On Stochastic Performance and Cost-aware Optimal Capacity Planning of Unreliable Infrastructure-as-a-Service Cloud[⋆]

Weiling Li[1], Lei Wu[2], Yunni Xia[1⋆⋆], Yuandou Wang[1], Kunyin Guo[1], Xin Luo[1], Mingwei Lin[4], and Wanbo Zheng[3]

1. College of Computer Science, Chongqing University, Chongqing, China
2. College of Mathematical Sciences, University of Electronic Science and Technology of China, Chengdu, China
3. College of Optoelectronic Engineering, Chongqing University, Chongqing, China
4. Faculty of Software, Fujian Normal University, Fuzhou, China

Abstract. Performance evaluation of cloud data-centers has drawn considerable attention from academy and industry. In this study, we present an analytical approach to the performance analysis of Infrastructure-as-a-Service cloud data-centers with unreliable task executions and resubmissions of unsuccessful tasks. Several performance metrics are considered and analyzed under variable load intensities, failure frequencies, multiplexing abilities, and service intensities. We also conduct a case study based on a real-world cloud data-center and employ a confidence interval check to validate the correctness of the proposed model. For the performance optimization and optimal capacity planning purposes, we are also interested in knowing the minimized expected response time subject to the constraint of request rejection rate, hardware cost in terms of the cost of physical machines and the request buffer. We show that the optimization problem can be numerically solved through a simulated-annealing-based algorithm.

Key words: IaaS cloud, performance, optimal capacity planning, Simulated annealing

1 Introduction

Cloud data-centers are key enablers for the scalability of the cloud platform.Cloud computing relies on data-centers to deliver expected services. The widespread adoption of the cloud computing paradigm mandates the exponential growth in

⋆ This work is in part supported by NSFC under Grant Nos. 61472051, 61170183, 61272093, 61572523, and 61202347; Young Scientist Foundation of Chongqing No. cstc2013kjrc-qnrc0079; Fundamental Research Funds for the Central Universities under project Nos. 106112014CDJZR185503 and CDJZR12180012; Science foundation of Chongqing No. cstc2014jcyjA40010; China Postdoctoral Science Foundation No. 2015M570770; Chongqing Postdoctoral Science special Foundation no. Xm2015078.

⋆⋆ Corresponding Author. Email: xiayunni@hotmail.com

the data-centers' computational, network, and storage resources. Managing the computational resources to deliver specified performance [1] is among the key challenges.

Expected request response time which decides system responsiveness and request rejection rate which determines users' satisfaction are usually considered as the most important performance metrics to evaluate a service system. As will be discussed later in this paper, cloud data-centers are usually subject to errors/faults, and error/failure-handling activities could have strong impact on final performance.Due to the difficulties with building monolithic models capable of capturing related factors, measurement-based approaches are frequently used [2]-[4]. However, these approaches are intractable due to exhaustive experimentations.Therefore, their value is limited. Comprehensive analytical performance models are more preferable in this situation. Although some other analytical models [5]-[8] are proposed, those works are limited mainly because they assume the system failure-free to simplify the performance/QoS calculation.

This paper focuses on analytical performance analysis of IaaS cloud data-centers with request rejection and resubmission. For this purpose, a stochastic model is proposed and product-form expressions of multiple performance metrics are derived.We validate the model by experiment based on a actual IaaS cloud and the results indicate our model is trustable.For the optimal capacity planning and cost saving purposes, which are conflicting with each other, we are also interested in knowing the best system responsiveness that can be achieved.We employ a simulated-annealing-based algorithm to solve this problem.

2 System Model

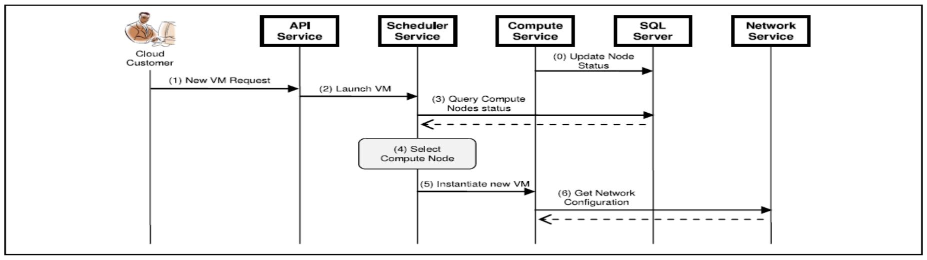


Fig. 1: Sequence chart of VM instantiation on OpenStack

IaaS cloud is a form of cloud computing that provides virtualized computing resources over the Internet. The cloud management unit of an IaaS data-center maintains a request buffer for consecutively-arrived requests, which can be usually described by an arrival rate, λ. The capacity of such buffer, denoted by c, can be specified before using (e.g., the capacity limit can be specified through the

