Percentile Performance Analysis of Infrastructure-as-a-Service clouds with task retrials

Li Zhu[1], Yuandou Wang[4], Wanbo Zheng[6], Lei Wu[3], Ye Yuan[5], Peng Chen[2], Yunni Xia[4]
1.State Grid Chongqing Jiangbei Power Supply Company, Chongqing, China
2. School of computers, Sichuan University, China
3.School of Mathematical Sciences, University of Electronic Science and Technology of China, China
4.School of computers, Chongqing University, China
5.Institute of Green and Intelligent Technology, Chinese Academy of Sciences
6. Chongqing Research Institute of China Coal Technology, China

Abstract—**Performance evaluation of cloud infrastructures and cloud-based applications is required to evaluate and quantify the cost-benefit of a strategy portfolio and the quality of service (QoS) experienced by end-users. For this purpose, we introduce an analytical framework to percentile-based performance analysis of unreliable Infrastructure-as-a-Service clouds with faulty and retrial tasks. The performance measured in a certain level percentile of the instantiation time predicted given variable load intensities, fault frequencies, multiplexing abilities, and instantiation processing delays. A case study based on a real-world campus cloud is carried out to show the correctness of the proposed theoretical model.**

I. Introduction

Cloud computing systems rely on sharing of resources to achieve coherence and economies of scale, similar to a utility over a network. With the help of the provision of on-demand access to computational resources, they deliver services at different levels: Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS), and Software-as-a-Service (SaaS). IaaS clouds supply users with resources in the form of virtual machine (VM) instances deployed in a provider data center, while PaaS and SaaS clouds offer services in terms of specific solution stacks and application software suites, respectively.

Performance guaranteed by cloud providers is one of the most important factors to user satisfaction. A performance guarantee is usually interpreted through Service-level-agreement (SLA) [1]-[3] between a user and its corresponding provider. For performance-oriented SLAs, a traditional way is to use mean/expected performance metrics, e.g., mean task response time and mean process completion time. This kind of guarantee could generally satisfy as many users as possible but sometimes lead to bad experiences of a part of users. For instance, a task response time distribution featuring 90% low durations and 10% ultra-high ones could manage to maintain a satisfactory averaged performance but the 10% part definitely causes bad user-perceived experience. Recent studies, instead, prefer percentile-based guarantees instead, which stipulates that no more than a given proportion of user-perceived delays should exceed a specified threshold.

A careful investigation into existing works suggests that we are in lack of comprehensive analytical performance models with efficient calculations of performance metrics. Existing solutions [4]-[12] are limited mainly because they usually assume failure-free VM instantiation to simplify the performance/QoS calculation. Instead of assuming reliable cloud provisioning, we present an analytical framework for percentile-based performance analysis of unreliable IaaS cloud with faulty virtual machine (VM) instantiation. Its performance is in particular expressed by a certain level percentile of the instantiation time and its corresponding SLA violation rate. The proposed model is capable of deriving performance results under variable load intensities, fault frequencies, multiplexing abilities, and VM instantiation processing distributions. To prove the correctness of the proposed calculation methods, we consider a real-world IaaS cloud as the test-bed and obtain experimental performance data. We show that the tails of theoretical performance results converge to the corresponding experimental ones.

II. System Model

An IaaS cloud provides users its highly scalable provision processing, storage, networks and other fundamental computing resources that can be adjusted on-demand. Users are therefore able to deploy and run arbitrary software, which can include operating systems and applications without managing or controlling the underlying infrastructure. An IaaS cloud is also responsible for system maintenance, backup and resiliency planning. This makes it well-suited for fluctuating workloads with unexpected changes. Its cloud management unit maintains a request buffer for incoming requests, which can be usually described by an arrival rate, $\lambda \in R^{+}$. The capacity of such buffer, denoted by $b \in N^{+}$, is specified before use. E.g., such capacity can be specified through the $FRAME\_SIZE$ property in OpenStack. Requests either leave by rate $\theta (\theta \leq 1)$ or are resubmitted by rate $1-\theta$ when the request buffer is fully occupied. The time needed for a resubmission is assumed to be r. The cloud management unit continuously processes requests from the request buffer on

