Available online at www.ijpe-online.com

International Journal of Performability Engineering

vol. 13, no. 3, May 2017, pp. 330-333
DOI: 10.1555/10.23940/ijpe.17.03.p9.315318

Performability Analysis of a Parallel Service Considering Multiple Types of Failures

Shengji Yu[a] and Xiwei Qiu[a,*]

[a]*School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, China*

Abstract

Parallel computing is an important approach to achieve a high throughput of serving user requests, which has significant influence on improving performance. Parallel computing service can be realized by hosting multiple copies of the software that performs the same service tasks on different physical machines running in parallel. However, the execution of the software may be interrupted by various kinds of failures, including software failures, hardware failures, and common cause failures (CCF) of co-located copies of the software caused by the failures of the host machine. To analyze the performability of a parallel service, unexpected change of performance caused by random failures and subsequent process of recovery should be counted. This paper presents a theoretical modeling approach encompassing Markov reward models to analyze the performability of a parallel service, which considers software failures, hardware failures, and common cause failures to ensure high fidelity. Simulation results are illustrated to verify the new model.

Keywords: performability; reliability; performance; common cause failure

(Submitted on January 22, 2017; First Revised on March 1, 2017; Final Revised on April 18, 2017; Accepted on April 19, 2017)

1. Introduction

Parallel computing is an essential approach to improve service performance [1]. In general, a parallel service has multiple copies of the software performing the same service. Those copies of the software can be hosted on multiple physical machines for running user requests in parallel. However, in practice, the performance metric is significantly affected by different random factors, particularly, software failures and hardware failures [2][3].

Recently, cloud computing technology enables that multiple copies of the software can also be hosted on an identical physical machine by using the multiple virtual machines for utilizing resources more efficiently. Although virtualization can ensure the isolation of co-located copies of the software, it also brings another complicated case of failures, i.e., CCFs of co-located copies of the software running on the same host which might malfunction due to certain hardware failures. Previous research seldom considered such new situation that causes strong correlation of failures. Also, reliability and performance are often studied separately. Nevertheless, to make more precise performance evaluation, the important reliability-performance correlation should be studied given multiple types of software/hardware failures, especially, common cause failures.

In this paper, we present a new performability model for analyzing the important R-P correlation of a parallel service. Multiple types of failures including software failures, hardware failures, and CCFs are fully taken into account to ensure high fidelity of the proposed model.

* Xiwei Qiu. Tel.: 159-8218-7255.
E-mail address: qiu_uestc@hotmail.com.

2. Description of the Parallel Service

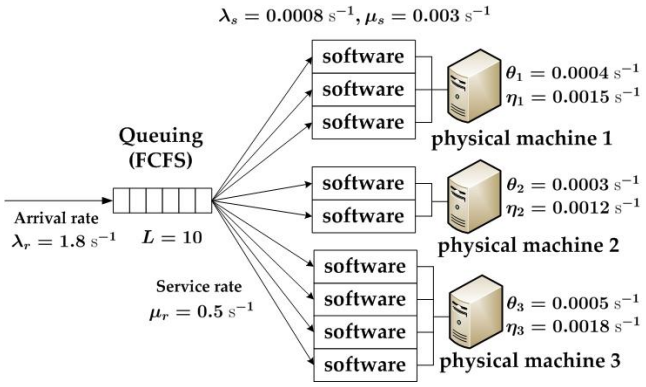


Figure 1. A scenario of the parallel service.

Fig. 1 describes a scenario of the parallel service. There are three heterogeneous physical machines can be used to host the copies of the software. As shown in Fig. 1, machine 1, 2, and 3 host three, two, and four copies of the software, respectively. Thus, the parallel service can serve nine user requests simultaneously. The parallel service keeps a request queue and the arrived requests are served on First-Come-First-Serve (FCFS) discipline.

We make the following assumptions for the performability model [4][5]:

1) The arrival of the user requests follows a Poisson distribution with the arrival rate λ_r. The service time of a user request is governed by an exponential distribution with the mean time $1/\mu_r$.
2) The hardware failures and recovery of machine n follow Poisson processes with the failure rate θ_n and the repair rate η_n, respectively. Once a hardware failure occurs, all co-located copies of the software on the machine are terminated. Once the failed machine is recovered, the co-located copies of the software are restored.
3) All copies of the software are homogeneous, which means the software failures and recovery of the copies follow the same Poisson processes with the same failure rate λ_s and the same repair rate μ_s, respectively.

