Performance Analysis of End-to-End Services in Virtualized Computing Environments*

Guofeng Yan[1,2] and Yuxing Peng[2]

[1] School of Computer and Communication, Hunan Institute of Engineering, China
[2] Science and Technology on Parallel and Distributed Processing Laboratory, National University of Defense Science and Technology, Changsha 410073, China
{gfyan,pengyuxing}@nudt.edu.cn

Abstract. In this paper, we present a novel stochastic analyzing model for e2e virtualized cloud services using hierarchical Quasi-Birth Death structures (QBDs). We divide the overall virtualized cloud services into three sub-hierarchies, and then, analyze each individual sub-hierarchy using QBDs. Our approach reduces the complexity of performance analysis. Our results are useful to prevent the cloud center from entering unsafe operation, and also reveal practical insights into load balancing and capacity planning for virtualized computing environments. ...

1 Introduction

Theoretical analyses on cloud services mostly rely on extensive research in performance evaluation of M/G/m queuing systems, as outlined in [1]. Using the distribution of response time, researchers discover the relationship among the maximal number of requests, the minimal service resources and the highest level of services [2]. However, as solutions for distribution of response time and queue length in M/G/m systems cannot be obtained in closed form, suitable approximations are sought. To ensure that the quality of service (QoS) perceived by end clients is acceptable, in [3], the performance of cloud server farms with general service time is analyzed. The researchers propose a general analytic model for e2e performance of cloud services. However, the proposed model is limited to the single arrival of requests and the start up delay of cold physical machines (PMs) has not been captured. The effect of virtualization on e2e cloud QoS need to be further studied based on these previous research.

2 System Model

We assume that L different servers, M distinct users, and N types of requests for each user. A type-k request, Req_k, is specified a type-k VM-*configuration*

* This work was supported by the National 973 Basic Research Program of China under Grant Number 2011CB302601, China Postdoctoral Science Foundation under Grant Number 2013M542561, and Scientific Research Fund of Hunan Provincial Education Department under Grant 14B040.

VM_k, and VM_k^i is the provisioning VM_k on server i. Assume that a global resource provisioning and deploying decision machine (RP&DDM) processes requests on a FCFS principle in our system and each request arrives stochastically at RP&DDM, and we define the size of the request Req_k as $|Req_k|$. Let $\mathbf{S} = \{active, passive\}$ be the state set of RP&DDM. In *active*, an arriving request can be served immediately; in *passive*, an arriving request can only be processed after PMs are redeployed. Assume that the requests of user k arrive according to a Poisson process with rate λ_k ($\lambda_1 = \cdots = \lambda_M$), each server maintains N different queues (i.e., $q_{i1}, q_{i2}, \cdots, q_{iN}$) for N different types of requests, and the processing time of Req_k on each server is exponentially distributed with parameter μ_k ($\mu_1 = \cdots = \mu_N = \mu$).

3 QBDs Stochastic Model for Cloud Computing System

Let $\lambda = \sum_{i=1}^{M} \lambda_i = M\lambda_i$ and Q_{req} denote the finite queue of all requests. We consider two state spaces: Λ and Λ_k^i. $\Lambda = \{Y(t), s_t\}$ describes the general characters of RP&DDM, where $Y(t) = 0, 1, 2, \cdots, Q$ denotes the number of requests in Q_{req}, and $s_t \in \mathbf{S}$ refers to the state of RP&DDM at time t. $\Lambda_k^i = \{(Y_k^i(t), r_{kj}^i, s_t^i) : j = 1, 2, \cdots, Q; r_{kj}^i \geqslant 0; s_t^i = 0, 1\}$ captures the characters of type-k request on server i, where r_{kj}^i, $Y_k^i(t)$ and s_t^i refer to the remaining size of the jth type-k request on server i, the the number of requests in q_{ik} at time t, and the current state of server i, respectively. Let $s_t = 0$ and $s_t = 1$ denote that s_t is *passive* and *active*, respectively. Then, each state of Λ can be expressed as a combination $(Y(t), s_t)$. We compute the transition rates of the QBDs according to [4]. Let the probabilities of RP&DDM being in *active*, and *passive* at time τ_i be $1 - e^{-p(\tau_i)}$, and $e^{-p(\tau_i)}$, respectively. Hence, we obtain the transition probabilities of RP&DDM from time τ_i to τ_{i+1} from $p_{00}^{\Lambda} = e^{-p(\tau_i)-p(\tau_{i+1})}$, $p_{01}^{\Lambda} = e^{-p(\tau_i)} - e^{-p(\tau_i)-p(\tau_{i+1})}$, $p_{11}^{\Lambda} = 1 - e^{-p(\tau_i)} - e^{-p(\tau_{i+1})} + e^{-p(\tau_i)-p(\tau_{i+1})}$, and $p_{10}^{\Lambda} = e^{-p(\tau_{i+1})} - e^{-p(\tau_i)-p(\tau_{i+1})}$.

