Journal of Theoretical and Applied Information Technology
31st October 2012. Vol. 44 No.2

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

RESEARCH ON PERFORMANCE MODELING OF TRANSACTIONAL CLOUD APPLICATIONS

[1]XIAOYING WANG, [2]XUHAN JIA, [3]LIHUA FAN, [4]WEITONG HUANG

[1] Department of Computer Technology and Applications, Qinghai University, Xining 810016, China

[2] Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China

E-mail: [1]wxy_cta@qhu.edu.cn , jiaxh@qhu.edu.cn , fanlh@qhu.edu.cn , [2]huangwt@tsinghua.edu.cn

ABSTRACT

As cloud computing has gained a lot of attention recently, performance modeling of cloud applications would be very important for various management issues, such as capacity planning and resource provisioning for the cloud providers. This paper conducts research on several performance modeling approaches of transactional cloud applications. Steady-state models are established for both single-tier and multi-tier applications, and then enhanced models are studied on the basis of basic models. Simulation experiments are designed and performed to validate the models we proposed, which illustrates that the measured results fit the analytic models very well.

Keywords: *Performance Modeling, Transactional Applications, Cloud Computing*

1. INTRODUCTION

Nowadays, cloud computing has become an emerging computing model by which users can gain access to their applications from anywhere, through any connected device [1]. The concept incorporates infrastructure as a service (IaaS), platform as a service (PaaS) and software as a service (SaaS) as well as Web 2.0 and other recent technology trends [2] which have the common theme of reliance on the Internet for satisfying the computing needs of the users [3]. By providing unbounded scale and differentiated quality of service, the cloud computing model can help improve business performance and control the costs of delivering IT resources to other organizations.

In cloud environments, performance issues are always concerned by users and resource providers. The QoS (Quality of Service) [4] is one of the most important metrics, which emphasizes end-to-end performance and usually can be quantified by response time, rejection ratio, and throughput and so on. Then, a clear and feasible performance model for cloud applications is very important, which can help to estimate the possible achieved performance according to the input resource amount and load information. However, modern applications are often built on complex architecture or running on multiple server groups, which makes the modeling work more difficult. In this paper, we attempt to conduct a series of research on performance modeling of cloud applications. Several common techniques for different scenarios are introduced and the drawbacks are discussed and improved. Much prior work has incorporated queuing theory to analyze the performance of Web applications. Here, we enhanced the queuing models in order to fit the executing system better.

2. RELATED WORK

Performance modeling is widely-used in various research areas related to QoS management and dynamic resource allocation. For instance, Bennani et al. [5] tried to solve the resource management problem of multiple application environments in large-scale datacenters. Chandra et al. [6] studied on the dynamic resource allocation issues on the basis of GPS (General Processor Sharing) mode. Menascé et al. [7] used similar settings and modified the model according to different resource allocation schemes – priority and CPU sharing. Furthermore, more and more recent work began to discuss complex application architectures and has proposed corresponding models for different scenarios. Menascé et al. [8] described how to monitor and tune E-commerce sites to keep certain QoS levels, in which close queuing network models are employed considering both CPU and memory resources. Urgaonkar et al. [9] proposed a flexible dynamic resource provisioning strategy to deal with varying workload using a G/G/1 queuing system to model each server. Doyle et al. [10] proposed a

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

model-based method to manage the resources aiming at different performance levels, which mainly considers memory-related resources. In contrast, in this paper we focus on the performance modeling of transactional cloud applications, and the analysis of time-domain and admission control effects is introduced into the original model for enhancement.

3. OVERVIEW OF TRANSACTIONAL CLOUD APPLICATIONS

3.1 Architecture

Usually in the cloud environment, there are many physical nodes constituting the underlying infrastructure. Virtual machines (VMs) serving for different tiers of different cloud applications are deployed above these physical nodes. Notably, VMs serving for the same tier of a certain application often locate on different physical nodes. Moreover, for a certain tier at one application environment, there may be more than one VM configured to serve the requests.