978-1- 5090-4429- 0/17/$31.00 © 2017 IEEE

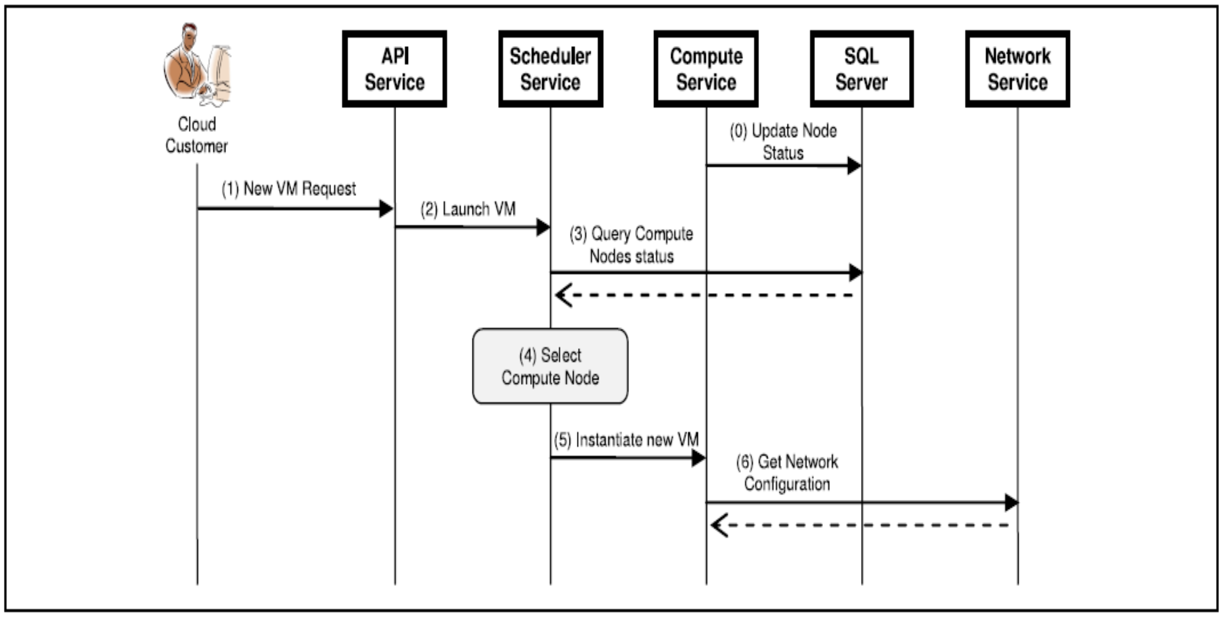

Fig. 1. Sequence chart of VM instantiation in OpenStack

an FIFO (first-in-first-output) basis and tries to instantiate the corresponding VMs on PMs on demand.

For the performance estimation purpose, we are concerned about instantiation response time, T, i.e., the interval time between request arrival and the corresponding VM being ready for execution, e.g., the time of $INSTANCE\_SPAWNED$ defined in OpenStack. As discussed earlier, most existing studies consider the mean value or hard deadline guarantee of the response time as the performance metric. A mean-value-based SLA could lead to bad user-perceived experience. On the other hand, a hard-deadline-based SLA is over pessimistic and usually guarantees the worst-case response time only. Instead, we consider percentile-based SLA and consider its corresponding SLA violation rate as the major performance metric.

As shown in Fig. 1, VM instantiation involves multiple steps and interactions with various services and components. T comprises of three parts: 1) the waiting time of a request in a request buffer; 2) the time required for the cloud management unit to spawn a VM, T_I. In OpenStack-based IaaS clouds, it is the interval time between $INSTANCE\_BUILDING$ time and $INSTANCE\_SPAWNED$ time; and 3) the time required for request retrial or resubmission due to the capacity limit or faulty instantiation , denoted by d.

