Hyper-Geometric Distribution Software Reliability Growth Model with Imperfect Debugging

Rong-Huei Hou & Sy-Yen Kuo

Department of Electrical Engineering
National Taiwan University
Taipei, Taiwan, R.O.C.
Email: sykuo@cc.ee.ntu.edu.tw

Yi-Ping Chang

Department of Business Mathematics
Soochow University
Taipei, Taiwan, R.O.C.

Abstract

Debugging actions during the test/debug phase of software development are not always performed perfectly. That is, not all the software faults detected are perfectly removed without introducing new faults. This phenomenon is called the imperfect debugging. The Hyper-Geometric Distribution software reliability growth Model (HGDM) was developed for estimating the number of software faults initially in a program. In this paper, we propose an extended model based on the HGDM incorporating the notion of imperfect debugging.

1 Introduction

Software Reliability Growth Models (SRGMs) have been studied by many authors [1]. Most of the SRGMs proposed are based on the assumption of perfect debugging. However, in reality, not all the software faults detected are perfectly removed without introducing new faults [2]. Most of the existing SRGMs do not take this into account, and only a few researchers extended the SRGMs considering imperfect debugging [3–8].

The Hyper-Geometric Distribution software reliability growth Model (HGDM) for estimating the number of initial software faults, first proposed by Tohma *et al.* [9], is shown to be attractive. Tohma *et. al.* [10–12], Jacoby *et. al.* [13–14], and Hou *et. al.* [15–16] have made a series of studies on the HGDM recently. In this paper, we propose an extended model based on the HGDM incorporating the notion of imperfect debugging. Furthermore, to make the proposed model more realistic and practical, we consider the situation where there is learning implicitly in the fault removal process. The growth curve of the cumulative number of discovered faults and a measure of software reliability for the HGDM with imperfect debugging are investigated. In addition, the relationship between the proposed model and the Goel-Okumoto NHPP model with imperfect debugging is discussed. Experiments have been performed by using two real test/debug data sets, and the results show the proposed model fits the data sets satisfactorily.

2 Review of HGDM

In this section, we briefly review the Hyper-Geometric Distribution software reliability growth Model (HGDM) [9–14]. In general, a program is assumed to have m faults initially before the test/debug phase starts. The collection of test operations performed in a day or a week is called a *"test instance"*. Test instances are denoted by t_i, $i = 1, 2, \ldots, n$ in accordance with the order of applying them. The *"sensitivity factor"*, w_i, represents how many faults are newly discovered or rediscovered during the application of test instance t_i. Some of the faults detected by t_i may have been detected previously by the application of $t_1, t_2, \ldots, t_{i-1}$.

Let C_{i-1} be the number of faults already detected so far by $t_1, t_2, \ldots, t_{i-1}$ and x_i be the number of faults newly detected by t_i. Then, some of the w_i faults may be those already counted in C_{i-1}, and the remaining w_i faults account for the newly detected faults. With the assumption that new faults will not be introduced into the program while correcting is being performed, the conditional probability $P(N_i = x_i \mid m, w_i, C_{i-1})$ can be formulated as:

$$P\left(N_i = x_i \mid m, w_i, C_{i-1}\right) = \frac{\binom{m - C_{i-1}}{x_i}\binom{C_{i-1}}{w_i - x_i}}{\binom{m}{w_i}}, \quad (1)$$

1071-9458/95 $4.00 ©1995 IEEE

where $\max(0, w_i - C_{i-1}) \leq x_i \leq \min(w_i, m - C_{i-1})$ for all $i \geq 0$, $C_{i-1} = \sum_{k=1}^{i-1} x_k$, $C_0 = 0$, $x_0 = 0$ and x_k is an observed instance of N_k. The expected value of C_i denoted by EC_i is [9–14]

$$\begin{cases} EC_0 = 0, \\ EC_i = m[1 - \prod_{j=1}^{i}(1 - p_j)], \ i = 1, 2, \ldots, n, \end{cases} \quad (2)$$

where $p_i = w_i/m$. Although various functions for p_i have been presented, the linear function $p_i = ei + f$ usually gives satisfactory results [11–12].

