Available online at www.ijpe-online.com

International Journal of Performability Engineering

vol. 13, no. 3, May 2017, pp. 334-336
DOI: 10.1555/10.23940/ijpe.17.03.p10.319321

Simulated Software Testing Process Considering Debuggers with Different Detection and Correction Capabilities

RUI PENG* and JUNTING LIU

Donlinks School of Economics & Management, University of Science & Technology Beijing, Beijing, China

Abstract

Software reliability growth models (SRGMs) have evolved from describing fault detection process (FDP) into incorporating fault correction process (FCP) as well. Restricted by mathematical tractability, analytical models are facing difficulties for more accurate description the real world situations, e.g. debuggers being different in terms of debugging capabilities and experiences. In this paper, a simulation approach is proposed to model FDP and FCP together considering debuggers with different contributions to the fault detection rate and different fault correction time.

Keywords: software reliability; non-homogeneous Poisson processes; queuing systems; non-homogeneous debuggers

(Submitted on August 13, 2016; First Revised on Oct. 30, 2016; Second Revised on Nov. 1, 2016; Accepted on November 10, 2016)

1. Introduction

Numerous software reliability growth models (SRGMs) have been proposed during the past three decades, among which non-homogeneous Poisson process (NHPP) models are considered as the most effective [1-3]. In order to provide mathematical tractability, SRGMs are always derived under restrictive assumptions, which confine their broad applications [4-6].

One common assumption is the immediate fault correction assumption. In reality, faults can only be corrected after they are detected. Thus FCP can be regarded as a delayed FDP and can be modeled together with FCP using a queuing system [7-9]. Besides, it is almost impossible to incorporate into the analytical models some important information, such as the different debugging capabilities of different debuggers. In [10], the number of debuggers is incorporated into FDP and FCP through a rate-based simulation approach. However, it is assumed that all debuggers have the same fault correction ability. In [11], the influence of the different debuggers on the fault correction process is simulated, however, with restricted to exponential fault correction time. In addition, the influence of different debuggers on the fault detection process is not considered.

In this paper, the assumption that all the debuggers have the same fault detection and correction capabilities is relaxed. The fault detection rate is assumed to be a function of the numbers of different debuggers and their respective fault detection rates, whereas the fault correction time of different debuggers are also assumed to be different. The general framework of the queuing model is proposed in Section 2, with description of the simulation framework. Numerical examples are shown in Section 3 to illustrate the simulation approach. Section 4 concludes.

2. The Modelling Framework

The whole fault detection and correction processes can be described using a queuing system, with FDP being the arrival process and the FCP being the departure process. The FDP is described as a NHPP characterized with intensity function $\lambda d(t)$ and mean value function md(t). It is assumed that there are totally k types of debuggers and the number of the i-th type debuggers is SN(i). The fault detection rate is assumed to be constant and it is a function of SN=[SN(1),…,SN(k)], denoted as b=f(SN(1),…,SN(k)). In practice, the form of this function can be estimated from similar projects or by experts.

The time for a type-i debugger to correct a fault is random, and it is allowed to observe arbitrary distribution. Figure 1 illustrates the fault detection and correction processes.

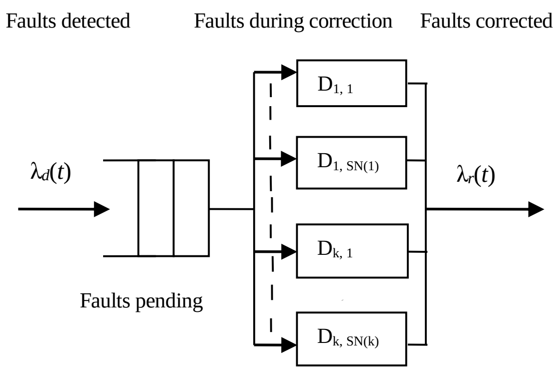

Figure 1: Software FDP and FCP Described as a Queuing System

If we use a to denote the expected number of faults initially embedded in the software, the mean value function for the FDP can be given as the GO NHPP model [12]:

$$m_d(t) = a(1-e^{-bt}) \qquad (1)$$

The FDP is simulated in the time interval of (0, T]. Since the general order statistics (GOS) models are closely related to NHPP models [13], simulation of FDP can be implemented through generation of a Poisson number of order statistics from a fixed cumulative density function of F(t)=md(t)/md(T), 0≤t≤T [14]. The simulation of the fault correction time is straightforward, with traditional inverse transformation approach. The FCP can be obtained based on the simulated FDP and the fault correction time.

3. Numerical Example

In this section, a specific model configuration is shown for illustration. It is assumed that there are totally two different kinds of debuggers. The parameters for the NHPP model of FDP are configured as a = 100 and b = 0.02SN(1)+0.01SN(2). The corrections for the two types of debuggers both observe lognormal distribution with parameters $(\mu,\sigma) = (0.3,0.1)$ and $(\mu,\sigma) = (0.6,0.1)$ respectively. For two different staffing levels SN = [5,3] and SN=[3,5], the simulation replicates 100 times in the time interval of (0, 100]. Then the collected fault detection and correction time data are grouped into the cumulative fault detection number d_i^j and fault correction numberc_i^j, $i = 1,\cdots,100, j = 1,\cdots,100$. Here i denotes the number of a simulation run, and j denotes the testing time till which the fault detection number and fault correction number are cumulated. Finally we can have the average cumulative number of detected faults and corrected faults as

$$\begin{cases} \bar{d}_i = \dfrac{1}{100}\sum_{j=1}^{100} d_i^j \\ \bar{c}_i = \dfrac{1}{100}\sum_{j=1}^{100} c_i^j \end{cases}, \quad i = 1,\ldots,100;\ j = 1,\ldots,100. \qquad (2)$$

The simulation results are shown in Figure 2. Note that the time line starts from 1 instead of 0 so that the fault detection number at the beginning can be positive. It can be seen clearly that when there are more debuggers of the first type, who can contribute more to fault detection rate and correct faults faster, the mean value functions of FDP and FCP are considerably higher.

