On The Development of Flexible Discrete SRGM with Two Types of Imperfect Debugging

P. K. Kapur*  S. S. Handa**  Deepak Kumar*  P. C. Jha*
*Department of Operation Research, University of Delhi, Delhi
**SQC & OR Unit, Indian Statistical Institute, Delhi

ABSTRACT
Computer based systems typically consist of hardware and software. Quality hardware can now be produced at a reasonable cost but the same cannot be said about software. Software Reliability Growth Models (SRGM) is used to estimate the reliability of a software product. These SRGM may use execution time i.e. CPU time or calendar time and are known as continuous SRGM. Also SRGM may use the test case as a unit of fault removal period that is countable and known as discrete SRGM. A test case can be a single computer test run executed in an hour, day, week or month even. A large number of models exist in continuous environment while fewer are there in discrete environment due to the difficulties in mathematical complexity involved. But it has been observed that the discrete models fit the actual data better as compared to their continuous counterpart. The debugging process of a fault on observation of a failure caused by it from the software is not always perfect; it is rather imperfect in practice. This imperfect removal may result either in the inability to remove an original fault perfectly or in the introduction of a new fault on removal of an old fault. In this paper we have developed a discrete dynamic combinational model and shown its applicability. The developed model has been validated on real life data sets.

KEYWORDS
Software Reliability Growth Model (SRGM); Non-Homogeneous Poisson Process (NHPP); Imperfect Debugging; Error generation; Fault; Simple Fault; Hard Fault; Complex Fault

1. INTRODUCTION
Proliferation of computer and its increasing application in day-to-day life has made software development a challenging task. Software Company makes effort to provide reliable software at reasonable cost. Software reliability models are used to estimate the reliability of a software product. Reliability is defined as “The ability of a system or component to perform its required functions under stated conditions for a specified period of time.” Models that describe the failure phenomenon and consequent enhancement in reliability due to fault removal are termed as Software Reliability Growth Models (SRGM).

NHPP based SRGM are generally classified into two categories. The first category contains models, which use the execution time (i.e. CPU time) or calendar time. Such models are called continuous time models (Bittanti 1998, Goel and Okumoto 1979, Kapur, Garg and Kumar 1999, Yamada, Ohba and Osaki 1983). The second category contains models, which use the test cases as a unit of fault removal period. Such models are called discrete time models, since the unit of software fault removal period is countable (Kapur, Garg and Kumar [8], Yamada and Osaki [21]). A test case can be a single computer test run executed in an hour, day, week or even month. Therefore, it includes the computer test run and length of time spent to visually inspect the software source code. A large number of models have been developed in the first group while fewer are there in the second group due to the difficulties in terms of mathematical complexity involved. The utility of discrete SRGM cannot be under estimated. As the software failure data sets are discrete, these models many times provide better fit than their continuous time counterparts. Therefore, in spite of difficulties in terms of mathematical complexity involved, discrete models are proposed regularly. Yamada et. al (Yamada and Osaki[21]) and Kapur et. al (Kapur, Garg and Kumar[8,9,11]) contributed towards discrete literature.

Kapur et al [7, 8] proposed an SRGM with three types of fault. For each type, the Fault Removal Rate per remaining faults is assumed to be time independent. The first type is modeled by an Exponential model of Goel and Okumoto [4]. The second type is modeled by Delayed S-shaped model of Yamada et al. [20]. The third type is modeled by three stages Erlang model proposed by Khoshogoftaar [19]. The total removal phenomenon is again modeled by the superposition of the three SRGM [13]. The fault is classified as simple if the time delay between failure observation, fault isolation and fault removal is negligible. If there is time delay, it is classified as a hard fault. If the removal of a fault after its isolation involves an even greater time delay, it is classified as a complex fault. Later they extended their model to cater for more types of faults [8].

Debugging process is usually imperfect because during testing all software faults may not be completely removed or repaired imperfectly as locating and removing a software fault is a difficult task or some new faults might be introduced during removal, which comes into existence after the removal of original faults. In real software development environment, the number of failures observed need not be same as the number of errors removed. If the number of failures observed is more than the number of faults removed then we have the case of imperfect debugging. Due to the complexity of the software system and the incomplete understanding of the software requirements, specifications and structure, the testing team may
not be able to remove the fault perfectly on detection of the failure and the original fault may remain or get replaced by another fault. The first phenomenon is known as imperfect debugging of type-I, the second is called fault generation known as imperfect debugging of type-II.