From Eq.(2), $(EC_i - EC_{i-1})/(m - EC_{i-1}) = p_i$ represents the ability of test workers to detect new faults at the application of t_i. If the test is completely random, that is, test data are taken randomly from the input space of the program, p_i will be a constant. In following sections, we propose an extended model under the assumption that p_i is a constant.

3 HGDM with Imperfect Debugging

In Section 2 we reviewed the HGDM, which assumes the detected faults are instantly and perfectly corrected without introducing any new faults. However, this assumption is not always valid and can be modified to be:

(i) When the detected faults are removed, it is possible to introduce new faults.

(ii) When a fault is newly discovered during the application of t_i, removal of the fault is instantaneous and the following may occur:

  (a) the fault is corrected with probability $1 - \theta_i$;

  (b) a new fault is introduced with probability θ_i.

On the basis of above assumption, we propose an extended model based on the HGDM incorporating the notion of imperfect debugging in the following.

3.1 Mean value function of the HGDM with imperfect debugging

Let $m_0 = m$ be the number of initial faults in a program before the test/debug phase. Suppose there are x_1 new faults discovered by t_1. Since the detected faults are instantly corrected and the fault introduction rate during the application of t_1 is θ_1, there are $\theta_1 x_1$ faults newly introduced during the removal of the detected faults. Therefore, the number of faults including initial faults and the faults newly introduced by t_1 are $m_1 = m_0 + \theta_1 x_1 = m + \theta_1 x_1$. Suppose there are x_2 new faults discovered by t_2. Since the fault introduction rate during the application of t_2 is θ_2, there are $\theta_2 x_2$ faults newly introduced during the removal of the detected faults. Therefore, the number of faults including initial faults and all the faults introduced so far by t_1 and t_2 are $m_2 = m_1 + \theta_2 x_2 = m + \theta_1 x_1 + \theta_2 x_2$.

Let m_i be the number of faults including the initial faults and all the faults already introduced so far by $t_1, t_2, \ldots, t_i$. The HGDM with imperfect debugging can be derived as follows.

$$\begin{aligned} m_0 &= m; \\ m_1 &= m_0 + \theta_1 N_1 = m + \theta_1 N_1; \\ &\cdots\cdots \\ m_n &= m + \sum_{i=1}^{n} \theta_i N_i. \end{aligned} \quad (3)$$

Therefore, Eq.(1) can be modified to be

$$P\left(N_i = x_i \mid m_{i-1}, w_i, C_{i-1}\right) = \frac{\binom{m_{i-1} - C_{i-1}}{x_i}\binom{C_{i-1}}{w_i - x_i}}{\binom{m_{i-1}}{w_i}}, \quad (4)$$

where $\max(0, w_i - C_{i-1}) \leq x_i \leq \min(m_{i-1} - C_{i-1}, w_i)$. Under the assumption that the test is completely random, we have

$$\frac{w_i}{m_{i-1}} = p \ \ \forall i = 1, 2, \ldots, n, \text{ and } 0 \leq p \leq 1. \quad (5)$$

For convenience, let

$$C_i^* = \sum_{k=1}^{i}(1 - \theta_k)N_k, \quad i = 1, 2, \ldots, n. \quad (6)$$

Since $m_{i-1} = m + \sum_{k=1}^{i-1} \theta_k N_k$, we have

$$\begin{aligned} EN_i &= E(E(N_i \mid N_{i-1})) = E[(m_{i-1} - C_{i-1})p] \\ &= p(m - EC_{i-1}^*), \quad i = 1, 2, \ldots, n. \end{aligned} \quad (7)$$

From Eqs.(6) and (7), we have

$$EC_i^* = \{1 - (1 - \theta_i)p\}EC_{i-1}^* + mp(1 - \theta_i).$$

The solution to this difference equation is [10]

$$EC_i^* = m[1 - \prod_{k=1}^{i}\{1 - (1 - \theta_k)p\}], \quad i = 1, 2, \ldots, n. \quad (8)$$

Therefore, EC_i $(i.e., \sum_{k=1}^{i} EN_k)$ for the HGDM with imperfect debugging is

$$\begin{cases} EC_1 = mp, \\ EC_i = mp\left[1 + \sum_{j=2}^{i}\prod_{k=1}^{j-1}\{1 - (1 - \theta_k)p\}\right], i = 2, \ldots, n. \end{cases} \quad (9)$$