Accordingly, Fig.1 shows an example routing architecture of a typical transactional cloud applications comprised of three tiers, including the web server tier, the application server tier and the database server tier. At each tier, there is a router in charge of dispatching requests to all the VMs serving for this tier according to certain kind of load balancing strategy. After a request is finished at a certain tier, it will be forwarded to the router at the next tier or leave the environment if it finishes at the last tier. Since the capacity of VMs might be changed from time to time, the performance modeling of such applications should incorporate the heterogeneity feature into consideration.

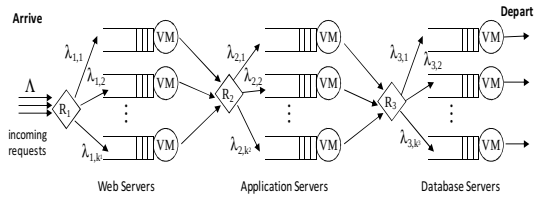


Fig.1 An Example Architecture of Cloud Applications

3.2 Performance Metrics

In the cloud environment, the infrastructure provider should guarantee some non-functional requirements of the customers. SLAs (Service Level Agreements) are usually used for negotiation between service providers and customers, which specify the QoS level and the corresponding cost model. Here, we discuss two typical QoS metrics as follows.

(1) End-to-end Response Time (denoted as *R*)

From the perspective of customers, the total time from when the request sent to the cloud service to when the request finished and results returned to the customer is called "end-to-end response time". It includes both the waiting time in the queue and the processing time in the service nodes. Since most services employ the first-come-first-serve scheduling strategy and the workload intensity varies significantly, the response time is often difficult to predict and control. The end-to-end response time is the most important metric since it directly reflects users' experience. According to the working mechanism of the cloud service, the response time is usually decided by the processing ability of the servers and the current workload situation.

(2) Throughput (denoted as *X*)

The throughput is a metric measuring how many transactions could be finished or how many operations could be done during one time unit. Generally, the throughput will increase as the workload intensity rises up. However, when the resource utilization reaches nearly 100%, the system will be saturated without additional ability to deal with more requests. In this case, the throughput might drop down due to the "thrashing" effect.

3.3 Operational Analysis

Queuing operational analysis is widely used to analyze the performance of complex systems, which provides a relatively simple way to characterize the features and statuses of the executing system quantitatively. The operational analysis gives some basic equations which can be used when characterizing the system. Two typical laws often used in relative research are as follows.

(1) Utilization Law

According to the utilization law, the utilization of a system (*U*) is equal to the product of total throughput (*X*) and the service demand (*D*), namely that $U=X\cdot D$. Here, the service demand means the average processing time of each request.

(2) Little's Law [11]

According to Little's Law, the population (*N*) in the system is equal to the product of the system throughput (*X*) and the end-to-end response time (*R*), namely that $N=X\cdot R$.

Journal of Theoretical and Applied Information Technology
31st October 2012. Vol. 44 No.2

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

4. PERFORMANCE MODELING FOR STEADY STATE

4.1 Single-tier Applications

For simplicity, we start the performance modeling discussion by examining the applications with only one tier. Note that there are still multiple VMs with different capacities providing service for the application. Denote the number of VMs as n. Incoming requests arrive at the environment continuously. Denote the arrival rates of the requests as Λ. The current service rates of the allocated VMs are denoted as (μ_1, μ_2,…,μ_n). According to the queuing theory, we can use several methods to set up the approximate performance model for this scenario as follows.

(1) Single M/M/1 queue. Overall, the simplest way is to setup a single M/M/1 queuing model, where the capacity of the server is the summary of all VM capacities, i.e. $\mu_1+\mu_2+\ldots+\mu_n$. Then, the mean response time could be estimated as $R = 1/\left(\sum_{i=1}^{n}\mu_i - \Lambda\right)$.