The concept of imperfect debugging was first introduced by Goel [3]. He introduced the probability of imperfect debugging in Jelinski and Moranda [5] model. Kapur and Garg [7] introduced the imperfect debugging in Goel and Okamoto [4] model. They assumed that the fault removal rate per remaining faults is reduced due to imperfect debugging. Thus the number of failures observed by time infinity is more than the initial fault content. Although these two models describe the imperfect debugging phenomenon of type I yet the software reliability growth curve of these models is always exponential. Obha & Chou [13] introduced the effect of error generation (imperfect debugging type II) into reliability modeling. Later based on their model other researchers in the field of reliability modeling further studied the effect of error generation. All these models have been named as Imperfect Debugging models. This is a common misconception in the study of reliability engineering. Most of the existing research doesn’t clearly distinguish between the two types of imperfect debugging leading to confusion in appropriate insight into the topic as in [16]. Zang et al [23] proposed a testing efficiency model that includes both types of imperfect debugging, modeling it on the number of failures experienced/ removal attempts. However in practice a fault is generated while removing some faults and existence of a generated fault is known only after the removal of original fault. Therefore the fault generation rate is proportional to the rate of fault removals. It may again be noted here that number of failures is not same as the number of removals. Recently, Kapur et al [11] proposed and validated an SRGM elaborating how an SRGM can describe the effect of two types of imperfect debugging approximately. More over Kapur et al [9] has shown the effect of two types of imperfect debugging on release time of software. All SRGMs discussed above are continuous in time. Discrete modeling incorporating two types of imperfect debugging has been done by Kapur et al. [11]. In this paper we propose and validate a new discrete SRGMs incorporating two types of imperfect debugging considering three level complexities of faults.

The paper is organized as follows: Section 2.1 describes the Non Homogenous Poisson Process. Section 2.2 describes the assumptions and notations for the proposed model. Section 2.3 discussed discrete SRGM considering two types of imperfect debugging. Section 2.4 discussed the model development considering faults of different severity/complexity with imperfect debugging and fault generation. The proposed models are validated using two data sets cited in literature in section 3. For estimation of parameters of the proposed model, SPSS is used. SPSS is a Statistical package for Social Sciences. Finally conclusions are drawn in section 4.

2. MODELLING SOFTWARE RELIABILITY GROWTH MODEL

2.1 DISCRETE SRGM BASED ON NHPP – A GENERAL DESCRIPTION

During the software testing phase a software system is executed with samples of test cases to remove software faults, which cause software failures. A discrete counting process \(N(n); n \geq 0\) is said to be an NHPP with mean value function \(m(n)\), if it satisfies the following conditions:

1. There are no failures experienced at \(n=0\), i.e., \(N(n=0)=0\).

2. The counting process has independent increments, that is, for any collection of the numbers of test cases \(n_1,n_2,\ldots,n_k\) where \((0=0n_1<n_2<\ldots<n_k)\), the \(k\) random variables \(N(n_1)-N(n_1),\ldots,N(n_k)-N(n_{k-1})\) are statistically independent.

3. For any numbers of test cases \(n_i\) and \(n_j\) where \((0\leq n_i \leq n_j)\), we have

\[
\Pr(N(n_i)-N(n_j)=x) = \frac{1}{x!} \left(\frac{m(n)}{m(n)}\right)^x \exp\left(-\frac{m(n)}{m(n)}\right) x \geq 0
\]  

4. The mean value function \(m(n)\), which is bounded above and is non-decreasing in \(n\) represents the expected cumulative number of faults detected by \(n\) test cases. Then the NHPP model with \(m(n)\) is

\[
\Pr(N(n)=x) = \frac{1}{x!} \left(\frac{m(n)}{m(n)}\right)^x \exp\left(-\frac{m(n)}{m(n)}\right) x \geq 0
\]  

A useful software reliability growth index, the fault detection rate per fault (per test case) after the \(n^{th}\) test case is given by

\[
q(n) = \frac{m(n+1)-m(n)-\tilde{h}(n)}{m(n)} \quad n \geq 0
\]  

Where, \(m(\infty)\) represents the expected number of faults to be eventually detected.