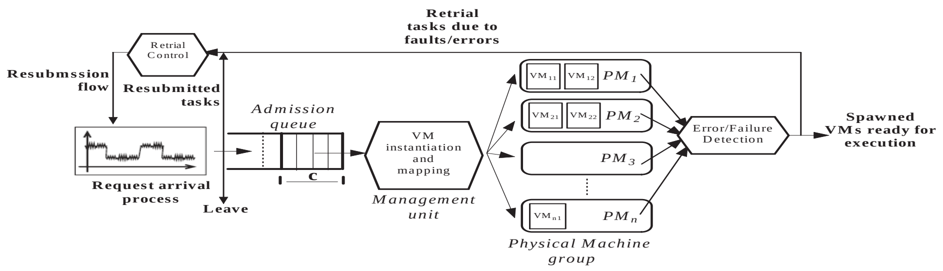


Fig. 2: The cloud provisioning control-flow of an IaaS data-center with task errors/faults

$FRAME\_SIZE$ property in OpenStack). Requests arrived either leave by rate θ or are resubmitted by rate $1-\theta$ when the capacity limit is reached. For the performance evaluation purpose, we are interested in knowing request response time, i.e., the expected interval time between request arrival and the corresponding VM ready for execution (e.g., the time of $INSTANCE\_SPAWNED$ defined in OpenStack).

As shown by Fig.1, VM instantiation requires multiple steps and interferences with various services and components.Averaged speed (or process rate) of the cloud management unit to spawn a VM, denoted by μ, can be obtained by the reciprocal of averaged instantiation times, e.g., intervals between $INSTANCE\_BUILDING$ times and $INSTANCE\_SPAWNED$ times in OpenStack. With the help of VM multiplexing [9] 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, is usually bounded. Note that high multiplexing level is not always welcomed because VM interference may cause performance and reliability degradation.

Moreover, Fig.1 suggests potential of unsuccessful VM instantiation because interactions with local or remote services and components are often error/failure-prone.

Errors/failues can strongly impact cloud performance due to the overhead needed to conduct compensation/transactional-rollback activities and re-instantiate the faulty request. Based on the above discussions, an abstract control-flow model of VM instantiation on unreliable IaaS cloud data-center is illustrated in Fig. 2. It abstracts away implementation details of IaaS cloud paradigm while preserving the control-flow contents useful for performance analysis in a context of queueing-networks. Its objective is to derive the quantitative effects of varying request arrival rates, VM instantiation rates, resource scale, and error intensity on cloud performance. The system under study is consequently mapped into an instance of queuing network problems solved in the following section.

3 Stochastic Analysis

Let $N(t) = n$ mean that the number of tasks waiting or being instantiated is n at time t, $M(t) = m$ mean that the number of requests being resubmitted m, and $X(t) = (N(t), M(t))$ denote the system state at time t, the resulting state space is therefore $E \in \{0, 1, ..., k\} \times \{0, 1, ..., \infty\}$. Since the inter-arrival time, VM instantiation time, and resubmission processing time are all exponentially distributed, $X(t)$ is a Markovian process on state space E.

Based on the state transition chart, the corresponding transition-rate matrix, Q, can be derived as:

$$Q = \begin{bmatrix} A_0 & C & & & & \\ B_1 & A_1 & C & & & \\ & B_2 & A_2 & C & & \\ & & \ddots & \ddots & \ddots & \\ & & & B_{k-1} & A_{k-1} & \widetilde{C} \\ & & & & \widetilde{B_k} & \widetilde{A_k} \end{bmatrix} \tag{1}$$

It is easy to see that $X(t)$ is irreducible and non-periodical. Let $\pi_{k,j}(t)$ denote the probability that the Markovian process is at state (k, j) and $\pi_{k,j} = \lim_{t\to\infty} \pi_{k,j}(t)$, we have that $\pi_{k,j}$ can be calculated as below if the stationary distribution exists:

$$\pi_{k,j} = \pi_{k,j-1} \prod_{l=g+1}^{j} \rho_l \tag{2}$$

where

$$\rho_l = \frac{e \times f \times \mu + (1-\theta)\lambda}{l \times \theta \times \mu' + e \times (1-f)\mu} \tag{3}$$