3.2 Characteristics of fault introduction rate

The fault introduction rate θ_i is the probability of fault introduction during the application of t_i, and it should satisfy the condition: $0 \leq \theta_i < 1$ for all $i \geq 1$. If θ_i equals zero for all $i \geq 1$, it is equivalent to the perfect debugging case. If θ_i is a constant for all $i \geq 1$, it assumes the probability of introducing a software fault

is a constant. However, the assumption of θ_i being a constant is not realistic. Since the skill to remove software faults will improve with the experience, the effect of learning in the progress of removing faults should be taken into consideration.

Traditionally, the *"decreasing exponential curve"* or the *"decreasing S-shaped curve "* as shown in Figure 1 is commonly used to interpret the human learning process under the imperfect debugging environment. The decreasing exponential learning curve assumes the fault introduction rate decreases faster at the initial application of test instances than at later stages. The decreasing S-shaped learning curve assumes that the fault introduction rate decreases slowly at the beginning stages of testing because the skill to remove faults is not experienced initially; at the subsequent stages, the rate decreases quickly because the skill becomes versed; at the later stages, the rate again decreases slowly.

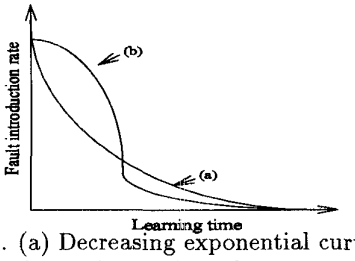

Figure 1. (a) Decreasing exponential curve; (b) decreasing S-shaped curve.

A continuous function $\theta(t)$ which can describe either the decreasing S-shaped curve or the decreasing exponential curve is given by

$$\theta(t) = \frac{1}{1+e^{at+b}}, \quad a > 0, \ t \geq 0. \tag{10}$$

Through simple calculation, the following may be obtained:

(i) If $b<0$, $\theta(t)$ is a decreasing S-shaped curve.

(ii) If $b\geq 0$, $\theta(t)$ is a decreasing exponential curve.

Based on above discussion for the continuous function $\theta(t)$, a discrete function θ_i can be defined as

$$\theta_i = \theta(i) = \frac{1}{1+e^{ai+b}}, \quad i = 1, 2, \ldots, n. \tag{11}$$

Thus, we can use θ_i to interpret a discrete decreasing S-shaped or exponential learning curve.

3.3 Estimation of parameters by least squares method

The parameters of the HGDM with imperfect debugging can be estimated by the least squares method. For the given sample observations, the sum of squares of errors is

$$S(m, p, \theta_i) = \sum_{i=1}^{n} (C_i - EC_i)^2, \tag{12}$$

where EC_i is given by Eq.(9) and $\theta_i = 1/(1+e^{ai+b})$. The IMSL MATH/LIBRARY Subroutine UNLSF [17] is used for the minimization of Eq.(12), and the estimates of these four parameters m, p, a, and b can be obtained.

4 Growth Curve of EC_i and a Measure of Software Reliability

4.1 A finite–fault category

A classification scheme was proposed by Musa and Okumoto [18] for software reliability growth models. The following theorem shows the HGDM with imperfect debugging belongs to the finite–fault category.

Theorem 1. Assuming $\theta_i = 1/(1+e^{ai+b})$ for $i \geq 1$, the upper bound of EC_i is finite.

Proof: Since θ_i is decreasing in i, we have

$$\begin{aligned} \lim_{i\to\infty} EC_i &\leq \lim_{i\to\infty} mp\{1 + \sum_{j=2}^{i}\prod_{k=1}^{j-1}[1-(1-\theta_1)p]\} \\ &= \lim_{i\to\infty} \frac{m}{1-\theta_1}\{1-[1-(1-\theta_1)p]^i\} = \frac{m}{1-\theta_1}. \end{aligned}$$

Moreover, since EC_i is increasing in i, the upper bound of EC_i is finite. □

4.2 Growth curve of EC_i

Since EC_i is increasing in i, we have the following definition [15].