(2) Multiple M/M/1 queues. From another point of view, we can separate the incoming requests to several streams, making the arrival rate to the ith VM as λ_i. In this case, the mean response time of the multiple queues should be balanced, which means $\begin{cases} R = 1/(\mu_i - \lambda_i), i = 1,2,...,n \\ \Lambda = \sum_{i=1}^{n}\lambda_i \end{cases}$.

(3) Homogenous M/M/n queue. In order to use an M/M/n queue to approximate the original system, we set the capacity of each VM server, denoted as μ, as the average of all the VM capacities. Then, according to the analytical results of the M/M/n queuing system, the mean response time can be derived as

$$\begin{cases} R = \frac{1}{\mu} + \frac{(n\rho)^n}{(1-\rho)^2 \cdot n\mu \cdot n!\left(\frac{(n\rho)^n}{(1-\rho)\cdot n!} + \sum_{i=0}^{n-1}\frac{(n\rho)^i}{i!}\right)} \\ \mu = \frac{\sum_{i=1}^{n}\mu_i}{n} \end{cases} \quad (1)$$

Where $\rho = \Lambda/(n\mu)$ is the system utilization.

(4) Heterogeneous M/M/n queue. On the basis of homogeneous M/M/n model, we attempt to exploit further to set up more accurate models considering heterogeneous serving capabilities of the VMs. First, we assume that the on-demand scheduler will always decide to send the current request to the fastest VM server. Without loss of generality, we assume the capacities of VMs (μ_1, μ_2,…,μ_n) are sorted in descending order. During the system execution, it might transit from one state to another state, where each state is determined by the current population in the system. According to the Markov state flow equation, the input and output speed of each state should be equal, namely that

$$p_{m-1} \cdot \Lambda = p_m \cdot \sum_{i=1}^{K}\mu_i, K = \min(m,n) \quad (2)$$

where p_m denotes the possibility of the mth state. Then, we can get each p_m by iteration, as follows

$$p_m = p_0 \cdot \frac{\Lambda^m}{\prod_{i=1}^{K}\sum_{j=1}^{i}\mu_j \cdot \left(\sum_{i=1}^{n}\mu_j\right)^{m-K}}, K = \min(m,n) \quad (3)$$

where p_0 is the possibility when the queuing system is totally empty. Since the summary of p_m should be 1 due to the probability law, we can derive the value of p_0 easily as

$$p_0 = 1\Big/\left(\sum_{m=1}^{n}\frac{\Lambda^m}{\prod_{i=1}^{m}\sum_{j=1}^{i}\mu_j} + \frac{\Lambda^n}{\left(1-\Lambda/\sum_{i=1}^{n}\mu_i\right)\cdot\prod_{i=1}^{n}\sum_{j=1}^{i}\mu_j}\right) \quad (4)$$

Then, all the values of p_m (m=1, 2, 3…) could also be calculated. Denote q^w as the average length of the waiting queue, and q^p as the number of requests being processed, then they can be computed as

$$\begin{cases} q^w = \sum_{m=n}^{\infty}(p_m \cdot (m-n)) \\ q^p = \sum_{m=0}^{n-1}(p_m \cdot m) + \sum_{m=n}^{\infty}(p_m \cdot n) \end{cases} \quad (5)$$

Finally, the mean response time could be obtained as $R = (q^w + q^p)/\Lambda$.

4.2 Multi-tier Applications

For multi-tier applications, we can divide the requests staying in the system as two categories according to their states, either being processed or waiting in the queue at a certain tier. Since there are multiple processing units in the multi-tier system, we can employ queuing network [12] to establish the analytic performance model. Here we assume the incoming requests are processed in the order as they arrive at each tier. Assume there are M tiers in the application architecture, and then we can set up a queuing network with M queues, corresponding to each tier respectively. Denote Λ as the arrival

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

rate of incoming requests, and s_i as the service demand at the ith tier. Then, according to the mean value analysis results [13], the expected mean response time of the queuing network can be derived as $R=\sum_{i=1}^{M}\frac{s_i}{1-\Lambda\cdot s_i}$. The above discussions in this section are suitable for the scenario that the system is in steady states, which means that the arrival rate is equivalent to the departure rate during a certain time period long enough. In this case, the throughput X is equal to the arrival rate Λ.