With the help of a VM multiplexing [13] mechanism supported by today's multi-core/multi-threading technologies, multiple VMs can be instantiated on a same PM. The maximum number of VMs that can be instantiated on a PM, denoted by $m \in N^+$, is usually bounded. The maximum number of VMs instantiated is $n \times m$ where $n \in N^+$ denotes the number of PMs. VM multiplexing helps to achieve significant resource saving in comparison with individual-VM-based resource provisioning. Note that a high multiplexing level is not always welcomed because interferences among multiplexed tasks/threads/processes may cause performance and reliability degradation.

Moreover, Fig. 1 shows the possibility of unsuccessful VM instantiation because interactions with local or remote services and components can be fault-prone. They are usually caused by, e.g., temporary connection losses to remote database, write message congestions, invalid input/output sequences, gateway failures, SQL activity failures, and unexpected user exits. They can strongly impact cloud performance due to the overhead needed to conduct compensation/transactional rollback activities and re-instantiate faulty VMs. This is especially true at high request load.

III. Stochastic Analysis

We denote ρ as the equivalent cloud utilization:

$$\rho = \frac{\lambda' \times \mathbb{E}(T_I)}{n \times m} \tag{1}$$

where T_I denotes the instantiation processing time, $\mathbb{E}(T_I)$ is the expectation of T_I, and λ' the effective request arrival rate. Note that λ' is different from λ since resubmitted requests join the original arrival flow.

It has been widely recognized that heavy-tailed distributions are well suited for modeling job processing and request handling activities in computer systems and networks. Thus, we employ it for the VM instantiation processing time. In particular, we choose the Pareto distribution where T_I is lower-bounded by $\widehat{x}$. The tail of T_I can therefore be obtained as:

$$\mathbb{P}\{T_I \geq \widehat{x}\} = [\frac{x}{\widehat{x}}]^{-v} \tag{2}$$

where $x \geq \widehat{x}$ and $v \in \{1, 2\}$. The above equation suggests that it is more likely to observe a longer instantiation processing time than the light-tailed one and its expectation is:

$$\mathbb{E}(T_I) = \frac{v \times \widehat{x}}{v - 1} \tag{3}$$

Let R denote the rejection rate due to the capacity limit, we have:

$$R = \frac{1}{(nm)^b \times (nm)!} (\lambda' \times \mathbb{E}(T_I))^{nm+b} P_0 \tag{4}$$

where P_0 denotes the probability that the waiting buffer is empty:

$$P_0 = [\sum_{i=0}^{nm-1} \frac{(\mathbb{E}(T_I) \times \lambda')^i}{i!} + \sum_{i=nm}^{nm+b} \frac{(\mathbb{E}(T_I) \times \lambda')^i}{(nm)^{i-nm}(nm)!}]^{-1} \tag{5}$$

Let λ'' denote the resubmission rate due to the capacity limit, we have:

$$\frac{\lambda''}{\lambda'} = \frac{R \times (1 - \theta)}{1 - R} \tag{6}$$

λ'' can therefore be calculated as:

$$\lambda'' = \lambda' \times \frac{R \times (1 - \theta)}{1 - R} \tag{7}$$

By combining the above equations, we have:

$$\begin{aligned} \lambda' &= (\lambda' \times f + \lambda'' + \lambda) \times (1 - R) \\ &= (\lambda' \times f + \lambda' \times \frac{R \times (1 - \theta)}{1 - R} + \lambda) \times (1 - R) \end{aligned} \tag{8}$$

Note that R is determined by λ'', b, m, n, and $\mathbb{E}(T_I)$ according to (4) and (5). Thus, the above equation suggests that λ' can be calculated if λ, b, m, and n are given and the distribution of T_I is known. However, the product form expression of λ' does not exist . Fortunately, we can obtain its numerical solution using tools, e.g., the **solve()** tool in **Matlab**.

A. *Response time estimation*

The instantiation time, T, is defined as the interval time between request arrival and the corresponding VM being successfully spawned and ready for execution. To analyze T, we first have to analyze the distribution of the time that a request resides in the IaaS cloud on condition that it is not rejected and resubmitted, T'. T' also stands for the sojourn time of a request on condition that it enters the request buffer once and only once.

