

## On Estimation of Number of Detectable Software Faults under Budget Constraint

Shinji Inoue<sup>1\*</sup>, Kenta Hotta<sup>2</sup>, Shigeru Yamada<sup>1</sup>

Department of Social Management Engineering  
Graduate School of Engineering, Tottori University  
4-101, Minami, Koyama-Cho, Tottori-Shi, Tottori 680-8552, Japan  
E-mail:<sup>1</sup>{ino, yamada}@sse.tottori-u.ac.jp, <sup>2</sup>M16T7006A@edu.tottori-u.ac.jp  
*\*Corresponding author*

(Received October 10, 2016; Accepted December 2, 2016)

### Abstract

Checking software quality and reliability of developed software system is one of the important activities for developing highly-reliable software system. In software testing phase, which is located in the final stage of software development process, software faults remaining the developed software system must be eliminated as possible as the testing manager could. Further, it is also important to develop testing plan to conduct testing activities efficiently under the certain constraints because there are some constraints, such as cost and delivery, in the actual testing phase. This paper discusses method to estimate the number of detectable faults under constant budget constraint by applying a two-dimensional Weibull type software reliability model, in which the software reliability growth process depends on testing time and testing effort factors. We finally show numerical examples of our method by using actual data.

**Keywords**—Two-dimensional Weibull-type software reliability model, Testing time, Testing effort, Optimization problem, Sensitivity analysis.

### 1. Introduction

Software system is recognized as the necessary infrastructure of our social life. Therefore, software is required high quality/reliability in despite of increasing the scale and the complexity of software. Further, saving the development cost and the quick delivery are also required in most of software development industries. Under such situation, it is required to measure, assess and predict software quality/reliability of the developed software at the final stage of testing. This must lead to software project management for delivering highly-reliable software to the user. Software reliability models (abbreviated as SRMs) (Musa et al., 1987; Pham, 2000; Yamada, 2014) are ones of the useful mathematical tools for measuring, assessing, and predicting software reliability by describing the software failure-occurrence or software fault-detection phenomenon. On the other hand, test planning must play an important role in making testing activities more effective. In the test planning, the test manager estimates shipping time and the testing effort to be expended to satisfy the required reliability objective. At this time, the test manager must be interested of how many faults can be detected as against to the total number of faults remaining before the testing-phase.

This paper discusses methods for estimating the testing time and effort for the purpose of maximizing the number of detectable faults under the budget constraint. Concretely, we use a bivariate (or two-dimensional) Weibull-type SRM (Inoue and Yamada, 2008; Inoue and Yamada, 2009), in which the software reliability growth process depends on both of the testing-time and the testing-effort expenditure, for describing the software reliability growth process observed. There must be no doubt that the software reliability growth process is supported by the testing-effort factors, due to the software debugging process in the actual testing phase. At this time, we must

have an interest of how many faults can we eliminate under the certain budget constraint. In this paper, we formulate optimization problems maximizing the number of detectable software faults under the budget constraints. Finally, we will offer numerical examples of our approach and give considerations of our methods by using actual data.

## 2. Bivariate Weibull-type Software Reliability Growth Model

### 2.1 Conventional Weibull-Type Model

Let  $\{Z(t), t \geq 0\}$  denote a counting process representing the number of faults detected up to testing-time  $t$ . Assuming  $Z(t)$  obeys the nonhomogeneous Poisson process (abbreviated as NHPP), we formulate the stochastic behavior of  $Z(t)$  as

$$\begin{cases} \Pr\{Z(t) = n\} = \frac{\{\theta(t)\}^n}{n!} \exp[-\theta(t)] \quad (n = 0, 1, 2, \dots), \\ \theta(t) = \int_0^t \mu(x) dx. \end{cases} \quad (1)$$

In Eq. (1),  $\Pr\{A\}$  is defined as the probability of event  $A$ ,  $H(t)$  is the mean value function representing the expected number of faults detected up to testing time  $t$ ,  $\mu(t)$  is the intensity. In the Weibull-type SRM (Keller and Miller, 1991), the mean value function is given by

$$\theta(t) \equiv \gamma(t) = \left(\frac{t}{\rho}\right)^\beta \quad (0 < \beta < 1; \rho > 0), \quad (2)$$

We call this NHPP model a Weibull-type model depending only the testing-time. In Eq. (2),  $\beta$  represents the reliability growth parameter,  $\rho$  the scale parameter, and  $\gamma(0) = 0$  and  $\gamma(\infty) = \infty$ . That is, infinite number of faults are detected during infinitely long test duration. Such situation is observed under the imperfect debugging environment in which the debugging activities is not perfect.