It is easy to see that ρ_l decreases with l. Consequently, there exists $u \in N^+$ such that $\rho_u < 1$ and

$$\begin{aligned} &\sum_{j=g+1}^{\infty} \{ \prod_{l=g+1}^{j} (\frac{e \times f \times \mu + (1-\theta)\lambda}{l \times \theta \times \mu' + e \times (1-f)\mu}) \} \\ &= \sum_{j=g+1}^{\infty} \{ \prod_{l=g+1}^{j} \rho_l \} < \sum_{j=g+1}^{u} (\rho_{g+1})^{j-g} + \sum_{j=u+1}^{\infty} (\prod_{l=g+1}^{j} \rho_l) \\ &< \sum_{j=g+1}^{u} (\rho_{g+1})^{j-g} + (\rho_{g+1})^{u-g} \sum_{j=u+1}^{\infty} (\prod_{l=u+1}^{j} \rho_l) \\ &< \sum_{j=g+1}^{u} (\rho_{g+1})^{j-g} + (\rho_{g+1})^{u-g} \sum_{j=u+1}^{\infty} (\rho_l)^{j-u} \\ &= \sum_{j=g+1}^{u} (\rho_{g+1})^{j-g} + (\rho_{g+1})^{u-g} \frac{\rho_u}{1-\rho_u} < \infty \end{aligned} \tag{4}$$

The above derivation leads to

$$\sum_{j=g+1}^{\infty}\{\prod_{l=g+1}^{j}(\frac{e\times f\times\mu+(1-\theta)\lambda}{l\times\theta\times\mu'+e\times(1-f)\mu})\}<\infty \tag{5}$$

and therefore the stationary distribution exists according to the limit theorems of birth-death processes.

Since the stationary distribution exists, we have the steady-state probabilities of each state as:

$$\pi Q=0,\sum_{i=0}^{k-1}\sum_{j=0}^{g}\pi_{i,j}+\sum_{j=0}^{\infty}\pi_{k,j}=1 \tag{6}$$

Since the stationary distribution exists, we also have:

$$\begin{aligned}&\pi_{k-1,g}\lambda+(1-\theta)\pi_{k,g-1}\lambda+e(1-f)\mu)\\&=\pi_{k,g}(e\times\mu+\lambda(1-\theta)+\theta\times g\times\mu')\\&\qquad-\pi_{k,g+1}(g+1)\times\theta\times\mu'\end{aligned} \tag{7}$$

From (2) we have:

$$\pi_{k,g+1}=\pi_{k,g}\frac{\lambda(1-\theta)+e\times f\mu}{(g+1)\theta\times\mu'+e\times(1-f)\mu} \tag{8}$$

Combining the above equation with (7), we have:

$$\pi_{k,g-1}\lambda(1-\theta)=\pi_{k,g}(e\times(1-f)\mu+g\times\theta\times\mu') \tag{9}$$

which suggests that $A_k=\widetilde{A_k}$ and $B_k=\widetilde{B_k}$.

According to (6) and (9), we have:

$$\begin{aligned}&T_0\frac{1}{W}A_0+T_1\frac{1}{W}B_1=0\\&T_i\frac{1}{W}C+T_{i+1}\frac{1}{W}A_{i+1}+T_{i+2}\frac{1}{W}A_{i+2}=0,0\le i\le k-2\\&T_{k-1}\frac{1}{W}C+T_k\frac{1}{W}A_k=0\end{aligned} \tag{10}$$

where T_0 is a basic solution of $T_0(V_kA_k+V_{k-1}C)=0$ and $\pi_{i,j}$ is subject to:

$$\begin{aligned}&(\pi_{0,0},...,\pi_{0,g})=T_0\frac{1}{W}\\&(\pi_{i,0},...,\pi_{i,g})=(\pi_{0,0},...,\pi_{0,g})V_i,0<i\le k\\&\pi_{k,j}=T_0\frac{1}{W}V_k\omega_1\prod_{l=g+1}^{j}\frac{e\times f\times\mu+(1-\theta)\lambda}{l\times\theta\times\mu'+e\times(1-f)\mu}\end{aligned} \tag{11}$$

where ω_1 is a column vector with its dimension being $g+1$ and is equal to $(0,..,0,1)^T$.

and, V_k is subject to:

$$\begin{aligned}
&V_0 = I \\
&V_1 = -A_0(B_1)^{-1} \\
&V_2 = -(V_0C + V_1A_1)(B_2)^{-1} \\
&V_i = -(V_{i-2}C + V_{i-1}A_{i-1})(B_i)^{-1}, 2 < i \leq k
\end{aligned} \tag{12}$$