For the state space Λ_k^i, let $r_{kj}^i(t)$ be the remaining request size of the jth type-k request on server i at time t, and $r_k^i(t)$ the queue state of type-k request on server i. Then, $Y_k^i(t) = \{r_k^i(t)\}_{k,i}$ is QBDs on Λ_k^i [1].

4 Performance Evaluation

Rejection Probability of $\boldsymbol{Req_k}$**.** Let p_{reject}^{full} and $p_{reject}^{passive}$ be the rejection probabilities of Req_k due to Q_{req} full and no *active* VM-*configuration*, respectively. Assume that s and $\pi_{(Q,s)}$ are the state of RP&DDM and the stationary state probability of the first hierarchical QBDs, respectively, $\frac{1}{\gamma_{passive}}$ is the mean searching delay to find a *passive* server, $\pi_{ik}(n, passive)$ and λ_{ik} are the stationary state probability. We can obtain p_{reject}^{full} and $p_{reject}^{passive}$ from $p_{reject}^{full} = \frac{1}{N} \sum_{s \in \{active, passive\}} \pi_{(Q,s)}$ and

$p_{reject}^{passive} = \sum_{i=0}^{L} \sum_{n=0}^{Q} \frac{\gamma_{passive} \cdot 1 - \pi_{passive}^{i} \cdot \pi_{ik}(n, passive)}{\lambda_{ik}}$. Then, the rejection probability p_{reject} is:

$$p_{reject} = \sum_{s \in \{active, passive\}} \pi_{(Q,s)} + \sum_{i=0}^{L} \sum_{n=0}^{Q} \frac{\gamma_{passive} \cdot p_{passive}^{i} \cdot \pi_{ik}(n, passive)}{\lambda_{ik}} \quad (1)$$

We use simulations to evaluate the acceptance probability (i.e., $1 - p_{reject}$). Let $L \in \{50, 100, 150, 200\}$ and $M \in [50, 600]$ users. We carry out our simulation with $\delta = 0.75$, $\lambda_k = 0.05$, $\mu = 0.5$, and $1/\gamma_{passive} = 5$. The experiment results and the calculative results are shown in Fig. 1. From Fig. 1, the calculating results of our analytical model and the simulation results are very similar. When $M > 350$, the acceptance probabilities seriously decrease with the increasing of M and it means that the system capacity of cloud computing is not enough for more than 350 users. Furthermore, we find that the acceptance probabilities for $L = 200$ are steady for all M and more than 0.95. These results show the benefits of adding more servers are reflected by having higher acceptance probabilities of user requests for a fixed μ and λ_k.