### 2.2 Bivariate Weibull-type Software Reliability Growth Model

In Eq. (2), the testing-time is measured on the basis of number of days or weeks, and so forth. In that case, it is hard to say that Eq. (2) reflects actual software reliability growth process, that is, we need to incorporate other factors influencing the software reliability growth process, such as the number of test-cases, the attained test coverage, for conducting highly accurate software reliability measurement, assessment and prediction. From such reason, a bivariate software reliability model has been proposed (Inoue and Yamada, 2008; Inoue and Yamada, 2009). Now, we define the two-dimensional stochastic process  $\{Z(s, u), s \geq 0, u \geq 0\}$  representing the number of faults detected up to the testing time  $s$  and the amount of testing effort expenditures  $u$ . Further, we assume the two-dimensional stochastic process  $\{Z(s, u), s \geq 0, u \geq 0\}$  follows

$$\begin{cases} \Pr\{Z(s, u) = n\} = \frac{\{\theta(s, u)\}^n}{n!} \exp[-\theta(s, u)] \quad (n = 0, 1, 2, \dots), \\ \theta(s, u) = \int_0^s \int_0^u \mu(x, y) dx dy, \end{cases} \quad (3)$$

where  $\theta(s, u)$  and  $\mu(s, u)$  represent the mean value function and the intensity. In the bivariate Weibull-type SRG (Inoue and Yamada, 2008), the bivariate mean value function is given by

$$\theta(s, u) \equiv \gamma(s, u) = \left( \frac{s^\alpha u^{1-\alpha}}{\rho} \right)^\beta$$

$$(0 \leq \alpha \leq 1, 0 < \beta < 1; \rho > 0), \quad (4)$$

where  $\alpha$  is the effect of the testing time factors to the software reliability growth process. When  $\alpha = 0$ , Eq. (4) is regarded as the testing-effort dependent SRGM (Yamada et al., 1993). And  $\alpha = 1$ , it is equivalent to the conventional one-dimensional Weibull-type SRGM in Eq. (2) essentially.

### 2.3 Parameter Estimation

Suppose that we have observed fault count data consisting of  $N$  data pairs,  $(s_i, u_i, y_i) (i = 0, 1, 2, \dots, N)$  with respect to the total number detected faults,  $y_i$ , detected during a constant interval  $(0, \Phi_i]$ , where  $\Phi_i \equiv (s_i, u_i)$ . From Eq. (4), we can obtain the following regression equations

$$\log \gamma(s, u) = -\beta \log \rho + \alpha \beta \log s + (1 - \alpha) \beta \log u, \quad (5)$$

by taking the natural logarithm of Eq. (4). And we then have the following multiple regression equation:

$$Y_i = a_0 + a_1 K_i + a_2 L_i + \varepsilon_i, \quad (6)$$

$$\begin{cases} Y_i = \log y_i, \\ K_i = \log s_i, \\ L_i = \log u_i, \\ a_0 = -\beta \log \rho, \\ a_1 = \alpha \beta, \\ a_2 = (1 - \alpha) \beta. \end{cases} \quad (7)$$

where, in Eq. (6),  $\varepsilon_i$  is a standard normal error term with homoscedasticity. From Eq's. (6) and (7),  $S(a_0, a_1, a_2)$ , is derived as

$$S(a_0, a_1, a_2) = \sum_{i=1}^N \varepsilon_i^2 = \sum_{i=1}^N \{Y_i - (a_0 + a_1 K_i + a_2 L_i)\}^2. \quad (8)$$

Solving the simultaneous equations, we can get the parameter estimates of  $a_0$ ,  $a_1$ , and  $a_2$ , respectively. Then, we can obtain parameter estimates  $\hat{\alpha}$ ,  $\hat{\beta}$ , and  $\hat{\rho}$  of the parameter  $\alpha$ ,  $\beta$ , and  $\rho$  as

$$\begin{cases} \hat{\alpha} = \frac{\hat{a}_1}{\hat{a}_1 + \hat{a}_2}, \\ \hat{\beta} = \hat{a}_1 + \hat{a}_2, \\ \hat{\rho} = \exp \left[ -\frac{\hat{a}_0}{\hat{a}_1 + \hat{a}_2} \right], \end{cases} \quad (9)$$

by using the estimated partial regression coefficients,  $\hat{a}_0$ ,  $\hat{a}_1$ , and  $\hat{a}_2$ .

### 3. Estimation of Number of Detectable Faults under Budget Constraint

The number of faults detected in testing phase must be influenced by the test duration and the debugging and correcting skill of the software testers, which are affected on the cost. That means the number of detectable faults depends on the budget for the testing activities. We formulate an

optimization problem by using Eq. (4) to estimate the number of detectable software faults under the budget constraint. The optimization problem is given by

$$\text{Max } H(s, u) \text{ subject to } p_s s + p_u u \leq I, \quad (10)$$

where  $p_s$  is the cost for conducting testing activities per testing time,  $p_u$  for expending cost per the testing effort, respectively, and  $I$  is the budget for the testing. Using the Kuhn-Tucker condition for deriving the optimal solution in Eq. (10), we can obtain

$$(s^*, u^*) = \left( \frac{\alpha I}{p_s}, \frac{(1-\alpha)I}{p_u} \right). \quad (11)$$