Let \(\tilde{N}(n)\) denotes the number of faults remaining in the system after the \(n^{th}\) test case is given as

\[
\tilde{N}(n) = N(\infty) - N(n)
\]  

The expected value of \(\tilde{N}(n)\) is given by:

\[
h(n) = m(\infty) - m(n)
\]
Suppose that \( n_d \) faults have been detected by \( n \) test cases. The conditional distribution of \( N(n) \), given that \( N(n) = n_d \), is given by

\[
\Pr\{N(n) = y \mid N(n) = n_d \} = \frac{\mathcal{L}(n)}{y!} \exp \left[ - \mathcal{L}(n) \right]
\]

which means a Poisson distribution with mean \( E(n) \), independent of \( n_d \).

Now, the probability of no faults detected between the \( n^th \) and the \((n+h)^th\) test cases, given that \( n_d \) faults have been detected by \( n \) test cases, is given by

\[
R(h \mid n) = \exp \left[ - \mathcal{L}(n+h) + m(n) \right] \quad n, h \geq 0
\]

The above conditional reliability function \( R(h|n) \) is called a software reliability function based upon a NHPP for a Discrete Time Model and is independent of \( n_d \).

2.2 MODEL ASSUMPTIONS AND NOTATIONS

SRGM discussed in the section 2.3 considers two types of imperfect debugging and proposed in the section 2.4 considers that software system consists of faults of different severity/complextiy and takes into account the time lag between the failure and fault isolation / removal processes. The general assumptions and notations are

ASSUMPTIONS

1. Failure observation / fault removal phenomenon is modeled by NHPP.
2. Software is subject to failures during execution caused by faults remaining in the software.
3. Software system consists of a finite number of simple faults, hard faults and complex faults.
4. Each time a failure is observed, an immediate effort takes place to know the cause of the failure in order to remove it. The time delay between the failure observation and its subsequent removal is assumed to represent the severity/complexity of fault. The more severe the fault, more the time delay.
5. Failure rate of the software is equally affected by faults remaining in the software.
6. During the fault isolation / removal, new fault can be introduced into the system.
7. The debugging process is imperfect in two ways- incomplete removals and error generation.

NOTATIONS

\( a \) Total fault content
\( a_i \) Initial fault content of type \( i \) faults, \( i = 1,2,3 \)
\( b \) Proportionality constant failure rate / fault isolation rate per fault of faults.
\( b_i \) Proportionality constant of failure/fault isolation rate per remaining fault type \( i, i = 1,2,3 \)

\( b(n) \) Fault Removal Rate per fault of faults by \( n \) test cases.
\( b_i(t) \) Fault Removal Rate per remaining fault of type \( i, i = 1,2 \).
\( p \) The probability of fault removal on a failure (i.e., the probability of perfect debugging) of faults.
\( p_i \) The probability of perfect debugging of fault type \( i, i = 1,2,3 \)
\( m_i(n) \) Mean number of removals caused by \( n \) test cases.
\( m_{i\alpha}(n) \) Mean number of type \( i \) faults removed by \( n \) test cases, \( i = 1,2,3 \)
\( m_i(n) \) Mean number of type \( i \) faults removed by \( n \) test cases, \( i = 1,2,3 \)
\( m_{i\beta}(n) \) Mean number of type \( i \) faults removed by \( n \) test cases, \( i = 1,2,3 \)
\( m_{3\alpha}(n) \) Mean number of removals caused from complex faults by \( n \) test cases.
\( m_{3\beta}(n) \) Mean number of removals caused from complex faults by \( n \) test cases.
\( \alpha \) The rate at which the faults may be introduced during the debugging process per detected faults.
\( \alpha_i \) The rate at which the faults may be introduced during the debugging process per detected faults of type \( i, i = 1,2,3 \)
\( \beta, \beta_i \) Constant parameters in the logistic learning function, \( i = 1,2,3 \)
\( \delta \) Constant time difference interval
\( r_i \) Proportion of fault type \( i, i = 1,2,3 \)

2.3 LITERATURE REVIEW

Discrete SRGM were earlier developed by Yamada & Osaki[21] and Kapur et. al[8,9,11]. However they did not consider the effect of imperfect debugging and fault generation. We review below an SRGM with two types of imperfect debugging due to Kapur et al[11].

A DISCRETE NHPP MODEL FOR SOFTWARE RELIABILITY GROWTH WITH IMPERFECT FAULT DEBUGGING & FAULT GENERATION.