Definition 1. EC_i is an exponential curve if $EC_{i+1} + EC_{i-1} - 2EC_i < 0$ for all $i \geq 1$. □

Theorem 2. Suppose the test is completely random in the test/debug phase, i.e., $p_i = p$ for all $i \geq 1$. The growth curve of EC_i shows an exponential growth curve no matter what θ_i is.

Proof: For all $i \geq 1$, we have

$$\begin{aligned} EC_{i-1} + EC_{i+1} - 2EC_i &= EN_{i+1} - EN_i. \\ &= -mp^2(1-\theta_i)\prod_{k=1}^{i-1}[1-(1-\theta_k)p] < 0. \end{aligned}$$ □

Theorem 2 indicates if p_i is a constant, EC_i shows an exponential curve no matter what θ_i is. In other words, if the growth curve of EC_i is not an exponential curve, p_i is not a constant.

4.3 A measure of software reliability

The term $m_i - C_i$ means the number of remaining faults after the application of t_i. Thereore, a measure of software reliability is defined as

$$R_i = 1 - \frac{m_i - C_i}{m} = \frac{\sum_{j=1}^{i}(1-\theta_j)N_j}{m}, \quad i = 0, 1, 2, \ldots, n.$$

Note that R_i is increasing in i and $0 \leq R_i \leq 1$ for all i. The expected value of R_i is

$$ER_i = E\left[\frac{\sum_{j=1}^{i}(1-\theta_j)N_j}{m}\right] = \frac{EC_i^*}{m}, \qquad (13)$$

where C_i^* is defined in Eq.(6). It is obvious that ER_i is increasing in i, and then we have the following definition [15].

Definition 2.

(i) ER_i is an exponential curve if $ER_{i+1} + ER_{i-1} - 2ER_i < 0$ for all $i \geq 1$;

(ii) ER_i is an S-shaped curve if there exists a finite integer $I > 1$ such that $ER_{i+1} + ER_{i-1} - 2ER_i \geq 0$ for all $i < I$ and $ER_{i+1} + ER_{i-1} - 2ER_i < 0$ for all $i \geq I$. □

From Eq.(13), we have

$$ER_{i+1} + ER_{i-1} - 2ER_i = \frac{EC_{i+1}^* + EC_{i-1}^* - 2EC_i^*}{m}. \qquad (14)$$

Since $\theta_i = 1/(1 + e^{ai+b})$, from Eq.(8), we have

$$EC_i^* = m[1 - \prod_{k=1}^{i}\{1 - p/(1 + e^{-ak-b})\}], \quad i = 1, 2, \ldots, n. \qquad (15)$$

Based on Eqs.(14) and (15), the following theorem of the growth curve of ER_i can be obtained similarly [15].

Theorem 3. Assuming $\theta_i = 1/(1 + e^{ai+b})$, we have:

(i) if $1 \geq p > e^{-(2a+b)(e^a-1)}$, then ER_i is an exponential curve;

(ii) if $0 \leq p \leq e^{-(2a+b)(e^a-1)}$, then ER_i is an S-shaped curve. □

5 Precise Relationship to Goel–Okumoto Model with Imperfect Debugging

In this section, under the assumption of $\theta_i = \theta$ for all $i \geq 1$, the mathematical relationship between the mean value function of the HGDM with imperfect debugging and that of the Goel–Okumoto NHPP Model with imperfect debugging [5] is shown.

The mean value function $D(i)$ of the Goel–Okumoto model with imperfect debugging is [5]:

$$D(i) = \frac{m}{1-\beta}[1 - e^{-(1-\beta)\phi i}], \qquad (16)$$

where m is the number of initial faults in a program, β is the fault introduction rate ($0 \leq \beta < 1$), and ϕ is the fault detection rate ($\phi > 0$).

If θ_i is a constant, from Eq.(9) we have

$$EC_i = \frac{m}{1-\theta}\{1 - e^{i\ln[1-(1-\theta)p]}\}. \qquad (17)$$

Comparing Eq.(16) with Eq.(17), since both θ and β are fault introduction rates, we can let

$$\theta = \beta \text{ and } p = \frac{1}{1-\beta}[1 - e^{-(1-\beta)\phi}]. \qquad (18)$$

Since $0 \leq (1 - e^{-(1-\beta)\phi})/(1-\beta) \leq 1$ if and only if $0 \leq \phi \leq 1$, the relationship between the mean value functions of these two models is given by Eq.(18) when $0 \leq \phi \leq 1$.