W is calculated as:

$$W = T_0(\sum_{i=0}^{k} V_i)\omega + T_0V_k\omega_1 \sum_{j=g+1}^{\infty} (\prod_{l=g+1}^{j} \frac{e \times f \times \mu + (1-\theta)\lambda}{l \times \theta \times \mu' + e \times (1-f)\mu})) \tag{13}$$

From (10), we have:

$$T_1\frac{1}{W} = -T_0\frac{1}{W}A_0(B_1)^{-1} = T_0\frac{1}{W}V_1 \tag{14}$$

and similarly:

$$T_2\frac{1}{W} = -T_0\frac{1}{W}(V_1A_1 + C)(B_2)^{-1} = T_0\frac{1}{W}V_2 \tag{15}$$

Consequently, we can finally have:

$$T_i = -T_0V_i \tag{16}$$

and the product form solution of T_i can be obtained by using this equation.

Combining (16) with (2), we can obtain the solutions of steady-state probabilities of all states, $\pi_{i,j}$. Note that a similar derivation skill can be found in [10].

4 Performance Results

We consider the following as the performance metrics: 1) Expected request response time, T; and 2) Request rejection rate, R.

As suggested by Fig. 2, T denotes the expected interval between request arrival and the moment of the corresponding VM being instantiated and ready for execution. Response time is a frequently used measure of efficiency and responsiveness of computer systems.Lower response time also allows for a higher system reliability since in an unreliable system failures/errors are more likely to happen when a longer response time is needed.

To analyze T, we first have to calculate the probability that a cloud task enters the resubmission state, P_r:

$$P_r = (1-\theta)(\sum_{j=0}^{\infty} \pi_{k,j}) + f(1 - \sum_{j=0}^{\infty} \pi_{k,j}) \tag{17}$$

The expected number of retrials of a cloud task, N_r, can be obtained as:

$$N_r = \frac{1}{1 - P_r} - 1 \tag{18}$$

The expected time for a task to wait before it is resubmitted to the arrival task flow, T_r, can therefore be calculated as:

$$T_r = \frac{\lambda'/\mu' + \frac{P_0(\lambda'/(g\times\mu'))(\lambda'/\mu')^g}{g!(1-\lambda'/(g\times\mu'))}}{\lambda'} \tag{19}$$

where P_0 denotes the probability that no task being resubmitted:

$$P_0 = \frac{1}{\sum_{l=0}^{g-1}\frac{(\lambda'/\mu')^l}{l!} + \frac{(\lambda'/\mu')^g}{g!}}\left(\frac{1}{1 - \lambda'/g\times\mu'}\right) \tag{20}$$

and λ' is the rate of resubmission flow into the arrival task flow:

$$\lambda' = (\lambda + \lambda')[(\sum_{j=0}^{\infty}\pi_{k,j})(1-\theta) + (1 - \sum_{j=0}^{\infty}\pi_{k,j})f] \tag{21}$$

where λ' can be calculated as:

$$\lambda' = \lambda\frac{\sum_{j=0}^{\infty}\pi_{k,j}(1-\theta) + f - \sum_{j=0}^{\infty}\pi_{k,j}\times f}{1 - \sum_{j=0}^{\infty}\pi_{k,j}(1-\theta) - f + \sum_{j=0}^{\infty}\pi_{k,j}\times f} \tag{22}$$

The expected total time for a task to spend before its final successful trial on condition that it is not rejected, T_b, can therefore be obtained as:

$$T_b = N_r(T_r\sum_{j=0}^{\infty}\pi_{k,j} + (T_r + T_v)(1 - \sum_{j=0}^{\infty}\pi_{k,j})) \tag{23}$$

where T_v is calculated as:

$$T_v = \frac{1}{\mu} + \sum_{j=0}^{g}\pi_{0,j}\frac{\rho(\rho\times e)^e}{\lambda''(1-\rho)^2\times(e)!} \tag{24}$$

$\rho = \frac{\lambda''}{e\times\mu}$ and

$$\lambda'' = (\lambda + \lambda')(1 - \sum_{j=0}^{\infty}\pi_{k,j}) \tag{25}$$

Finally, we have T as:

$$T = T_b + T_v \tag{26}$$

Request rejection rate, R, can be expressed as the ratio of the number of rejected tasks, due to the capacity constraint or PM failures/errors, to the total number of requests submitted to the IaaS data-center. For user satisfaction, a low rejection rate is always preferable.