The model incorporates two cases of imperfect debugging during software fault removal phenomenon with logistic fault removal rate. The expected cumulative number of faults removed between the \( n^{th} \) and \((n+1)^{th}\) test cases is proportional to the number of faults remaining after the execution nth test run. Following the above assumptions the general difference equation describing the removal phenomenon can be written as:

\[
\frac{m_{f}(n+1) - m_{f}(n)}{\delta} = b(n+1) \left[ a(n) - m_f(n) \right]
\]

where, \( a(n) = a(1 + \alpha \delta)^n \), and

\[
b(n+1) = \frac{b p}{1 + \beta(1-bp\delta)^{n+1}}
\]
Solving the above difference equation (2.8) using probability generating function (PGF) under initial condition \( m_r(n=0)=0 \), we get mean value function for the removal phenomenon

\[
m_r(n) = \frac{abp\delta}{1+\beta(1-bp\delta)^{n+1}} \left[ (1+\alpha\delta)^n - (1-bp\delta)^n \right] \]

\[
\frac{m_r(n+1)-m_r(n)}{1+\beta(1-bp\delta)^{n+1}} = \frac{abp\delta}{(\alpha\delta+bp\delta)} \left[ \frac{\alpha e^{\alpha t} - e^{\beta t}}{\alpha + \beta} \right] \]

\[
\text{... (2.10)}
\]

Note that in this case exact solution for \( m_r(n) \) is obtainable. While same may not be true for other forms of \( a(n) \).

**DERIVATION OF EQUIVALENT CONTINUOUS MODEL**

We can derive the equivalent continuous SRGM of the above discrete mean value function as follow;

Let us define \( t = n\delta \).

We know that

\[
Lt \left( \frac{1}{x} \right)_{x \to 0} = e
\]

As \( \delta \to 0 \)

\[
Lt 1-bp\delta^n = 1-bp\delta^{\frac{t}{\delta}} \to e^{-bpt} \]

\[
\delta \to 0
\]

\[
a(n) = a + \alpha \delta \to ae^{\alpha t} = a(t)
\]

\[
\text{and } b(n+1) = \frac{bp}{1+\beta(1-bp\delta)^{n+1}} \to \frac{bp}{(1+\beta e^{-bpt})} = b(t)
\]

As \( \delta \to 0 \)

\[
\frac{abp\delta}{1+\beta(1-bp\delta)^{n+1}} \left[ (1+\alpha\delta)^n - (1-bp\delta)^n \right] \to \frac{abp}{(\alpha + \beta)} \left[ \frac{\alpha e^{\alpha t} - e^{\beta t}}{\alpha + \beta} \right]
\]

\[
\text{... (2.11)}
\]

\[
\frac{dm_r(t)}{dt} = b(t)(a(t) - m_r(t))
\]

\[
\text{... (2.15)}
\]

The corresponding differential equation is given by

\[
\frac{dm_r(t)}{dt} = b(t)(a(t) - m_r(t))
\]

\[
\text{... (2.16)}
\]

\[
\text{where, } a_1(n) = a_1 + \alpha_1 m_{r}(n) \text{ and } b_1(n+1) = b_1
\]

\[
\text{... (2.17)}
\]

Solving the above differential equation with initial condition \( m_{r}(n=0) = 0 \), we get mean value function as equation (2.14).

**A. MODELLING SIMPLE FAULTS**

It is assumed that simple faults which can be removed instantly as soon as they are observed. Hence fault removal of simple faults can be modeled as one-stage process

\[
m_{r}(n+1)-m_{r}(n) = p_{1}b_{1}(n+1) \]

\[
a_{1}(n)-m_{r}(n)
\]

\[
\text{... (2.18)}
\]

It may be noted here that, \( m_{r}(\infty) = (a_1 l(1-\alpha_1)) > a_1 \).