5. MODELING THE PERFORMANCE IN SATURATED SCENARIO

When the arrival rate of the application requests keeps rising and becomes higher than the summarized processing capability of the entire system, the resources will be fully utilized and the queue will become longer and longer. The system enters into a saturated scenario and cannot converge to an equivalent state. In this case, the steady-state performance model couldn't be directly applied. Nevertheless, the workload intensity varies significantly in realistic cloud environments, and thus the overload situation occurs from time to time. Consequently, we have to consider how to model the performance for systems in the saturated scenario.

5.1 Time-domain Analysis

In order to deal with the saturated scenario, we divide the time period into multiple domains, and denote the domain length as T. Assume that there are q^* requests waiting in the queue when the next domain is about to start, and denote the current time as t^*. Since the system utilization reaches nearly 100% in the saturated scenario, the summarized service rates of the system should be $\mu_1+\mu_2+\ldots+\mu_n$. In this case, the length of the waiting queue will continuously rise up, and thus the average queue length q' of the next examined period should be

$$q'=\frac{1}{T}\int_{t^*}^{t^*+T}\left(q^*+\left(\Lambda-\sum_{i=1}^{n}\mu_i\right)\cdot t\right)dt = q^*+\frac{T}{2}\left(\Lambda-\sum_{i=1}^{n}\mu_i\right) \quad (6)$$

In the saturated situation, the system throughput reaches the highest, which is equal to the summarized service rates of all VMs, namely that $X=\sum_{i=1}^{n}\mu_i$. Then, according to Little's Law, the expected mean response time in the next time period could be calculated as $R=(n+q')\Big/\sum_{i=1}^{n}\mu_i$.

5.2 Admission Control

When the incoming workload intensity becomes heavier or some nodes in the system failed, there might be insufficient resource capacity to deal with the heavy load. Although some peaks only last for a short time, the accumulated requests would block the following ones and delay them a lot. To handle such problems, modern systems often adopt admission control methods to refuse some of the incoming requests. The aim of dynamic admission control should be guaranteeing the requests being finished meeting the target specified in SLAs. Hence, when the population of the system reaches a certain threshold, the next incoming request should be refused.

To model the admission control effect, we should first determine the maximum number of requests that the system can hold. Operational analysis could be leveraged here. Denote the maximum resource utilization specified by the service provider as θ, and the target response time as r^{SLA}. Then, according to Little's Law, the maximum permitted population should be calculated as

$$P^{\max}=\theta\cdot\max\left(0,r^{SLA}-\sum_{i=1}^{M}s_i\right)\Big/\max_{i=1..M}(s_i) \quad (7)$$

where s_i denotes the service demand of the multi-tier application at the ith tier and θ can be specified by the service provider to make surplus room for maintenance considerations.

6. VALIDATION EXPERIMENTS

In this section, we conducted a series of experiments to evaluate the models we proposed. During the experiments, the input workloads are generated from a real-world web trace (*App 1*) and a sin load function (*App 2*) respectively. For single-tier application experiments, the average requests sizes of the two applications are 22.5 and 35.1 respectively. The tested system is comprised of four VMs with the capacity of (300, 600, 500, 400). For multi-tier application experiments, the tested cloud environment is comprised of three tiers. Each tier has (5, 5, 7) nodes and the capacities of nodes at different tiers are (350,900,600) respectively.