3. Markov Model for Multiple Types of Failures

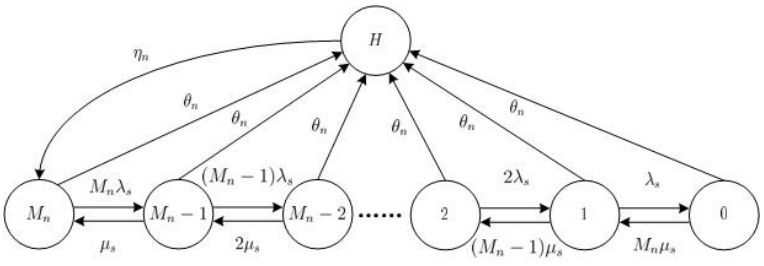


Figure 2. Markov model for analyzing multiple types of failures.

We first present a Markov Model for analyzing multiple types of failures existed in a physical machine. Suppose physical machine n hosts M_n copies of the software. The corresponding Markov model is shown in Fig. 2. The state m ($m = M_n,\ldots,1,0$) represents that the number of available copies of the software is m. The state H represents that a hardware failure of the machine occurs, which results in all copies are down (i.e., CCFs of the co-located copies). Denote π_m and π_H as the steady probabilities for the model to stay at state m and H, respectively. The corresponding *Chapman-Kolmogorov* equations can be derived as

$$(M_n\lambda_s + \theta_n)\pi_{M_n} = \eta_n\pi_H + \mu_s\pi_{M_n-1} \tag{1}$$

$$((M_n - m)\mu_s + m\lambda_s + \theta_n)\pi_m = (m+1)\lambda_s\pi_{m+1} + (M_n - m + 1)\mu_s\pi_{m-1}, \quad m = M_n - 1, M_n - 2,\ldots,2,1 \tag{2}$$

$$(M_n\mu_s + \theta_n)\pi_0 = \lambda_s\pi_1 \tag{3}$$

$$\eta_n \pi_H = \theta_n \sum_{m=0}^{M_n} \pi_m \tag{4}$$

$$\sum_{m=0}^{M_n} \pi_m + \pi_H = 1 \tag{5}$$

Denote X_n as the number of available copies in physical machine n. After solving (1)-(5), the probability mass function (pmf) of X_n can be further obtained as

$$p_n(x) = \Pr(X_n = x) = \pi_x \qquad (x = 1,2,...,M_n) \tag{6}$$

$$p_n(0) = \Pr(X_n = 0) = \pi_0 + \pi_H \tag{7}$$

Now, for the entire parallel service supported by N heterogeneous physical machines, its parallel service ability can be quantified as the number of all available copies, which can be denoted as X. Since $X = X_1 + X_2 + \cdots + X_n$, its pmf can be derived as

$$p(x) = \Pr(X = x) = \sum_{x_1+\cdots+x_N=x}^{x_1,\ldots x_N} p_n(x_n) \tag{8}$$

4. Analysis of the Performability

To evaluate the correlation between reliability and performance, we first set X as an input parameter during performance modelling, and then use a Bayesian approach to remove it. Given the condition that $X = x$, the performance model described by a birth-death process is shown in Fig. 3.

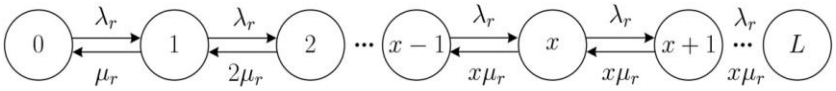


Figure 3. Performance model of the parallel service given $X = x$.

The state i ($i = 0,1,...,L$) represents the number of requests in the queue, where L is the sum of the length the queue and the maximal number of requests can be served simultaneously. If $i < x$, all i requests can be immediately served by available copies of the software. If $i \geq x$, only x requests can be served, so the exit rate of a user request is $x\mu_r$ for all $i \geq x$. Denote q_i as the steady probability for the parallel service staying at state i, which can be derived as

$$q_0 = \left[\sum_{i=0}^{x-1} \frac{\lambda_r^i}{i!\mu_r^i} + \sum_{i=x}^{L} \frac{\lambda_r^i}{x^{i-x} x!\mu_r^i} \right]^{-1} \tag{9}$$

$$q_i = \frac{\lambda_r^i}{i!\mu_r^i} q_0 \qquad (i = 1,2,....,x-1) \tag{10}$$

$$q_i = \frac{\lambda_r^i}{x^{i-x} x!\mu_r^i} q_0 \qquad (i = x, x+1,...., L) \tag{11}$$

If the request queue is full, a new arrival request is discarded so as to avoid overflowing of the queue, which leads to a blocking failure. Thus, given the condition that $X = x$ the conditional performance of the parallel service can be obtained as