6 Numerical Examples and Data Analysis

To validate the proposed model, two software fault data sets are performed. The sum of squares of errors $SSE = \sum_{i=1}^{n}(C_i - \widehat{C}_i)^2$ is adopted as the evaluation criterion. From Eq.(13), two measures related to ER_i, the observed software reliability $R1_i$ and the estimated software reliability $R2_i$, are defined in the following, respectively.

$$R1_0 = 0, \text{ and } R1_i = \frac{\sum_{j=1}^{i}(1-\widehat{\theta}_j)x_j}{\widehat{m}}, \quad i = 1, 2, \ldots, n;$$

$$R2_0 = 0, \text{ and } R2_i = \frac{\sum_{j=1}^{i}(1-\widehat{\theta}_j)\widehat{EN}_j}{\widehat{m}}, \quad i = 1, 2, \ldots, n.$$

6.1 First Data Set

The first test/debug data set is presented in [19]. It is the collection of the cumulative number of discovered faults for the 81 test instances. The cumulative number of discovered faults up to the test instance t_{81} is 460.

The least squares estimates of parameters of the HGDM with imperfect debugging are $\widehat{m} = 366.5$, $\widehat{p} = 0.0295$, $\widehat{a} = 0.2806$, $\widehat{b} = -4.0461$. The sum of squares of errors SSE is 6567.9. Therefore, the number of faults introduced during the observation period as a result of imperfect debugging is approximately 93.

The observed and estimated growth curves of the cumulative number of detected faults are shown in Figure 2. It can be seen that the estimated growth curve fits the data nicely. Since $\widehat{b} = -4.05 < 0$, θ_i is a decreasing S-shaped curve as shown in Figure 3. The curves of $R1_i$ and $R2_i$ are plotted in Figure 4. Obviously, the curve of $R2_i$ is an S-shaped cure. In fact, the curve of $R2_i$ being S-shape can be easily determined by Theorem 3 instead of burdensome plotting.

The least squares estimates of parameters of the HGDM are $\widehat{m} = 545.67$ and $\widehat{p} = 0.0239$, and the sum of squares of errors SSE is 8181.18. Based on SSE, the HGDM with imperfect debugging fits the data more accurately than the HGDM.

6.2 Second Data Set

This data is presented in [20]. It is the collection of the cumulative number of discovered faults for the

25 test instances. The least squares estimates of the parameters of the HGDM with imperfect debugging are $\widehat{m} = 135.97$, $\widehat{p} = 0.1291$, $\widehat{a} = 0.9401$, $\widehat{b} = 11.0243$. The sum of squares of errors SSE is 766.1. Since all the values of $\theta_i, i = 1, 2, \ldots, 25$ tend to zero, the number of faults introduced during the fault removal process is approximately 0.

The observed and estimated growth curves of the cumulative number of detected faults are shown in Figure 5. The estimated growth curve fits the data satisfactorily. Since $\widehat{b} = 11.0243 > 0$, θ_i is a decreasing exponential curve as shown in Figure 6. Moreover, since all the values of θ_i are negligible, the fault removal process can be regarded to be perfect debugging. Accordingly, the proposed model can be reduced to the HGDM. The curves of $R1_i$ and $R2_i$ are plotted in Figure 7. Similarly, the curve of $R2_i$ being exponential shape can be easily determined by Theorem 3.

The least squares estimates of parameters of the HGDM are $\widehat{m} = 135.98$ and $\widehat{p} = 0.129$, and the sum of squares of errors SSE is 766.1. Obviously, the HGDM with imperfect debugging is equivalent to the HGDM for this data set. Thus, we can still apply the HGDM with imperfect debugging even though the fault removal process is perfect debugging.