**B. MODELLING HARD FAULTS**

It is assumed that the hard faults consume more testing effort when compared with simple faults. This means that the testing team will have to spend more time to analyze the cause of the failure and therefore requires greater effort to remove them. Hence the removal process for such faults is modeled as a two-stage process and the differential equation for the failure and removal phenomenon under the assumptions 1-5 are given as

\[
\frac{d}{dt}m_{2}(t) = b_{2}[a_{2} - m_{2}(t)]
\]

\[
\text{... (2.19)}
\]

\[
\frac{d}{dt}m_{2r}(t) = b_{2}[m_{2}(t) - m_{2r}(t)]
\]

\[
\text{... (2.20)}
\]

\[
b_{2}(t) = \frac{b_{2}}{1+\beta_{2}e^{-\beta_{2}t}}
\]

\[
\text{... (2.21)}
\]

The first stage of the two-stage process model is given by the equation (2.19). This stage describes the failure observation process. The second stage of the two-stage process given by equation (2.20) describes the delayed fault removal process. During this stage the fault removal rate is assumed to be time dependent considering the learning effect of the debugging team. With each fault removal insight is gained into the nature of faults present and function described in equation (2.21) called logistic function can account for that. Note that \( b(t) \to b_2 \) as \( t \to \infty \).

Solving the above system of differential equations with \( m_{2}(t = 0) = 0 \) and \( m_{2r}(t = 0) = 0 \), the continuous SRGM is given as proposed by Kapur et. al [9] and Kumar et. al [14] with two types of imperfect debugging and correspondingly obtain their discrete versions.

**2.4 PROPOSED MODEL**

**MODELLING THE FAULTS OF DIFFERENT SEVERITY/ COMPLEXITY**

In this paper we propose a discrete SRGM incorporating the two types of imperfect debugging as well as considering complexity of faults. First we discuss the continuous SRGM
$m_{2r}(t) = a_2 \left\{ \frac{1 - 1 + b_2 t}{1 + \beta_2 e^{-b_2 t}} \right\}$ \hspace{1cm} \ldots (2.22)

The removal phenomenon of the model can be derived in one stage directly assuming fault detection rate per remaining fault equals to $b_2 \left[ \frac{1}{1 + \beta_2 e^{-b_2 t}} - \frac{1}{1 + \beta_2 + b_2 t} \right]$ as follows

$$\frac{dm_{2r}(t)}{dt} = b_2 \left[ \frac{1}{1 + \beta_2 e^{-b_2 t}} - \frac{1}{1 + \beta_2 + b_2 t} \right] (a_2 - m_{2r}(t))$$ \hspace{1cm} \ldots (2.23)

Note here also $b_2 \left[ \frac{1}{1 + \beta_2 e^{-b_2 t}} - \frac{1}{1 + \beta_2 + b_2 t} \right] \rightarrow b_2$ as $t \rightarrow \infty$.

Equivalently the differential equation describing the removal process under two types of imperfect debugging can be directly written as

$$\frac{dm_{2r}(t)}{dt} = p_2 \hat{b}(t) [a(t) - m_{2r}(t)]$$ \hspace{1cm} \ldots (2.24)

where, $\hat{b}(t) = b_2 \left[ \frac{1}{1 + \beta_2 e^{-b_2 t}} - \frac{1}{1 + \beta_2 + b_2 t} \right]$, and

$$a(t) = a_2 + \alpha_2 m_{2r}(t)$$

Solving the differential equation (2.24) under initial condition $m_{2r}(t=0)=0$, we get mean value function as

$$m_{2r}(t) = \frac{a_2}{1 - \alpha_2} \left[ 1 - \left( \frac{1 + \beta_2 + b_2 t}{1 + \beta_2 e^{-b_2 t}} \right)^{p_2(1-\alpha_2)} \right]$$ \hspace{1cm} \ldots (2.25)

The discrete version of two-stage process continuous time model given by equation (2.22) for hard faults is given by the following difference equations

$$m_{2r}(n+1) - m_{2r}(n) = b_2 \cdot a_2 - m_{2r}(n)$$ \hspace{1cm} \ldots (2.26)

$$m_{2r}(n+1) - m_{2r}(n) = b_2(n+1) \cdot m_{2r}(n+1) - m_{2r}(n)$$ \hspace{1cm} \ldots (2.27)

where, $b_2(n+1) = \frac{b_2}{1 + \beta_2 + b_2 n}$ \hspace{1cm} \ldots (2.28)

Solving the above system of difference equations using the probability generating function (PGF) with the initial conditions $m_{2f}(n = 0) = 0$ and $m_{2r}(n = 0) = 0$, we get

$$m_{2r}(n) = a_2 \left\{ \frac{1 - 1 + b_2 n}{1 + \beta_2 + b_2 n} \right\}$$ \hspace{1cm} \ldots (2.29)