$$R = \theta\sum_{j=0}^{\infty}\pi_{k,j} \tag{27}$$

5 Case Study and Model Validation

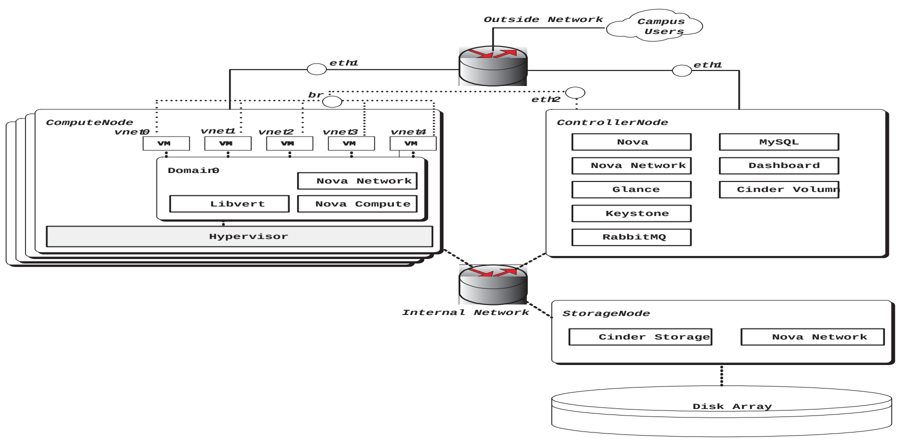


Fig. 3: The architectural view of the Course-Management and Assignment-Submission cloud

For the model validation purpose, we conduct a case study on a real-world cloud data-center, the **Course-Management and Assignment-Submission cloud for undergraduate students of ChongQing University (CQU). Its protocol and architectural views are illustrated in Fig.3.**

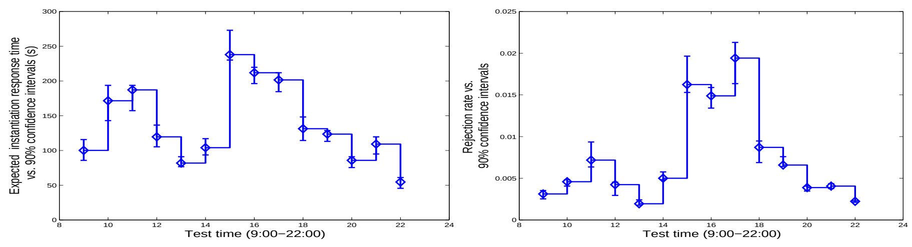
The cloud system is based on a symmetric server group of 6 Sugon I450 servers (4-CPU Intel Xeon 5506/128G RAM/15TB RAID but only 3-CPU/8G RAM/4TB RAID is assigned as cloud users' space). Each PM can therefore concurrently support no more than 32 VMs. The capacity of the waiting buffer for requests c is 16. The faulty rate f is 0.13-0.79%. The occurrence rate of impatient wait is 11.3% when the waiting buffer is fully occupied, meaning that $\theta = 0.113$

As shown in Table 1, the logfile covers time-stamps of each request's arrival and departure time in consecutive periods of 60 minutes from 09:00 to 22:00, Feb. 26, 2016 .

For the model validation purpose, we derive 90% confidence intervals from the experimental performance data . By using a normal distribution as the fitting function, we derive the confidence interval of T as:

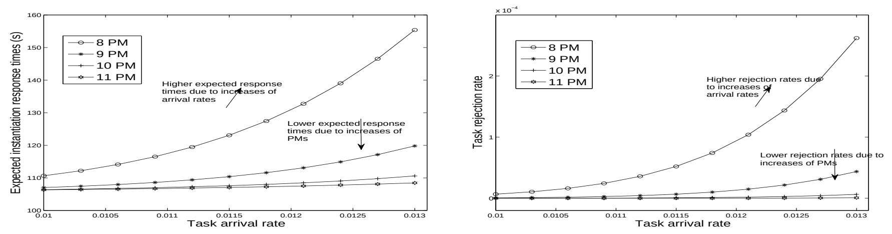
$$intv(T) = [\bar{ct} - z_{1-a/2}\frac{sdv}{\sqrt{\hat{s}}}, \bar{ct} + z_{1-a/2}\frac{sdv}{\sqrt{\hat{s}}}] \tag{28}$$

where $\bar{ct}$ stands for the mean of experimental request response time, sdv its standard deviation, $\hat{s}$ the sample size, z the z-distribution, and α the confidence level.