6.1 Validation of Steady-state Models

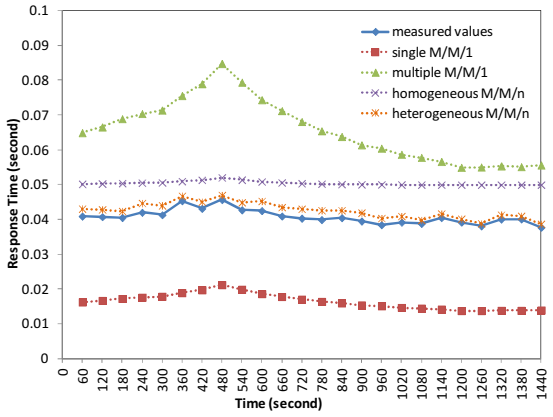
Fig.2 shows the evaluation results of steady-state models. Fig.2 (a) shows the comparison of the

Journal of Theoretical and Applied Information Technology
31st October 2012. Vol. 44 No.2

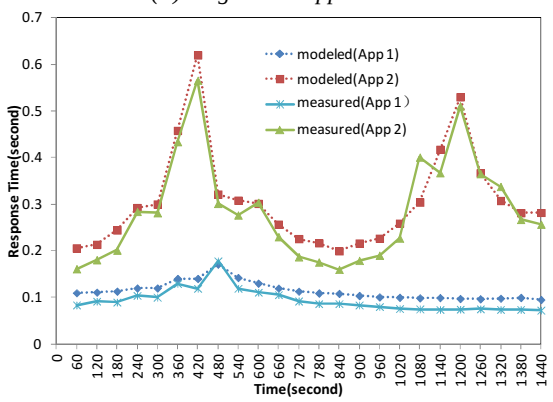
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

results by the four different modeling approximation methods and the realistic values measured, by examining *App 1*. It can be observed that the heterogeneous M/M/n model we set up reaches closest to the measured values.

Fig.2 (b) shows the comparison of response time for multi-tier applications, from which we can see that the results derived from the previously established model could fit the measured results quite well.



(A) Single-Tier Application



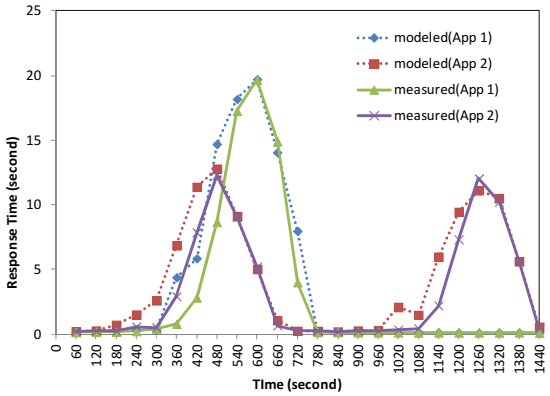
(B) Multi-Tier Applications

Fig.2 Validation Results Of Steady-State Models

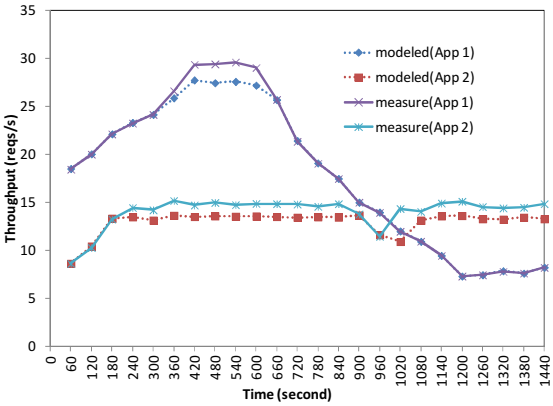
6.2 Validation of Models for Saturated Situation

Fig.3 shows the evaluation results of the established models for saturated situations. Fig.3 (a) illustrates the comparison of the response time of two single-tier applications in saturated situation, where the steady-state models become inapplicable. In contrast, the time-domain analysis could help to predict the performance level when the load becomes too heavy. As seen, the modeled values are coincident with the measured response times.