Since the instantiation processing time, T_I, is heavy tailed, the tail of T' can be asymptotically approximated as:

$$\mathbb{P}\{T' \geq \widehat{t}\} \sim \mathbb{P}\{T_I \geq (1 - \rho)\widehat{t}\} \tag{9}$$

In the following, we have to calculate the probability that a cloud request enters the retrial control unit, P_r:

$$P_r = (1 - \theta)R + f(1 - R) \tag{10}$$

As can been seen, a successfully instantiated request may experience several retrials before its final successful instantiation due to the capacity limit and instantiation faults. The expected number of retrials of a cloud request on condition that it is finally successfully instantiated, N_r, can therefore be obtained as:

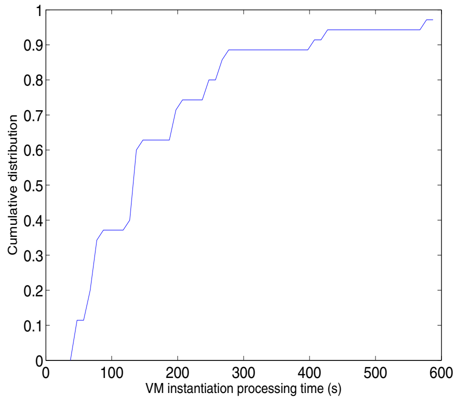

Fig. 2. The cumulative distribution of VM instantiation processing times

$$N_r = \frac{1}{1 - P_r} - 1 \tag{11}$$

N_r has two part, namely the expected number of retrials due to the capacity limit, N_{rc}, and the expected number of retrials due to faulty instantiations, N_{rf}:

$$N_{rc} = N_r \times \frac{f(1 - R)}{P_r} \tag{12}$$

$$N_{rf} = N_r \times \frac{(1 - \theta)R}{P_r} \tag{13}$$

instantiation time can therefore asymptotically approach the sum of the time needed for retrials due to the capacity limit, the time needed for retrials due to faulty instantiations, and the time needed for the last successful trial:

$$\mathbb{P}\{T \geq t\} \sim \mathbb{P}\{T' \times N_{rf} + d \times N_{rc} + T' \geq t\} \tag{14}$$

IV. Case study and model validation

For the model validation purpose, we conduct a case study on a real-world IaaS cloud, the **Course Management and Assignment Submission cloud** for undergraduate students of ChongQing University (CQU). It implements the IaaS architecture and is built by the XenServer and OpenStack toolkits. The cloud system is based on a symmetric server group of 6 Sugon I450 servers (4-CPU Intel Xeon 5506/128G RAM/15TB RAID but only 3-CPU/8G RAM/4TB RAID is assigned as cloud users' space), each of which serves as a PM. Each PM can concurrently support no more than 8 VMs. The capacity of the waiting buffer for requests is 16. The fault rate is 0.13-0.79%. The request arrival rate and instantiation processing time vary test by test. The occurrence rate of impatient wait is 11.3% when the request buffer is fully occupied.

The cloud continuously processes requests from students and tries to instantiate each request into a corresponding VM instance. Each VM is deployed on only one PM for data consistency and integrity. The test logfile covers time stamps