4. Conclusions

This paper simulates the software fault detection and correction processes considering the impact of debuggers' different capabilities on the fault detection rate and the fault correction time. In the future, we will study the optimal combination of different types of debuggers and the optimal software release time, considering both the software reliability and the cost function.

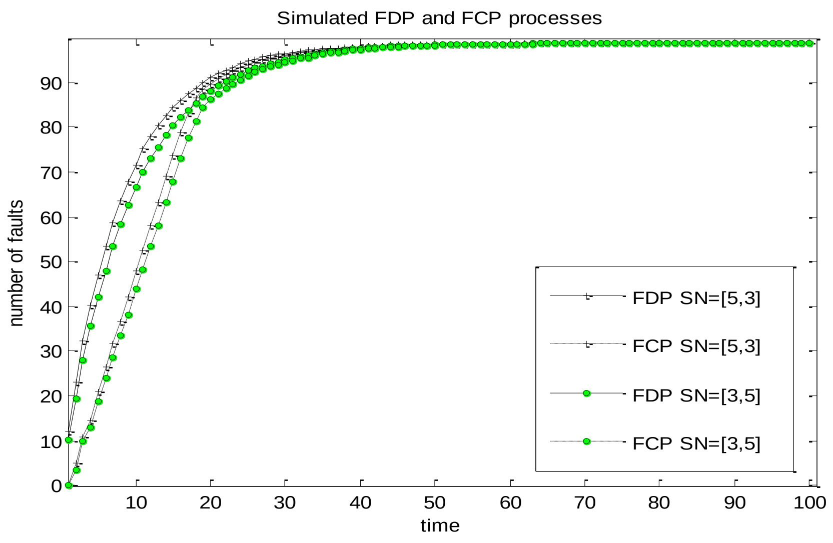

Figure 2: Simulated FDP and FCP Processes

References

1. Pham, H. System Software Reliability. London, U.K.: Springer-Verlag 2006.
2. Xiao, X., H. Okumura, and T. Dohi. NHPP-Based Software Reliability Models using Equilibrium Distribution. IEICE Transactions on Fundamentals of Electronics Communications and Computer Sciences 2012; E95A (5):894-902.
3. Peng, R., Y. F. Li, W. J. Zhang, and Q. P. Hu. Testing Effort Dependent Software Reliability Model for Imperfect Debugging Process Considering Both Detection and Correction. Reliability Engineering & System Safety 2014; 126: 37-43.
4. Shibata, K., K. Rinsaka, T. Dohi. M-SRAT: Metrics-based software reliability assessment tool. International Journal of Performability Engineering 2015; 11 (4): 369-379.
5. Bisi, M., N. Goyal. Early prediction of software fault-prone module using artificial neural network. International Journal of Performability Engineering 2015; 11 (1), 43-52.
6. Xiao, X., Dohi, T. On the role of Weibull-type distributions in NHPP-based software reliability modeling. International Journal of Performability Engineering 2013; 9 (2), 123-132.
7. Xie, M., Q. P. Hu, Y. P. Wu, and S.H. Ng. A Study of the Modeling and Analysis of Software Fault-Detection and Fault-Correction Processes. Quality and Reliability Engineering International 2007; 23 (4): 459-470.
8. Wu, Y. P., Q. P. Hu, M. Xie, and S.H. Ng. Modeling and Analysis of Software Fault Detection and Correction Process by Considering Time Dependency. IEEE Transactions on Reliability 2007; 56 (4): 629-642.
9. Huang, C. Y., and W. C. Huang. Software Reliability Analysis and Measurement Using Finite and Infinite Server Queueing Models. IEEE Transactions on Reliability 2008; 57 (1): 192-203.
10. Lin, C. T., and C. Y. Huang. Staffing level and cost analyses for software debugging activities through rate-based simulation approaches. IEEE Transactions on Reliability 2009; 58 (4): 711-724.
11. Peng, R., F. Shahrzad. Simulation of software fault detection and correction processes considering different skill levels of debuggers. Proceedings of IEEE Pacific Rim International Symposium on Dependable Computing, PRDC 2014; 157-158.
12. Goel, A. L., and K. Okumoto. Time-Dependent Error Detection Rate Model for Software Reliability and Other Performance Measures. IEEE Transactions on Reliability 1979; 28: 206-211.
13. Kuo, L., and T. Y. Yang. Bayesian Computation for Nonhomogeneous Poisson Processes in Software Reliability. Journal of the American Statistical Association 1996; 91 (434): 763-773.
14. Lewis, P. A. W., and G. S Shedler. Simulation of Nonhomogeneous Poisson Processes by Thinning. Naval Research Logistics Quarterly 1979; 26 (3): 403-413.

Rui Peng received his Ph. D degree from Industrial & Systems Engineering Department, National University of Singapore in 2011. He is currently an associate professor in Donlinks School of Economics & Management, University of Science & Technology Beijing. His research interests include software reliability, networked system reliability, and defense strategies.

Junting Liu is an undergraduate student in Donlinks School of Economics & Management, University of Science & Technology Beijing.