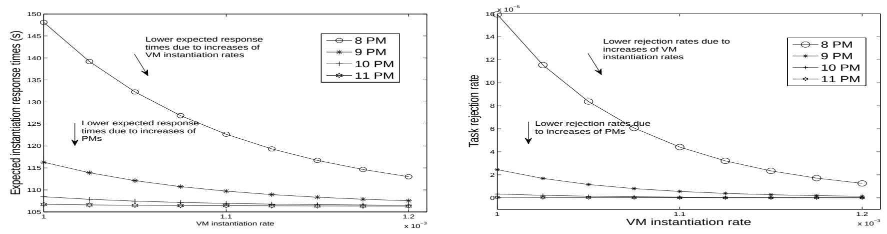
(a) Response time (b) Rejection rate

Fig. 4: Validation through confidence interval check



(a) Response time (b) Rejection rate

Fig. 5: Analytical performance results vs. arrival rate at different number of PMs



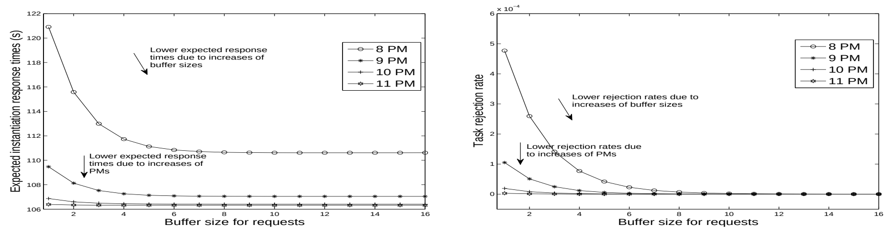
(a) Response time (b) Rejection rate

Fig. 6: Analytical performance results vs. VM instantiation rate at different number of PMs

Finally, the confidence interval of R is also based on a Bernoulli distribution as the fitting function:

$$intv(R) = [\bar{r} - z_{1-a/2}\sqrt{\frac{\bar{r}-\bar{r}^2}{\hat{s}}}, \bar{r} + z_{1-a/2}\sqrt{\frac{\bar{r}-\bar{r}^2}{\hat{s}}}] \tag{29}$$

where $\bar{r}$ stands for the experimental rejection rate.



(a) Response time (b) Rejection rate

Fig. 7: Analytical performance results vs. request buffer size at different number of PMs

Table 1 and Fig. 5 imply the correctness of the proposed theoretical model.

Fig. 6 illustrates performance changes with variations in request arrival rates when $m = 2, c = 8, \mu = 0.00125, \mu' = 0.01, g = 4, f = 0.08, \theta = 0.1$. Increasing arrival rate leads to higher expected response time and rejection rate. IaaS cloud maintains a small number of PMs. It can be seen that clouds with more PMs are more resistant to performance loss when arrival rate increases.

Fig. 7(a) illustrates performance changes with variations in VM instantiation rates when $m = 2, c = 8, \mu' = 0.01, g = 4, f = 0.08, \theta = 0.1, \lambda = 0.01$. Increasing VM instantiation rate leads to lower expected response time and rejection rate. It can also be seen that clouds with fewer PMs are more sensitive to performance improvements when VM instantiation rate increases.

Fig. 7(b) illustrates performance changes with variations in buffer sizes. The growth of such size leads to lower expected response time and rejection rate. Such performance improvements are strong when the size is small. It is also seen that clouds with more PMs always have higher performance.

6 Cost-aware Optimal Capacity Planning

As shown in previous sections, the proposed model is capable of giving validated performance results when input parameters are known. We are also interested in deciding optimal system capacity with highest performance and under cost constraints, which are conflicting with each other.

This problem can be formulated as:

$$\begin{aligned} Min \quad & T(n, c, m, g) \\ s.t. \quad & R(n, c, m, g) < RJ \\ & n \times cpm < CPM \\ & cbf(c) < CBUF \\ & cm(m) \times n < CM \\ & GLO < g < GUP \end{aligned} \tag{30}$$

Table 1: Theoretical results and confidence intervals

Test time	T	CI	$R(10^{-3})$	CI(10^{-3})
09:00	100.1s	[85.8-115.7s]	3.12	[2.52-3.53]
10:00	171.6s	[143.0-193.7s]	4.59	[4.06-4.82]
11:00	187.2s	[157.3-195.2s]	7.18	[6.35-9.35]
12:00	119.6s	[105.3-136.5s]	4.24	[2.94-4.52]
13:00	81.9s	[76.7-91.0s]	1.94	[1.76-2.41]
14:00	104.0s	[93.6-117.0s]	5.06	[4.82-5.76]
15:00	237.9s	[230.1-273.0s]	16.24	[15.29-19.65]
16:00	211.9s	[196.3-219.7s]	14.88	[13.41-15.88]
17:00	201.5s	[184.6-211.9s]	19.41	[16.35-21.29]
18:00	131.3s	[114.4-148.2s]	8.71	[6.88-9.47]
19:00	123.5s	[113.1-128.7s]	6.59	[6.41-7.59]
20:00	85.8s	[75.4-91.0s]	3.88	[3.47-4.06]
21:00	109.2s	[94.9-119.6s]	4.06	[3.76-4.35]
22:00	54.6s	[45.5-61.1s]	2.23	[2.12-2.29]

