compute π_W in order to compute $P(W > 0)$. To do this, we firstly use the property (7) and we obtain:

$$\begin{aligned} \frac{(1 + cv^2)}{2c}E(H) &= \frac{1}{c} \times \frac{(1 + cv^2)}{2}E(H) \\ &= \frac{1}{c} \times E(H_+), \end{aligned} \tag{13}$$

where H_+ is the time interval from an arbitrary point during a service time to the end of the service time, i.e., the residual service time, and $E(H_+)$ is its mean. The LST of H_+ is given in [13] as:

$$H_+^*(s) = \frac{1 - H^*(s)}{sE(H)}. \tag{14}$$

Then, we define three probabilities P_x, P_y and $P_{z,k}$ as follows:

- P_x, the probability of completing the service residual of a cloud service request which found in the service when the nth cloud service request arrived to the system.

$$P_x \triangleq P(A > H_+) = H_+^*(\lambda). \tag{15}$$

- P_y, the probability of completing the service of a cloud service request before a new cloud service request arrives to the system.

$$P_y \triangleq P(A > H) = H^*(\lambda). \tag{16}$$

- If a server completes the service of cloud service request, this server will be idle. If the buffer is nonempty, this server as well may complete a second service, and if the buffer is still nonempty, a new service may be completed, and so on until the buffer gets empty or new cloud service request arrives. Thus, the probability of k services completed by a single server is given by:

$$\begin{aligned} P_{z,k} &= \Big[\prod_{i=2}^{k} P\Big(A > H | A > (k-i)H + H_+\Big)\Big] \\ &\quad \times \Big[P(A > H_+)\Big]. \end{aligned} \tag{17}$$

Note that the definitions of P_x, P_y and $P_{z,k}$ are same as in [9].

As we know, when the nth service request finds $(L_n \geq c)$ service requests in the system with c servers, it has to wait in the queue until the completion of service of $(L_n - c + 1)$ service requests. But, in this paper, as we assumed that the time required to empty the system with c servers is c times smaller than with a single server when all c servers are busy, and based on the mathematical analysis of the $M/G/c/k$ queue, we demonstrate that the number of service requests N which could be completed their services so that the nth service request begins its service is:

$$N = c \times E[\frac{L_n}{c}], \tag{18}$$

where $E[\frac{L_n}{c}]$ is the integer part of $\frac{L_n}{c}$.

Thus, by using (14), (15), (17) and (18), we obtain the formula of the delay probability π_W as follows:

$$\pi_W = \sum_{L_n=c}^{k-1} \Big[P_x P_{z,2} \ldots P_{z,E(\frac{L_n}{c})}\Big]. \tag{19}$$

V. Revenue and cost analysis

A. Service charge

In this paper, the service level is reflected by the waiting time of requests. Hence, we define the service charge function for a service request with mean execution requirement $E(r)$ and mean waiting time $E(W)$ as follows:

$$C = \begin{cases} aE(r), & if \quad 0 \leq E(W) \leq D; \\ 0, & if \quad E(W) > D. \end{cases} \tag{20}$$

(i) D: deadline. If the mean waiting time of the new service request $E(W)$ exceed D, a SLA (Service Level Agreement) violation event occurred.

(ii) If $E(W)$ is no longer than D, then the service provider considers the service request is processed successfully with high quality of service and charges a customer $aE(r)$, where a is the service charge per unit amount of service.

(ii) If $E(W)$ is longer than D, then a service provider considers that the service request has been waiting too long, so there is no charge and the service is free.

B. Expected revenue of a service provider

Since the proposed cloud service model is provided with finite capacity and impatient customers (i.e. some customers may feel that waiting time is too long to endure and they will abandon the system without obtaining service), then, the number of service requests processed in one unit of time is λ_f, the expected revenue of a service provider R per unit time can be written as:

$$R = \lambda_f C. \tag{21}$$

C. Power consumption cost

Power dissipation and circuit delay in digital CMOS circuits can be accurately modeled by simple equations, even for complex microprocessor circuits. CMOS circuits have dynamic, static, and short-circuit power dissipation; however, the dominant component in a well-designed circuit is dynamic power consumption P. In this paper, we adopt the following dynamic power model similar to one adopted in [1]:

$$P = aCV^2 f, \tag{22}$$

where a is an activity factor, C is the loading capacitance, V is the supply voltage, and f is the clock frequency [14]. In the ideal case, the supply voltage and the clock frequency are related in such a way that $V \propto f^{\phi}$ for some constant $\phi > 0$ [15]. The processor execution speed s_p is usually linearly proportional to the clock frequency, namely, $s_p \propto f$. For ease of discussion, we will assume that $V = bf^{\phi}$ and $s_p = zf$, where b and z are some constants. Hence, The power consumption P can be written as $P = aCV^2 f = aCb^2 f^{2\phi+1} = aCb^2(\frac{s_p}{z})^{2\phi+1} = \gamma s_p^{\alpha}$ where $\gamma = \frac{aCb^2}{z^{2\phi+1}}$ and $\alpha = 2\phi + 1$. We set $b = 1.16$, $aC = 7.0$, $z = 1.0$ and $\phi = 0.5$, $\alpha = 2.0$ and $\gamma = 9.4192$, hence, the value of P calculated by the equation (22) is close to the value of the Intel Pentium M processor ([16]).

Since the power for speed s_p is γs_p^{α}, the average amount of energy consumed by a server in one unit of time is $\frac{\lambda_f E(H)}{c}\gamma s_p^{\alpha} = \frac{\lambda_f E(r)}{c}\gamma s_p^{\alpha-1}$. The average amount of energy consumed by an c-server system in one unit of time is $c\frac{\lambda_f E(r)}{c}\gamma s_p^{\alpha-1} = \lambda_f E(r)\gamma s_p^{\alpha-1}$, where $\lambda_f E(r)$ is the average number of busy servers. Let P^* the power consumption when a server is idle and, let ε the cost of energy per Watt, then, the total cost of energy consumption of the c-server system in one unit of time under c servers is

$$C_{\text{pcc}} = \big(\lambda_f E(r)\gamma s_p^{\alpha-1} + cP^*\big)\varepsilon. \tag{23}$$

D. Profit function

Before defined the profit function, F, we assume that the rental cost of one server for unit of time is C_{rc}. The cost of a service provider is the sum of the cost of infrastructure renting and the cost of energy consumption, i.e., $C_{\text{pcc}} + cC_{\text{rc}}$. Then, the expected profit function per unit time is:

$$\begin{aligned} F &= R - \big(C_{\text{pcc}} + cC_{\text{spc}}\big) \\ &= \lambda_f C - \big(\big(\lambda_f E(r)\gamma s_p^{\alpha-1} + cP^*\big)\varepsilon + cC_{\text{rc}}\big). \end{aligned} \tag{24}$$

The result presented in Figure 1 show revenue R and the profit F for different values of λ when the task execution time follows the gamma distribution. Note that the choice of the gamma distribution remains a proposition, our model can accommodate other distributions without any change.

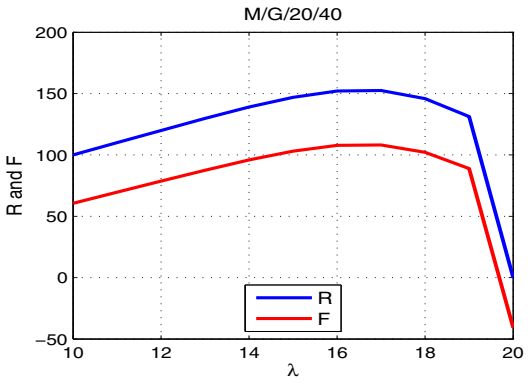

Fig. 1. Revenue R and Profit F versus λ.

As can be seen in this Figure, the revenue R and the profit F in one unit of time as a function of λ where $c = 20$, $k = 40$, $E(r) = 1$ billion instructions, $s_p = 1$ billion instructions per second, $a = 10$ unit per one billion instructions, $D = 1$ second, $d = 0.5$, $\alpha = 2.0$, $\gamma = 9.4192$, $\varepsilon = 0.1$ unit per Watt and $C_{\text{rc}} = 1.5$ unit per second. We observe that both R and F increase with λ almost linearly, then progressively decrease, and finally drop sharply after certain point. In other words, more service requests bring more revenue and profit; as the queuing length increase with the increase of service requests, some customers leave the system without obtaining service, which will directly result in revenue losses for a cloud provider; however, after the number of service requests reaches certain point, the mean waiting time exceed the deadline D and the SLA violation event occurred. Therefore, there is no revenue and the profit is negative.

VI. Profit Maximization

Our optimization problem cannot be solved in closed form, therefore we have to resort to a numerical solution and all computational programs are coded by MATLAB.