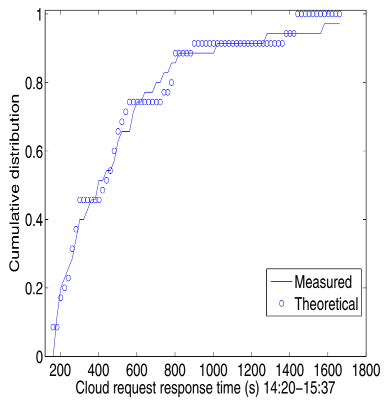

Fig. 3. Test 1 at 14:20-16:07

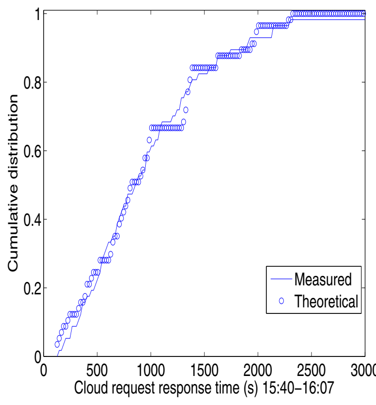

Fig. 4. Test 2 at 14:20-16:07

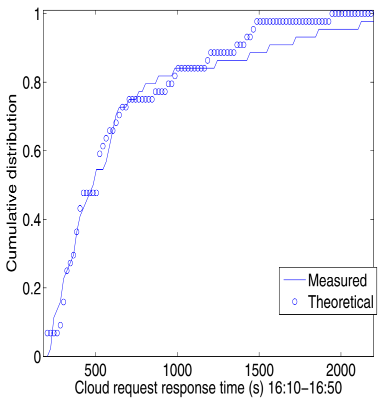

Fig. 5. Test 3 at 16:10-17:30

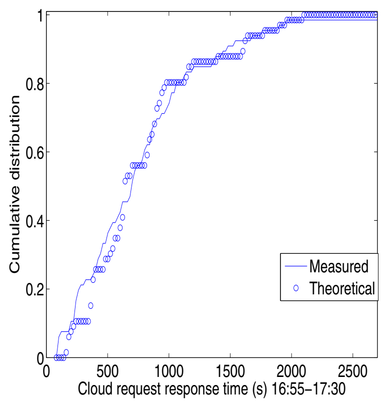

Fig. 6. Test 4 at 14:20-16:07

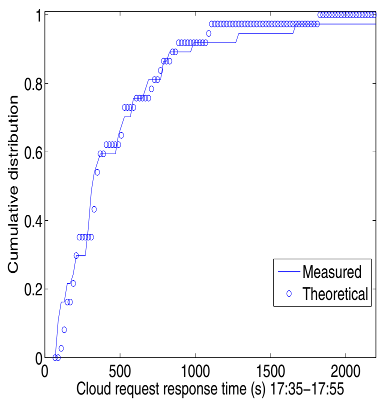

Fig. 7. Test 5 at 14:20-16:07

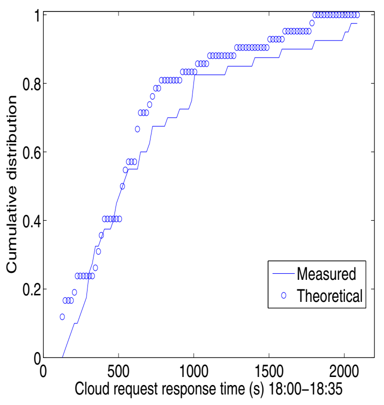

Fig. 8. Test 6 at 16:10-17:30

of each request's arrival, departure, failure, and successful instantiation times in consecutive periods from 14:20 to 18:35, Feb. 26, 2016. As can be seen from the figures comparing theoretical results and corresponding measured ones, theoretical tailed distributions well converge to those of measured empirical ones derived from the test logfile. The correctness and accuracy of our proposed methods are therefore validated. From the theoretical tailed distributions, we can easily calculate the SLA violation rate given the response time threshold. For instance, if we look at the theoretical tailed distribution at 14:20-15:37 and tolerate a maximum response time of $600s$. The theoretical violation rate can be calculated as 0.23 because the corresponding cumulative distribution suggests $\mathbb{P}\{T \leq 600\} = 0.77$. If we tolerate a maximum response time of $400s$, the violation rate is increased to 0.55 because a higher performance threshold is less likely to be met.

Note that we have considered other test data sets of industrial clouds published online in the beginning but realized that very few are available and they are insufficient to fit our model setup. Other related studies use simulative performance data based on the **CloudSim** simulation tool. Using such data seems less convincing than using real measured one.