The removal process of the above model can be derived in one stage directly assuming fault detection rate per remaining fault

$$\hat{b}_2(n+1) = \frac{b_2(1 + \beta_2 + b_2(n+1)) - b_2(1 + \beta_2 + b_2 n^{n+1})}{(1 + \beta_2 + b_2 n^{n+1})(1 + \beta_2 + b_2 n)}$$ \hspace{1cm} \ldots (2.30)

The removal process of discrete SRGM for hard faults with two types of imperfect debugging can be obtained by solving the difference equation

$$m_{2r}(n+1) - m_{2r}(n) = p_2 \hat{b}_2(n+1) \cdot a_2(n) - m_{2r}(n)$$ \hspace{1cm} \ldots (2.31)

where, $\alpha_2(n) = a_2 + \alpha_2 m_{2r}(n)$ and

$$\hat{b}_2(n+1) = \frac{b_2(1 + \beta_2 + b_2(n+1)) - b_2(1 + \beta_2 + b_2 n^{n+1})}{(1 + \beta_2 + b_2 n^{n+1})(1 + \beta_2 + b_2 n)}$$

The exact solution of (2.31) is not obtainable and therefore we propose an approximate solution using the continuous SRGM given by (2.25) and the following relations in continuous and discrete time space.

$$e^{-b_2 t} \approx (1 - b_2)^n \quad \text{and} \quad (1 + x)^n \approx 1 + nx$$

Hence the corresponding approximate discrete mean value function can be written as

$$m_{2r}(n) = \frac{a_2}{1 - \alpha_2} \left( \frac{1 + p_2(1-\alpha_2)(\beta_2 + b_2 n)}{1 + \beta_2 p_2(1-\alpha_2)(1-b_2)} \right)$$ \hspace{1cm} \ldots (2.32)

It can be noted here that, $m_{2r}(x)(1-\alpha_4) > a_2$.

C. MODELLING THE COMPLEX FAULTS

If removal of a fault after its detection involves even a greater time delay, it is classified as complex fault. These faults can require more effort for removal after isolation. Hence they be modeled as a three stage process as follows

$$\frac{dm_{3f}(t)}{dt} = b_3 \left( a_3 - m_{3f}(t) \right)$$ \hspace{1cm} \ldots (2.33)
\[
\frac{dm_{3u}(t)}{dt} = b_3 (m_{3f} \ t - m_{3u} \ t ) \\
\frac{dm_{3u}(t)}{dt} = b_3 (m_{3u} \ t - m_{3r} \ t )
\]

where \( b(t) = \frac{b_3}{1 + \beta_3 e^{-b_3 t}} \)

The first stage of the three-stage process is given by the equation (2.33). This stage describes the failure observation process. The second stage given by equation (2.34) describes the fault isolation process. The third stage given by equation (2.35) describes the fault removal process. During this stage the fault removal rate is assumed to be logistic learning function as before and is used to represent the knowledge gained by the removal team. Solving the above system of differential equations with the initial conditions \( m_{3f}(t=0) = 0, m_{3u}(t=0) = 0 \) and \( m_{3r}(t=0) = 0 \), we get

\[
m_{3r}(t) = a_3 \left\{ 1 - \frac{1 + b_3 t + b_3^2 t^2 / 2}{1 + \beta_3 e^{-b_3 t}} \right\}
\]

(2.36)

The removal phenomenon of the above model can be derived in one stage directly assuming fault removal per remaining fault as given by (2.37)

\[
b_3 \left[ \frac{1}{1 + \beta_3 e^{-b_3 t}} - \frac{(1 + b_3 t)}{1 + \beta_3 + b_3 t + (b_3^2 t^2 / 2)} \right] \]

(2.37)

It is observed that

\[
b_3 \left[ \frac{1}{1 + \beta_3 e^{-b_3 t}} - \frac{(1 + b_3 t)}{1 + \beta_3 + b_3 t + (b_3^2 t^2 / 2)} \right] \rightarrow b_3 \text{ as } t \rightarrow \infty.
\]

Equivalently the differential equation describing the removal process under two types of imperfect debugging can be directly written as

\[
\frac{dm_{3r}(t)}{dt} = p_3 b(t) [a(t) - m_{3r}(t)]
\]