where n, k, m, g, as previous defined, serve as decision variables, cpm denotes the cost of one single PM, $\boldsymbol{cbf} : \boldsymbol{N}^+ \rightarrow \boldsymbol{Real}$ is a function to identify the cost of the request buffer, $\boldsymbol{cm} : \boldsymbol{N}^+ \rightarrow \boldsymbol{Real}$ is a function to identify the cost of maintaining a certain level of multiplexing. Request arrival rate, VM instantiation rate, abandonment rate of impatient request, faulty rate and resubmission rate are given and serve as input constants in the optimization.

The proposed optimization problems are examples of non-linear, integer (or discrete) programming problems. As n, c, m, g are allowed to be integer, the overall problem is a nonlinear, integer programming problem. In general, it can be shown that all integer programming problems belong to the class of NP-hard problems.

We employ the simulated annealing approach to solve the proposed optimization problem. Given a current state i with energy E_i, a next state $(i+1)$ with energy E_{i+1} is produced through perturbation . If the energy difference between the two consecutive states is smaller than or equal to zero then the state $i+1$ is accepted as the current state. If it is greater than zero, then the state $i+1$ is accepted with probability $exp((E_i - E_{i+1})/k_B T)$, where k_B is the Boltzmann constant and TP the temperature. This rule of accepting the new state is called Metropolis criterion and the algorithm is known as the Metropolis algorithm.

The process of finding the optimal solution for combinatorial optimization problems has an interesting analogy with the physical annealing process. Solutions in a combinatorial optimization problem are equivalent to the states of the solid while the cost of a solution is equivalent to the energy of a state. Given the current solution i with cost $f(i)$, a next solution $(i+1)$ with cost $f(i+1)$ is generated using a generation mechanism from the neighborhood of the current solution. The probability of accepting the next solution is 1 if $f(i+1) \leq f(i)$, oth-

Algorithm optimize_T: Algorithm for solving the cost-aware performance optimization problem.

Input: (1) Expected response time calculation model; (2) Rejection rate computation model; (3) $n^{(init)}$, $c^{(init)}$, $m^{(init)}$, $g^{(init)}$: initial guesses for the number of PMs, the size of request buffer, the maximum number of concurrent VMs that a PM can support, the maximum number of resubmitted tasks that the cloud management unit can concurrently support; (4) the cost models of PMs, request buffer, and multiplexing level; (5) upper and lower bounds of g; (6) TP_{low} the lower bound of temperature; (7) n_I: maximum number of iterations per temperature; (8) a: temperature scale factor.

Output: Optimal numbers of n, c, m, g that minimize T.

```
1. declare $TP$: temperature;
2. declare $i$: number of iterations;
3. declare $TV$: vector of $T$;
4. declare $r$: random number;
5. declare $d$: number of past solutions that need to be checked;
6. $n <- n^{(init)}, c <- c^{(init)}, m <- m^{(init)}, g <- g^{(init)}$;
7. $T <- T(n, c, m, g)$;
8. $TP <- T$;
9. while $TP > TPlow, GLO < g < GUP, cbf(c) < CBUF, n*cpm < CPM,$
   $R(n, c, m, g) < RJ, cm(m) < CM$
10.    $i <- 1$;
11.    while $i <= n_I$
12.        $(n', c', m', g') <- generate\_neighbor(n, c, m, g)$;
13.        $T' <- T(n', c', m' g')$;
14.        $r <- rand()$;
15.        if $T' < T$ or $e^{(T-T')/TP} > r$
16.            $n <- n', c <- c', m <- m', g <- g'$;
17.            $T <- T'$;
18.        $i <- i+1$;
19.    $TV <- store\_last\_T(TV,T)$;
20.    if $check\_termination\_criteria(TV,d)$
21.        break;
22.    $TP <- TP*a$;
23. print $n, c, m, g, T$;
```

Fig. 8: The simulated-annealing-based algorithm

Table 2: Optimal solutions of the cost-aware expected-response-time minimization problem