7 Conclusions

In this paper, we propose an extended model based on the HGDM by elimination of the assumption that the detected faults in a program can be perfectly removed. To make the proposed model more realistic and practical, we consider the situation where there is learning in the fault removal process. The experimental results show the HGDM with imperfect debugging can fit the data sets satisfactorily. Moreover, we can apply the HGDM with imperfect debugging even though the fault removal process is perfect. The growth curve of the cumulative number of discovered faults is investigated. The relationship between the proposed model and the Goel–Okumoto model with imperfect debugging is also discussed.

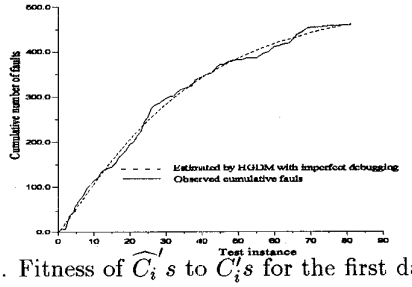

Figure 2. Fitness of $\widehat{C_i}'s$ to $C_i's$ for the first data set.

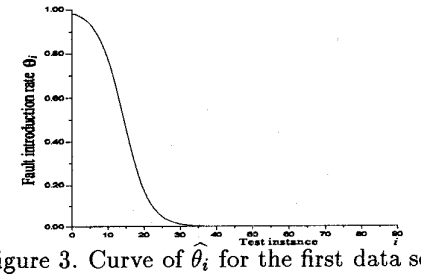

Figure 3. Curve of $\widehat{\theta_i}$ for the first data set.

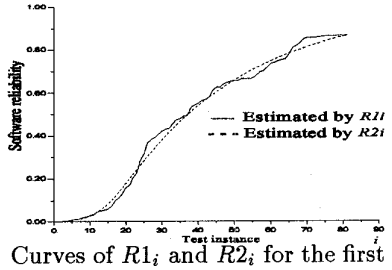

Figure 4. Curves of $R1_i$ and $R2_i$ for the first data set.

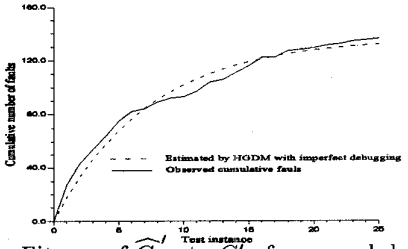

Figure 5. Fitness of $\widehat{C_i}'s$ to $C_i's$ for second data set.

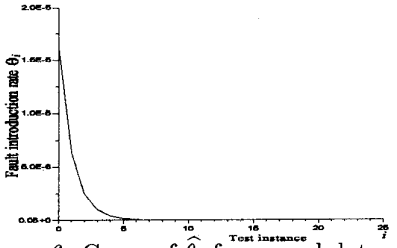

Figure 6. Curve of $\widehat{\theta_i}$ for second data set.

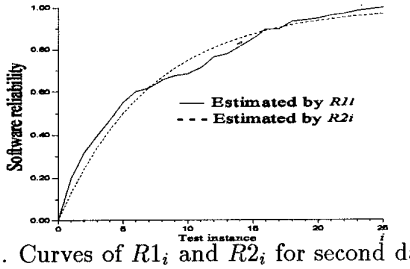

Figure 7. Curves of $R1_i$ and $R2_i$ for second data set.

Acknowledgment. We would like to express our gratitude for the support of the National Science Council, Taiwan, R.O.C., under Grant NSC 84–0408–E002–008. Reviewers' comments are also highly appreciated.

References

[1] C. V. Ramamoorthy and F. B. Bastani, "Software Reliability — Status and Perspectives", *IEEE Trans. Software Engineering*, Vol. 8, No. 4, pp. 354–371, 1982.

[2] B. Littlewood, "Theories of Software Reliability: How Good Are They and How Can They Be Improved?", *IEEE Trans. Software Engineering*, Vol. 6, No. 5, pp. 489–500, 1980.

[3] U. Sumita and J. G. Shanthikumar, "A Software Reliability Model with Multiple–Error Introduction & Removal", *IEEE Trans. Reliability*, Vol. R–35, No. 4, pp. 459–462, 1986.

[4] M. Ohba and X. M. Chou "Does Imperfect Debugging Affect Software Reliability Growth?", *Proc. 11th Int. Conf. Software Engineering*, pp. 237–244, 1989.