A. Optimal size

Given λ, s_p, k, $E(r)$, a, D, d, α, γ, ε and C_{rc}, our objective is to find c such that profit is maximized. In Figure 2, we demonstrate the profit F in one unit of time as a function of c and λ, using the same parameters in Figure 1. We notice that there is an optimal choice of c such that F is maximized. For $\lambda = 14.9, 13.9, 12.9$, the optimal value of c is $20, 19, 17$, respectively.

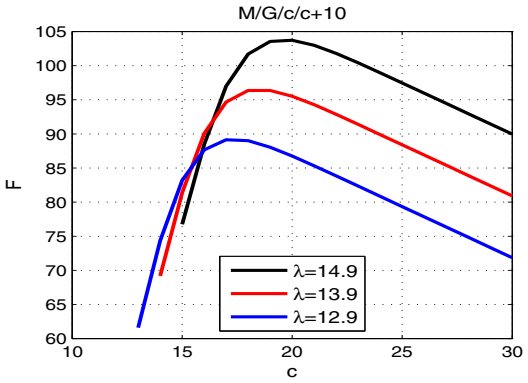

Fig. 2. Profit F versus c and λ.

As can be see in the Figure 2, when c is small, the profit F is low. Which explains that when c is small, the blocking probability and the abandonment probability increase, which result in revenue losses for a cloud provider. Hence, the revenue R is low as well as the profit F. However, as c further increases, the blocking probability and the abandonment probability are significantly decrease, but the cost of a service provider (i.e., the rental cost and base power consumption) increases, so that the profit is actually reduced. Hence, there is an optimal choice of c which maximizes the profit.

B. Optimal speed

Given λ, c, k, $E(r)$, a, D, d, α, γ, ε and C_{rc}, our objective is to find s_p such that profit is maximized. In Figure 3, we demonstrate the profit F in one unit of time as a function of s_p and λ, using the same parameters in Figures 1 and 2. We notice that there is an optimal choice of s_p such that F is maximized. For $\lambda = 14.9, 13.9, 12.9$, the optimal value of s_p is $1.05, 1.00, 0.95$, respectively.

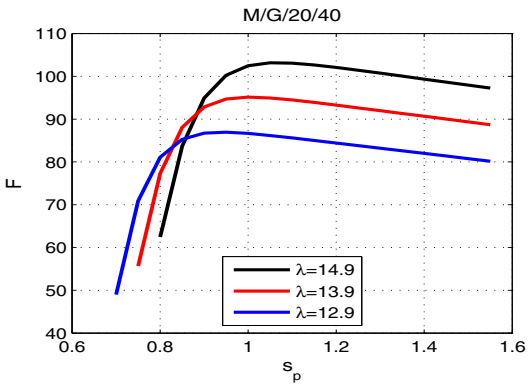

Fig. 3. Profit F versus s_p and λ.

As can be see in the Figure 3, when s_p is small, the profit F is low. Which explains that when s_p is small, the blocking probability and the abandonment probability increase, which result in revenue losses for a cloud provider. Hence, the

revenue R is low as well as the profit F. However, as s_p further increases, the cost of energy consumption increases. Hence, the increased revenue is much less than the increased cost. As a result, the profit is reduced. Therefore, there is an optimal choice of s_p such that the profit is maximized.

C. Optimal size and speed

Given λ, k, $E(r)$, a, D, d, α, γ, ε and C_{rc}, our objective is to find c and s_p such that profit is maximized. In Figure 4, we demonstrate the profit F in one unit of time as a function of c and s_p, using the same parameters in Figures 1, 2 and 3, where $\lambda = 14.9$. The optimal value is $c = 17$ and $s_p = 1.26$, which result in the maximal profit $F = 104.63$.

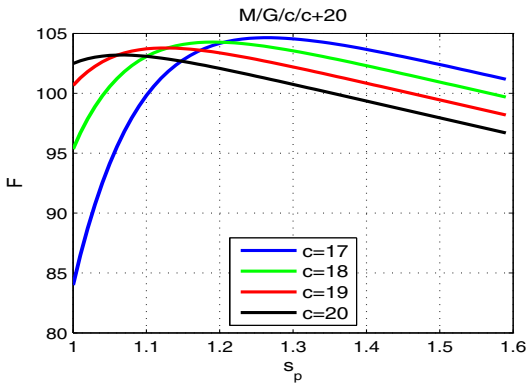

Fig. 4. Profit F versus s_p and c.