Input condition	Constraints	Optimal solution	minimized T
$\lambda = 0.0118, \mu = 0.01, \mu' = 0.01, f = 0.2, \theta = 0.24$	$RJ = 0.007, CPM = 1.2, CBUF = 3, CM = 1.4, GLO = 3, GUP = 4$	$c = 15, n = 1, m = 7, g = 4$	$T = 128.5959s$
$\lambda = 0.0222, \mu = 0.0085, \mu' = 0.01, f = 0.2, \theta = 0.24$	$RJ = 0.007, CPM = 4.8, CBUF = 3, CM = 5.6, GLO = 2, GUP = 4$	$c = 16, n = 3, m = 15, g = 4$	$T = 185.6574s$
$\lambda = 0.01, \mu = 0.0043, \mu' = 0.02, f = 0.08, \theta = 0.17$	$RJ = 0.001, CPM = 4.8, CBUF = 12, CM = 5.6, GLO = 5, GUP = 8$	$c = 1, n = 2, m = 13, g = 8$	$T = 50.5507s$

erwise, the next solution is accepted with probability $exp((f(i) - f(i+1)/TP))$. TP is the temperature or the control parameter. Thus, simulated annealing can be viewed as an iteration of Metropolis algorithm evaluated at decreasing values of the temperature. Although simulated annealing does not guarantee 100% optimality of the solution, it can be proved that asymptotically it converges to the global optimal solution.

For example, we assume the cost function of cbf and cm are as follows:

$$cbf(c) = \begin{cases} 1 & if \;\; 0 < c \leq 8 \\ 1.5 & elseif \;\; 9 < c \leq 32 \\ 2.2 & elseif \;\; 33 < c \leq 64 \end{cases} \tag{31}$$

$$cm(c) = \begin{cases} 1 & if \;\; 0 < c \leq 8 \\ 1.6 & elseif \;\; 9 < c \leq 16 \\ 4.3 & elseif \;\; 17 < c \leq 32 \end{cases} \tag{32}$$

Based on the above cost functions, we employ the algorithm shown in Fig.8 to derive optimal solutions for different parameter settings given in Table 2.

7 Conclusions and Further Studies

A comprehensive performance-determination model is proposed in this work. for failure/error-prone IaaS cloud data-centers with request rejection and resubmission. We consider expected request response time and request rejection as the performance metrics and study the impact of varying system conditions (error intensity, VM instantiation rate, multiplexing ability, request load, etc.) on cloud performance. For the model validation purpose, we conduct a confidence interval check based on performance test results of a real-world cloud application. For the optimal capacity planning purpose, we formulate the proposed performance model into a optimization problem. aiming at minimizing expected response time with constraints of request rejection rate and system capacity cost (in terms of the cost of physical machines and the request buffer). We show that the optimization problem can be solved through a simulated-annealing-based algorithm.

References

1. J. Rao, Y. D. Wei, J. Y. Gong, C. Z. Xu.: QoS Guarantees and Service Differentiation for Dynamic Cloud Applications.J.IEEE Trans. Netw. Service Manag., vol. 10, pp. 43–55 (2013)
2. S. Ostermann, R. Iosup, N. Yigitbasi, T. Fahringer.: A performance analysis of EC2 cloud computing services for scientific computing. Proc. Int. Conference on Cloud Computing, pp. 931–945 (2009)
3. N. Yigitbasi, A. Iosup, D. Epema, S. Ostermann.:C-meter: A framework for performance analysis of computing clouds. Proc. Int. Symp. on Cluster Comput. and the Grid, pp. 472–477, (2009)
4. E. Deelman, G. Singh, M. Livny, J. B. Berriman, J. Good.:The cost of doing science on the cloud: the Montage example. Proc. Int. Conf. High Perf. Comput. Netw. Storage and Anal., pp. 1–12 (2008)
5. 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. IEEE Trans. Indus. Info.,vol. 11, pp. 495–504 (2015)
6. K. Xiong, H. Perros.:Service performance and analysis in cloud computing. Proc. World Conf. on Services-I, pp. 693–700 (2009)
7. Y. S. Dai, B. Yang, J. Dongarra, G. Zhang.: Cloud service reliability: Modeling and analysis. Proc. IEEE Pacific Rim Int. Symp. On Dependa. Comput., pp. 784–789 (2009)
8. B. Yang, F. Tan, Y. S. Dai, S. C. Guo.: Performance evaluation of cloud service considering fault recovery. Proc. Int. Conf. on Cloud Comput., pp. 571–576 (2009)
9. X. Meng.:Efficient Resource Provisioning in Compute Clouds via VM Multiplexing. Proc. Int. Conf. Aut.Comput. (ICAC2010). pp. 11–20 (2010)
10. R. Zhu, Y. Zhu.: Performance Analysis of Call Centers Based on M/M/s/k+G Queue with Retrial, Feedback and Impatience. Proc.Int. Conf. on Grey Sys. and Inte. Services, pp. 1779–1784 (2009)