Moreover, we also conducted experiments for multi-tier applications by examining the throughput metric, and the evaluation results are as shown in Fig.3 (b). Here, the admission control effects exhibit in the figure, especially during the time region (300~800) and (1200~1440). Due to suppression of the excessive requests, the throughput could be controlled and the system could be maintained in a stable state. Also, the estimated throughput values depict as consistent with the results of practical experiments, showing the feasibility and the applicability of our modeling approach.



(A) Single-Tier Applications



(B) Multi-Tier Applications

Fig.3 Validation Of Models For Saturated Situation

7. CONCLUSION AND FUTURE WORK

This paper describes the general architecture of transactional cloud applications and some common QoS metrics. Then, the performance modeling methods of such applications are studied in detail based on the queuing theory. On the basis of steady-state model, we enhance the model further to deal with the saturated scenario. Results of evaluation experiment illustrate the effectiveness and the feasibility of our analytic model. As possible future work, we intend to exploit deeper into the specific details into the application

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

performance modeling and also wider across other types of cloud applications.

ACKNOWLEDGMENTS

This research is supported in part by National Natural Science Foundation of China (No. 60963005) and Tsinghua – Tencent Joint Laboratory for Internet Innovation Technology (No. 2011-1).

REFRENCES:

[1] IBM. (2009). Seeding the Clouds: Key Infrastructure Elements for Cloud Computing. Available: ftp://ftp.software.ibm.com/common/ssi/sa/wh/n/oiw03022usen/OIW03022USEN.PDF

[2] J. M. Willis, "Who Coined the Phrase Cloud Computing?," IT Management and Cloud Blog, December, vol. 31, 2008.

[3] Wikipedia. Cloud Computing. Available: http://en.wikipedia.org/wiki/Cloud_computing

[4] X. Wang, Y. Xue, L. Fan, et al., "Research on Adaptive QoS-Aware Resource Reservation Management in Cloud Service Environments," in 2011 IEEE Asia-Pacific Services Computing Conference (APSCC), 2011, pp. 147-152.

[5] M. N. Bennani and D. A. Menasce, "Resource allocation for autonomic data centers using analytic performance models," in Proceedings of the 2nd IEEE International Conference on Autonomic Computing (ICAC'2005), Seattle, WA, 2005, pp. 229–240.

[6] A. Chandra, W. Gong, and P. Shenoy, "Dynamic Resource Allocation for Shared Data Centers Using Online Measurements," in Proceedings of ACM Sigmetrics 2003, San Diego, CA, 2003.

[7] D. A. Menascé and M. N. Bennani, "Autonomic Virtualized Environments," in Proceedings of IEEE International Conference on Autonomic and Autonomous Systems(ICAS'06), Silicon Valley, CA, USA, 2006, p. 28.

[8] D. A. Menascé, D. Barbará, and R. Dodge, "Preserving QoS of e-commerce sites through self-tuning: a performance model approach," in the 3rd ACM conference on Electronic Commerce, 2001, pp. 224-234.

[9] B. Urgaonkar, G. Pacifici, P. Shenoy, et al., "An analytical model for multi-tier internet services and its applications," in the 2005 ACM SIGMETRICS international conference on Measurement and modeling of computer systems, 2005, pp. 291-302.

[10] R. P. Doyle, J. Chase, O. Asad, et al., "Model-Based Resource Provisioning in a Web Service Utility," in The 4th USENIX Symposium on Internet Technologies and Systems, Seattle, Washington, USA, 2003.

[11] J. McKenna, "A Generalization of Little's Law to Moments of Queue Lengths and Waiting Times in Closed, Product-Form Queueing Networks," Journal of Applied Probability, vol. 26, pp. 121-133, 1989.

[12] P. J. Denning and J. P. Buzen, "The Operational Analysis of Queueing Network Models," ACM Computing Surveys (CSUR), vol. 10, pp. 225-261, 1978.

[13] F. Baskett, K. M. Chandy, R. R. Muntz, et al., "Open, Closed, and Mixed Networks of Queues with Different Classes of Customers," Journal of the ACM, vol. 22, pp. 248-260, 1975.