We also study the impact of varying parameters on cloud performance. We first investigate how variations in request input rate affect performance. Fig. 8(a) illustrates performance changes with variations in request arrival rates when $n = 2, m = 1, b = 4, \theta = 0.05, f = 0.06, d = 17, \widehat{R} = 0.0024, \widehat{y} = 300s, \widehat{G} = 0.2$, and the distribution of T_I is given in Fig. 2. Increasing arrival rate leads to higher violation rate and rejection rate as expected. Fig. 8(b) illustrates performance changes with variations in fault rates of VM instances when $\lambda = 0.05, n = 2, m = 1, b = 4, \theta = 0.05, d = 17, \widehat{R} = 0.0024, \widehat{y} = 400s$, and $\widehat{G} = 0.2$. Increasing it results in higher violation and rejection rates. Fig. 8(c) illustrates performance changes with variations in request buffer size when $\lambda = 0.05, n = 2, m = 1, \theta = 0.05, d = 17, \widehat{R} = 0.0024, \widehat{y} = 180s$, and $\widehat{G} = 0.16$. Increasing buffer size lowers the violation and rejection rates.

V. Conclusions and Further Studies

In this work, we introduce a comprehensive percentile-based performance determination model for fault-prone IaaS clouds with request rejection and resubmission. We measure the SLA violation rate with a given threshold of instantiation time and request rejection rate under variable system conditions (fault intensity, VM instantiation time distribution, multiplexing ability, and request load). For the model validation purpose, we conduct a case study based on a real-world campus cloud

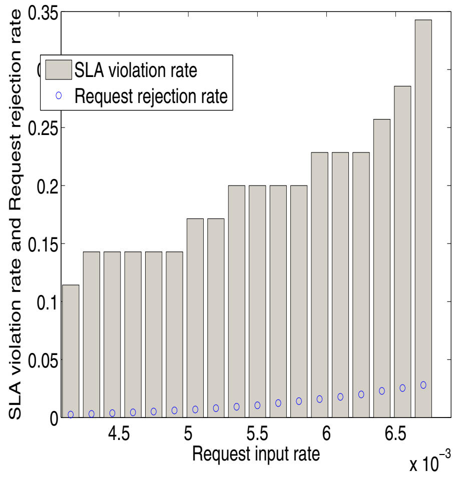

Fig. 9. Performance trends with variable arrival rates

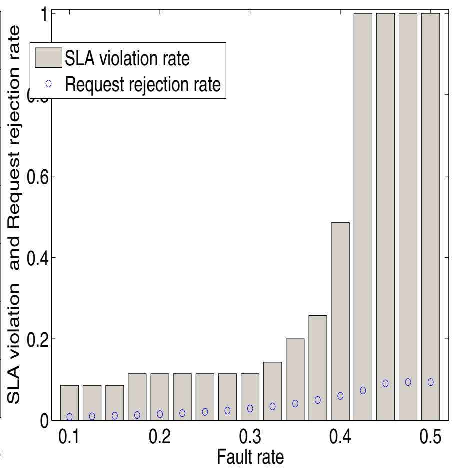

Fig. 10. Performance trends with variable fault rates

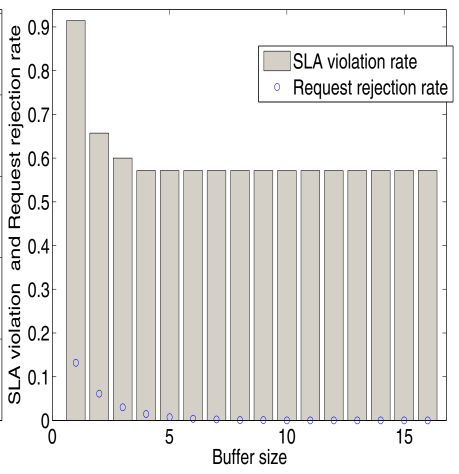

Fig. 11. Performance trends with variable buffer sizes

and show that theoretical distribution of instantiation time well converges to its corresponding empirical ones derived based on measured performance data.