Substituting Eq. (11) into Eq. (10), we can estimate the maximum number of detectable faults considering with the budget constraint as

$$H(s^*, u^*) = \left\{ \frac{\left( \frac{\alpha I}{p_s} \right)^\alpha \left( \frac{(1-\alpha)I}{p_u} \right)^{1-\alpha}}{\rho} \right\}^\beta \quad (12)$$

$p_s$	$p_u$	$I$	$s^*$	$u^*$	$\hat{H}(s^*, u^*)$
0.1	10	10	17.56	0.824	263.8
0.1	9	10	17.55	0.916	286.6
0.05	10	10	35.11	0.824	296.3
0.05	9	10	35.11	0.916	322.0
0.1	10	9	15.80	0.742	238.5
0.05	10	9	31.60	0.742	267.9
0.1	9	9	15.80	0.824	259.1
0.05	9	9	31.60	0.824	291.1

(i) DS1

$p_s$	$p_u$	$I$	$s^*$	$u^*$	$\hat{H}(s^*, u^*)$
0.1	10	10	29.47	0.705	254.4
0.1	9	10	29.47	0.784	276.5
0.05	10	10	58.94	0.705	285.8
0.05	9	10	58.94	0.784	310.6
0.1	10	9	26.53	0.635	230.0
0.05	10	9	53.05	0.635	258.4
0.1	9	9	26.53	0.705	250.0
0.05	9	9	53.05	0.705	280.8

(ii) DS2

Table 1. Sensitivity analysis for the maximum number of detectable faults under the budget constraints

#### 4. Numerical Example

We show numerical examples for our method to estimate the maximum number of detectable faults under budget constraint. We use the following fault counting data collected in actual software testing: DS1:  $(s_i, u_i, y_i)(i = 0, 1, 2, \dots, 24; s_{24} = 24 \text{ (weeks)}, u_{24} = 0.9095, y_{24} = 296)$  and DS2:  $(s_i, u_i, y_i)(i = 0, 1, 2, \dots, 22; s_{22} = 22 \text{ (weeks)}, u_{22} = 0.9198, y_{24} = 212)$ , where  $u_i$ , the testing effort factor, represents the testing coverage attained up to testing time  $s_i$  (Fujiwara and Yamada, 2002). Table 1 shows the sensitivity analysis for the maximum number of detectable faults under the budget for DS1 and DS2, respectively. Parameters in Eq. (4) have been estimated as  $\hat{\alpha} = 0.17555$ ,  $\hat{\beta} = 0.95652$  and  $\hat{\rho} = 0.00415$  for DS1, and  $\hat{\alpha} = 0.29472$ ,  $\hat{\beta} = 0.95336$  and  $\hat{\rho} = 0.00856$  for DS2. From Table 1, we can see that the maximum number of detectable faults has been increased with the decreasing the values of  $p_s$  or  $p_u$  and with the increasing of the budget  $I$ .

#### 5. Conclusion

This paper discussed a method to estimate the maximum number of detectable faults under the budget constraint based on a bivariate Weibull-type SRGM. Our approach is quite simple approach. However, it might be useful to know how many faults to be detected in the testing under the budget constraint and how much do they need to expend the testing effort and to spend testing time for detecting certain number of faults for developing test plan. Regarding the objective function in our optimization problem, it must be interesting to use the software reliability or MTBF as the objective function from the viewpoint of practical application of our method.

#### Acknowledgement

This research was partially supported by the JSPS KAKENHI (C), Grant No. 16K00098.

#### Reference

- Fujiwara, T., & Yamada, S. (2002). C0 coverage-measure and testing-domain metrics based on a software reliability growth model. *International Journal of Reliability, Quality and Safety Engineering*, 9(4), 329-340.
- Inoue, S., & Yamada, S. (2008). A bivariate Weibull-type software reliability growth model and its goodness-of-fit (in Japanese). *Journal of Information Processing Society of Japan*, 49(8), 2851-2861.
- Inoue, S., & Yamada, S. (2009, December). Two-dimensional software reliability measurement technologies. In *Proceedings of the 2009 IEEE International Conference on Industrial Engineering and Engineering Management*, Hong Kong, China, 223-227.
- Keller, P. A., & Miller, D. R. (1991). On the use and the performance of software reliability growth models. *Reliability Engineering & System Safety*, 32(1-2), 95-117.
- Musa, J. D., Iannio, D., & Okumoto, K. (1987). *Software Reliability, Measurement, Prediction, Application*, McGraw-Hill, New York.
- Pham, H. (2000). *Software Reliability*, Springer-Verlag, Singapore.
- Yamada, S. (2014). *Software Reliability Modeling Fundamentals and Applications*, Springer Tokyo, Japan.
- Yamada, S., Hishitani, J., & Osaki, S. (1993). Software reliability growth with a Weibull test-effort. *IEEE Transactions on Reliability*, 42(1), 100-106.