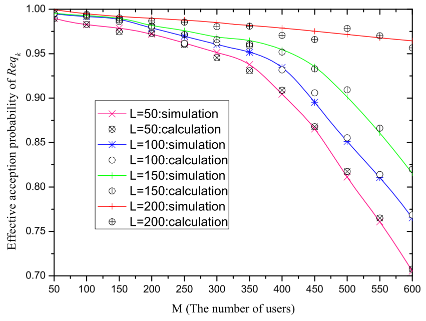

Fig. 1. Effective request acceptance probability of Req_k *vs* different number of users with $\mu = 0.5$ and $\lambda_k = 0.05$

E2e Response Delay. Let T_w denote the waiting time in the steady state, and $W(x)$ and $W^*(y)$ be the CDF of T_w and its LST, respectively. Let $f(z)$ be the generation function of Q_l, and we can obtain $f(z) = W^*(\lambda(1 - z))$. Let $z = 1 - y/\lambda$ and we have $W^*(y) = \sum_{k=0}^{k=L-1} \pi_{(k,s)} + \sum_{k=L}^{k=2L} \pi_{(k,s)}(1 - y/\lambda)^{k-L}$ according to [1]. Hence, we get $E[T_w] = \sum_{k=L}^{k=2L} \frac{(k-L)\pi_{(k,s)}}{\lambda}$.

Similarity, let T_d and T_e be the resource deploying delay and the mean request executing time, respectively. Then, $E[T_d]$ and T_e can be calculated by $E[T_d] = \frac{1/\gamma_{active} + p_{passive}^{i}/\gamma_{passive}}{p_{accept}}$ and $T_e = \sum_{k=1}^{N} \frac{\lambda_{ik}(L\delta p_{11}^{t} + L(1-\delta)p_{01}^{t})}{\lambda \mu_k}$, respectively, where $1/\gamma_{active}$ and $1/\gamma_{passive}$ are the mean search delay when server i being in states *active* and *passive*. Therefore, the mean e2e delay of Req_k, T_{e2e}, is:

$$T_{e2e} = E[T_w] + E[T_d] + E[T_e] \quad (2)$$

The calculating results and simulation results in Fig. 2 show increasing the number of users will increase the response delay. For a small cloud server cluster (for example, $L < 50$), increasing the number of user will increase rapidly the response time of request. But for a large cloud server cluster (for example, $L > 200$), the response delay of request does not distinctly varies due to its enough capacity. Based on these information, cloud computing systems can achieve an more effective admission control policy to guarantee e2e cloud QoS.

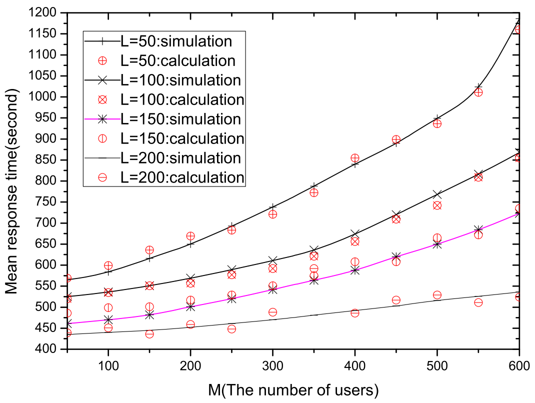

Fig. 2. Mean response time *vs* different number of users with $\mu = 0.5$ and $\lambda_k = 0.05$

5 Conclusion

We analyze the effect of virtualization on the IaaS cloud service quality. Our model is flexible in terms of scalability and diversity of requests and cloud computing clusters. However, we do not consider the remaining time of time slots when a request is finished before the given time slot terminates. Therefore, how to improve the precision of the analytical results is our further work.

References

1. Ma, B.N.W., Mark, J.W.: Approximation of the mean queue length of an M/G/c queueing system. Operations Research 43, 158–165 (1998)
2. Yang, Y., Zhang, Y., Wang, A., et al.: Quantitative survivability evaluation of three virtual machine-based server architectures. Journal of Network and Computer Applications 36, 781–790 (2013)
3. Khazaei, H., Mišić, J., MiMišić, V.B.: Performance analysis of cloud computing centers using M/G/m/m+r queueing systems. IEEE Transactions on Parallel and Distributed Systems 23(5), 936–943 (2012)
4. Yan, G.F., Wang, J.X., Chen, S.H.: Performance analysis for (X, S)-bottleneck cell in large-scale wireless networks. Information Processing Letters 111, 267–277 (2011)