(2.39)

where,

\[
b(t) = b_3 \left[ \frac{1}{1 + \beta_3 e^{-b_3 t}} - \frac{(1 + b_3 t)}{1 + \beta_3 + b_3 t + (b_3^2 t^2 / 2)} \right], \text{ and}
\]

\[
a(t) = a_3 + \alpha_3 \ m_{3r}(t)
\]

Solving the differential equation (2.39) under initial condition \( m_{3r}(t=0)=0 \), we get mean value function as

\[
m_{3r}(t) = a_3 \left[ 1 - \frac{\left( 1 + \beta_3 + b_3 t + (b_3^2 t^2 / 2) \right)e^{-b_3 t}}{1 + \beta_3 e^{-b_3 t}} \right]^{p_3(1-\alpha_3)}
\]

(2.40)

The discrete version of three-stage process continuous time model given by equation (2.36) for complex faults is given by the following difference equations

\[
m_{3f}(n+1) - m_{3f}(n) = b_3 \ a_3 - m_{3f}(n)
\]

(2.41)

\[
m_{3u}(n+1) - m_{3u}(n) = b_3 \ m_{3f}(n+1) - m_{3u}(n)
\]

(2.42)

\[
m_{3r}(n+1) - m_{3r}(n) = b_3 \ m_{3u}(n+1) - m_{3r}(n)
\]

(2.43)

where \( b_3 (n+1) = \frac{b_3}{1 + \beta_3 (1-b_3)^{n+1}} \)

Solving the above system of difference equations using the probability generating function (PGF) with the initial conditions \( m_{3f}(n = 0) = 0, m_{3u}(n = 0) = 0 \) and \( m_{3r}(n = 0) = 0 \), we get

\[
m_{3r}(n) = a_3 \left\{ 1 - \left( 1 + b_3 n + \frac{b_3^2 n + 1}{2} \right) 1 - b_3^n \right\}
\]

(2.44)

The above model can be derived in one stage directly assuming fault detection rate per remaining fault given by equation (2.45) as follows

\[
b_3 (1 + \beta_3 + b_3 (n+1) + b_3^2 (n+1)(n+2) )
\]

\[
\tilde{b}_3 (n+1) = \frac{b_3 (1 + \beta_3 (1-b_3)^{n+1}) (1 + b_3 (n+1))}{(1 + \beta_3 (1-b_3)^{n+1}) (1 + b_3 + b_3 n + b_3^2 n(n+1) / 2)}
\]

(2.45)
On The Development of Flexible Discrete SRGM with Two Types of Imperfect Debugging

\[ m_{3r}(n+1) - m_{3r}(n) = b_3(n+1) - m_{3r}(n) \quad \ldots (2.46) \]

The discrete SRGM for complex faults with two types of imperfect debugging can be obtained by solving the difference equation

\[ m_{3r}(n+1) - m_{3r}(n) = p_3b_3(n+1) - m_{3r}(n) \quad \ldots (2.47) \]

where \( a_3(n) = a_3 + \alpha_3 m_{3r}(n) \) and

\[ b_3(n+1) = \frac{-b_3(1 + \beta_3 b_3(n+1)^{n+1})(1 + b_3(n+1))}{(1 + \beta_3 b_3(n+1)^{n+1})(1 + b_3(n+1)) + \frac{b_3^2(n(n+1))}{2}} \quad \ldots (2.48) \]

The exact solution of (2.47) is not obtainable and therefore we propose an approximate solution as in the case of hard faults, given by

\[ m_{3r}(n) = \frac{a_3}{(1 - a_3)} \left( \frac{1 + \frac{b_3^2(n(n+1))}{2}}{1 + \frac{b_3^2(n(n+1))}{2}} \right)^n \quad \ldots (2.49) \]

It can be noted here that, \( m_{3r}(\infty) = (a_3/(1 - a_3)) > a_3 \).