VI. Acknowledgements

This work is supported by NSFC under Grant No. 61472051; Fundamental Research Funds for the Central Universities under project Nos. 106112014CDJZR185503 and CDJZR12180012; Science foundation of Chongqing Nos. cstc2014jcyjA40010 and cstc2014jcyjA90027; Chongqing Social Undertakings and Livelihood Security Science and Technology Innovation Project Special Program No. cstc2016shmszx90002; China Postdoctoral Science Foundation No. 2015M570770; Chongqing Postdoctoral Science special Foundation No. Xm2015078; Universities Sci-tech Achievements Transformation Project of Chongqing No. KJZH17104.

Yunni Xia (email: xiayunni@hotmail.com) is the correspondence author of this work.

References

[1] H.L. Zhang, P.P. Li, Z.G. Zhou, ”Performance Difference Prediction in Cloud Services for SLA-Based Auditing, ” in *Proc. Service-Oriented System Engineering (SOSE) IEEE Symposium on*, pp. 253-258, May 2015.

[2] D. Mazmanov, C. Curescu, H. Olsson, A. Ton, J. Kempf, ”Handling Performance Sensitive Native Cloud Applications with Distributed Cloud Computing and SLA Management,” in *Proc. Utility and Cloud Computing (UCC), IEEE/ACM 6th International Conference on*, pp. 470-475, July 2013.

[3] R. Garg, H. Saran, R. S. Randhawa, M. Singh, ”A SLA framework for QoS provisioning and dynamic capacity allocation, ”in *Proc. Quality of Service, IEEE International Workshop on*, pp. 129-137, July 2002.

[4] K. Xiong, H. Perros, ”Service performance and analysis in cloud computing,” in *Proc. World Conf. on Services-I*, pp. 693-700, July 2009..

[5] Y. S. Dai, B. Yang, J. Dongarra, G. Zhang, ”Cloud service reliability: Modeling and analysis,” in *Proc. IEEE Pacific Rim Int. Symp. On Dependa. Comput.*, pp. 784-789, November 2009.

[6] Y. N. Xia, M. C. Zhou, X. Luo, S. C. Pang, Q. S. Zhu, J. Li, ”Stochastic Modeling and Performance Analysis of Migration-enabled and Error-prone Clouds,” in *IEEE Trans. Indus. Info.*, vol. 11, no. 2, pp. 495-504, May 2015.

[7] B. Yang, F. Tan, Y. S. Dai, S. C. Guo, ”Performance evaluation of cloud service considering fault recovery,” in *Proc. Int. Conf. on Cloud Comput.*, pp. 571-576, December 2009.

[8] S. He, L. Guo, M. Ghanem, Y. Guo, ”Improving Resource Utilisation in the Cloud Environment using Multivariate Probabilistic Models,” in *Proc. Int. Conf. on Cloud Comput.*, pp. 574-581, June 2012.

[9] D. Bruneo, ”A Stochastic Model to Investigate Data Center Performance and QoS in IaaS Cloud Computing Systems,” *IEEE Trans. Par. Distr. Sys.*, vol. 25, no. 3, pp. 560-569, March 2014.

[10] D. Bruneo, S. Distefano, F. Longo, A. Puliafito, and M. Scarpa, ”Workload-Based Software Rejuvenation in Cloud Systems,” *IEEE Trans. Comput.*, vol. 62, no. 6, pp. 1072-1085, JUNE 2013.

[11] R. Ghosh, K. S. Trivedi, V. K. Naiky, and D. S. Kim, ”End-to-End Performability Analysis for Infrastructure-as-a-Service Cloud: An Interacting Stochastic Models Approach,” in *Proc. Pacific Rim Int. Symp. on Dependa. Comput.*, pp. 125-132, Dec. 2010.

[12] H. Khazaei, J. Misic, and V. B. Misic, ”A Fine-Grained Performance Model of Cloud Computing Centers,” *IEEE Trans. Par. Distr. Sys.*, vol. 24, no. 11, pp. 2138-2147, Nov. 2013.

[13] X. Meng, ”Efficient Resource Provisioning in Compute Clouds via VM Multiplexing,” in *Proc. Int. Conf. Aut. Comput. (ICAC2010)*, pp. 11-20, June 2010.