[5] P. K. Kapur and R. B. Garg, "Optimum Software Release Policies for Software Reliability Growth Models under Imperfect Debugging", *R.A.I.R.O.*, Vol. 24, No. 3 pp. 295–305, 1990.

[6] K. Tokunoh, S. Yamada, and S. Osaki, "A Markovian Imperfect Debugging Model for Software Reliability Measure", *IEICE Trans. Fundmentals*, Vol. E75-A, No. 11, pp. 1590–1596, 1992.

[7] P. K. Kapur, P. S. Grover and S. Younes, "Modeling an Imperfect Debugging Phenomenon with Testing Effort", *Proc. Int. Symposium on Software Reliability Engineering*, Monterey, California, pp. 178–183, November 1994.

[8] P. Zeephongsekul, G. Xia and S. Kumar, "Software–Reliability Growth Model: Primary–Failures Generate Secondary-Faults Under Imperfect Debugging", *IEEE Trans. Reliability*, Vol. R–43, No. 3, pp. 408–413, 1994.

[9] Y. Tohma, K. Tokunaga, S. Nagase, and Y. Murata, "Structural Approach to the Estimation of the Number of Residual Software Faults Based on the Hyper–Geometric Distribution", *IEEE Trans. Software Engineering*, Vol. 15, No. 3, pp. 345–355, March 1989.

[10] Y. Tohma, R. Jacoby, Y. Murata, and M. Yamamoto, "Hyper–Geometric Distribution Model to Estimate the Number of Residual Software Faults", *Proc. COMPSAC–89*, Orlando, pp. 610–617, September 1989.

[11] Y. Tohma, H. Yamano, M. Ohba, and R. Jacoby, "Parameter Estimation of the Hyper–Geometric Distribution Model for Real Test/Debug Data", *Proc. Int. Symposium on Software Reliability Engineering*, Austin, Texas, May 17–18, pp. 28–34, 1991.

[12] Y. Tohma, H. Yamano, M. Ohba, and R. Jacoby, "The Estimation of Parameters of the Hypergeometric Distribution and Its Application to the Software Reliability Growth Model", *IEEE Trans. Software Engineering*, Vol. SE–17, No. 5, pp. 483–489, May 1991.

[13] R. Jacoby and Y. Tohma, "The Hyper–Geometric Distribution Software Reliability Growth Model (HGDM): Precise Formulation and Applicability", *Proc. COMPSAC90*, Chicago, pp. 13–19, October 1990.

[14] R. Jacoby and Y. Tohma, "Parameter Value Computation by Least Square Method and Evaluation of Software Availability and Reliability at Service–Operation by the Hyper–Geometric Distribution Software Reliability Growth Model (HGDM)", *Proc. 13th Int. Conf. Software Engineering*, pp. 226–237, 1991.

[15] R. H. Hou, S. Y. Kuo, and Y. P. Chang, "Applying Various Learning Curves to Hyper–Geometric Distribution Software Reliability Growth Model", *Proc. Int. Symposium on Software Reliability Engineering*, Monterey, California, pp. 7–16, November 1994.

[16] R. H. Hou, I. Y. Chen, Y. P. Chang, and S. Y. Kuo, " Optimal release policies for hypergeometric distribution software reliability growth model with scheduled delivery time", *The Asia–Pacific Software Engineering Conference*, Tokyo, pp. 445–452, December 1994.

[17] IMSL MATH/LIBRARY, *FORTRAN Subroutines for Mathematical Applications*, 1991.

[18] J. D. Musa, A. Iannino, and K. Okumoto, "Software Reliability — Measurement, Prediction, Application", McGraw–Hill, INc., 1987.

[19] K. Kanoun, M. R. Bastos Martini, and J. Moreira de Souza, "A Method for Software Reliability Analysis and Prediction. Application to the Tropico–R Switching System", Research Report from LAAS–CNRS, France, 1989.

[20] R. Jacoby and Y. Tohma, "Note on the Application of the Hyper–Geometric Distribution to Estimate the Total Number of Faults Initially Resident in a Software Under Test", IECE Technical Report, FTS88–9, pp. 51–58, May 1988.