VII. Conclusion

As cloud computing becomes more and more popular, understanding the economics of cloud computing becomes critically important. From cloud service provider's perspective, profit is one of the most important considerations, and it is mainly determined by the characteristics of the applications and the configuration of a multiserver system. In this paper, the problem of optimal multiserver configuration for profit maximization in a cloud computing environment is studied. Our valuation model takes such factors into considerations as the amount of a service, the service-level agreement, the quality of a service, the cost of renting, the cost of energy consumption, a service provider's margin and profit, etc. We treated a multiserver system as a queuing model with finite capacity and impatient customers. Also, due to the nature of the cloud environment, we assumed general service time for service requests, which makes our model flexible in terms of diversity of service time. According to our related works, there are no analytical formula to calculate performance measures of $M/G/c/k$ queuing model. Hence, a portion of this work is devoted to the calculation of mean waiting time of the new service request arrival to the system and the delay probability. We focused on the calculation of these latter, because our performance metric in this paper is the task waiting time. Then, we formulated and solved the problem of optimal multiserver configuration for profit maximization in a cloud computing environment. Our study can be easily extended to consider the profit maximization problem in a heterogenous cloud environment.

Acknowledgment

References

[1] J. Cao, K. Hwang, K. Li, and A. Y. Zomaya, *Optimal Multiserver Configuration for Profit Maximization in Cloud Computing*, IEEE Transactions on Parallel and Distributed Systems, vol. 24, no. 6, pp. 1087-1096, 2013.

[2] Y. J. Chiang and Y. C. Ouyang, *Profit Optimization in SLA-Aware Cloud Services with a Finite Capacity Queuing Model*, Mathematical Problems in Engineering, vol. 2014, pp. 01-11, 2014.

[3] J. Mei, K. Li, A. Ouyang, and K. Li, *A profit maximization scheme with guaranteed quality of service in cloud computing*, IEEE Transactions on Computers, vol. 64, no. 11, pp. 30643078, 2015.

[4] M. Mazzucco, D. Dyachukand and R. Deters, *Maximizing cloud providers revenues via energy aware allocation policies*, IEEE 3rd International Conference on Cloud Computing, pp. 131138, 2010.

[5] H. Khazaei, J. Misic, and B. M. Vojislav, *Performance Analysis of Cloud Computing centers using $M/G/m/m + r$ queueing systems*, IEEE Transactions on Parallel and Distributed Systems, vol. 23, no. 5, pp. 936-943, 2012.

[6] H. Kimura, *Approximations for the delay drobability in the M/G/s Queue*, Mathematical and Computer Modelling, vol. 22, no. 10-12, pp. 157-165, 1995.

[7] J. M. Smith, *M/G/c/K Blocking Probability Models and System Performance*, Journal of Performance Evaluation, vol. 52, pp. 237-267, 2003.

[8] S. A. Nozaki and S . M. Ross, *Approximations in Finite-Capacity Multi-Server Queues with Poisson Arrivals*, Journal of Applied Probability, vol. 15, no. 4, pp. 826-834, 1978.

[9] A. Outamazirt, M. Escheikh, D. Aïssani, K. Barkaoui and O. Lekadir, *On the Modeling and Performance Evaluation of Cloud Computing Centers Using M/G/c/c+r Queuing System*, Proceedings of the 10th Workshop on Verification and Evaluation of Computer and Communication System (VECoS 2016), Published on CEUR-WS: http://ceur-ws.org/Vol-1689/, pp 77-84, 2016. Proceedings of the 10th Workshop on Verification and Evaluation of Computer and Communication System, Tunis, Tunisia, pp. 77-84, 2016.

[10] T. Kimura, *Diffusion Approximation for an M/G/m Queue*, Journal of Operations Research, vol. 31, issue 2, pp. 304-321, 1983.

[11] D. Y. Burman and D. R. Smith, *A light-traffic theorem for multi-server queues*, Mathematics of Operations Research, vol. 8, no. 1, pp. 15-25, 1983.

[12] J. Kllerstrm, *Heavy traflic theory for queues with several servers I*, Journal of Applied Probability, vol. 11, no. 3, pp. 544-552, 1974.

[13] L. Kleinrock, *Queueing Systems*, vol. 1, Theory. Wiley-Interscience, 1975.

[14] A.P. Chandrakasan, S. Sheng and R.W. Brodersen, *Low-Power CMOS Digital Design*, IEEE Journal Solid-State Circuits, vol. 27, no. 4, pp. 473-484, 1992.

[15] B. Zhai, D. Blaauw, D. Sylvester and K. Flautner, *Theoretical and Practical Limits of Dynamic Voltage Scaling*, Proc. 41st Design Automation Conf., pp. 868-873, 2004.

[16] *Enhanced Intel speedstep technology for the Intel Pentium M processor*, White Paper, 2004.