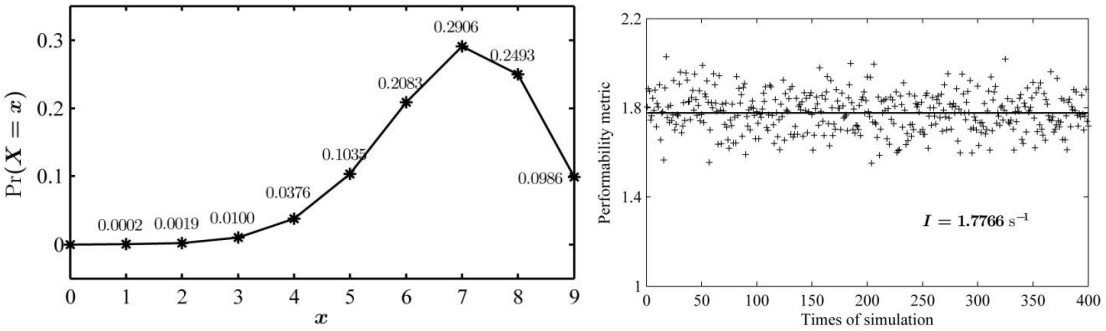
$$I \mid x = \lambda_r (1 - q_L) \tag{12}$$

where q_L can be calculated from (9) and (11). Since the pmf of X has been derived from (8), we can further implement a Markov reward model to analyze performability. The performability metric of the parallel service is defined as the mean time for successfully serving a user request (i.e., $T = 1/I$). In addition to steady probability $p(x)$, each state x also has a reward value $I \mid x$. Now, the performability of the parallel service is evaluated as

$$T = \frac{1}{\sum_{x=1}^{N} p(x) \cdot I \mid x} \tag{13}$$

5. Examples

Take the scenario shown in Fig. 1 as an example. From (1)-(7), the pmf of X_1 can be derived as $p_1(3) = 0.4613$, $p_1(2) = 0.3906$, $p_1(1) = 0.1263$, $p_1(0) = 0.0218$. In a similar way, the pmf of X_2 and X_3 also can be derived. Then, the pmf of X can be derived as shown in Fig. 4 (a). From (13), the performability metric of the parallel service can be evaluated as $T = 0.5629$ s, which also implies the expected throughput of the parallel service is $I = 1.7766\ \mathrm{s}^{-1}$. To verify analytical results obtained by our performability model, a simulation program has been developed, which is developed to simulate the service process of 500 requests of the parallel service. The comparison between simulation results obtained by running the program 400 times and theoretical results is shown in Fig. 4 (b), which witness that the proposed performability model is justified.



(a) (b)

Figure 4. (a) Pmf of X, (b) Comparison between simulation results and theoretical results for the expected throughput of the parallel service.

6. Conclusions

In this paper, we systemically studied the performability model of a parallel service for capturing the important R-P correlation. The model also considers multiple types of failures, including hardware failures, software failures, and CCFs of co-located copies of the software caused by failures of the host machine. Numerical examples illustrated the analysis of the performability of the parallel service, and the proposed performability model was further verified by simulation results.

References

1. R. A. Kendall, E. Apra, D. E. Bernholdt, E. J. Bylaska, and M. Dupuis. "High Performance Computational Chemistry: An overview of NWChem a Distributed Parallel Application," *Computer Physics Communications*, vol. 128, no. 1, pp. 260-283, 2000
2. J. F. Meyer, "On Evaluating the Performability of Degradable Computing Systems," *IEEE Transactions on Computer*, vol. 100, no. 8, pp. 720-731, 1980
3. B. Yang, F. Tan, and Y. S. Dai, "Performance Evaluation of Cloud Service Considering Fault Recovery," *Journal of Supercomputing*, vol. 65, no. 1, pp. 426-444, 2013
4. K. S. Trivedi, "Probability and Statistics With Reliability, Queuing, and Computer Science Applications," Wiley, New York, 2001
5. X. Ni, J. Zhao, W. Song, and H. Li, "Reliability Modeling for Two-Stage Degraded System Based on Cumulative Damage Model," *International Journal of Performability Engineering*, vol. 12, no. 1, pp. 89-94, 2016

Shengji Yu is currently pursuing the Ph. D. degree in computer science with the School of Computer Science and Engineering, University of Electronic Science and Technology of China (UESTC). His current research interests include cloud computing, decision making, reliability, and autonomic computing.

Xiwei Qiu is currently a postdoctoral research fellow at the Collaborative Autonomic Computing (CAC) laboratory at UESTC. His research interests include cloud computing, reliability, modeling and optimization, and energy-efficient computing.