**D. MODELLING TOTAL FAULT REMOVAL PHENOMENON**

The proposed model is the sum of mean value function of simple, hard and complex faults. Equations (2.18), (2.32) and (2.49) are mean value functions of respective faults. Thus, the mean value function of superimposed NHPP is:

\[ m_i(n) = m_{si}(n) + m_{hi}(n) + m_{ci}(n) \]

Substituting the mean value function of \( m_{si}(n) \), \( m_{hi}(n) \) and \( m_{ci}(n) \). We get

\[ m_i(n) = \frac{a_i}{(1 - a_i)} \left( 1 - (1 - p_i(1 - \alpha_i) b_i)^n \right) \]

\[ + \frac{a_2}{(1 - a_2)} \left( 1 - \frac{1 + p_2(1 - \alpha_2)(\beta_2 + b_2)^n}{1 + \beta_2 p_2(1 - \alpha_2)(1 - b_2)^n} \right) \]

\[ + \frac{a_3}{(1 - a_3)} \left( 1 - \frac{1 + p_3(1 - \alpha_3)(\beta_3 + \delta b_3^n + \frac{b_3^2(n(n+1))}{2})}{1 + \beta_3 p_3(1 - \alpha_3)(1 - b_3)^n} \right) \quad \ldots (2.50) \]

where \( a_1 = a_{r_1}, a_2 = a_{r_2}, a_3 = a_{r_3} \) and \( a_1 + a_2 + a_3 = a \)

3. **PARAMETER ESTIMATION AND COMPARISON**

The success of Software Reliability Growth model depends heavily upon quality of failure data collected. The parameters of the SRGM are estimated based upon these data. Method of least squares or maximum likelihood has been suggested and widely used for estimation of parameters of an SRGM. We use SPSS (Statistical Package for Social Sciences) to find solution of nonlinear models.

**3.1 GOODNESS OF FIT CRITERIA**

1. **The Mean Square Fitting Error (MSE):**

The model under comparison is used to simulate the fault data, the difference between the expected values, \( \hat{y}(t_i) \) and the observed data \( y_i \) is measured by MSE as follows.

\[ MSE = \frac{1}{k} \sum_{i=1}^{k} (\hat{y}(t_i) - y_i)^2 \quad \ldots (3.1) \]

Where \( k \) is the number of observations. The lower MSE indicates less fitting error, thus better goodness of fit.

2. **Coefficient of Multiple Determination (R^2):**

We define this coefficient as the ratio of the sum of squares resulting from the trend model to that from constant model subtracted from 1.

\[ R^2 = 1 - \frac{\text{residual SS}}{\text{corrected SS}} \quad \ldots (3.2) \]

R^2 measures the percentage of the total variation about the mean accounted for the fitted curve. It ranges in value from 0 to 1. Small values indicate that the model does not fit the data well. The larger R^2, the better the model explains the variation in the data.

3. **Prediction Error (PE):**

The difference between the observation and prediction of number of failures at any instant of time \( i \) is known as PE_i. Lower the value of Prediction Error better is the goodness of fit. The average of PEs is known as bias. Lower the value of Bias better is the goodness of fit.

4. **Variation:**

The standard deviation of PE is known as variation.

\[ \text{Variation} = \sqrt{\frac{1}{N-1} \sum_i (PE_i - \text{Bias})^2} \quad \ldots (3.3) \]

Lower the value of Variation better is the goodness of fit.

5. **Root Mean Square Prediction Error:**

It is a measure of closeness with which a model predicts the observation.

\[ RMSPE = \sqrt{\text{Bias}^2 + \text{Variation}^2} \quad \ldots (3.4) \]

Lower the value of Root Mean Square Prediction Error better is the goodness of fit.
3.2 MODEL VALIDATION

The model discussed and proposed in this paper is validated and compared on two real life data sets cited in literature [15,18]. The success of Software Reliability Growth model depends heavily upon quality of data collected. The parameters of the SRGM are estimated based upon these data.

Data set I: This data set is cited from Mishra(1983)[15]. The failure data set consists of three types of errors: critical, major, and minor. It was tested for 38 weeks in which total 231 faults were detected. Statistical parameters of the proposed model are estimated. The estimated values of parameters are given in table 1. Goodness of fit measures are given in table 2 which shows better fit as compared to Kapur et al [11] model. The Fitting of the model is illustrated graphically in figure 1.

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>( a )</td>
<td>397.205</td>
</tr>
<tr>
<td>( b(b_1) )</td>
<td>0.049</td>
</tr>
<tr>
<td>( b_2 )</td>
<td>-</td>
</tr>
<tr>
<td>( b_3 )</td>
<td>-</td>
</tr>
<tr>
<td>( r_1 )</td>
<td>-</td>
</tr>
<tr>
<td>( r_2 )</td>
<td>-</td>
</tr>
<tr>
<td>( r_3 )</td>
<td>-</td>
</tr>
<tr>
<td>( p(p_1) )</td>
<td>0.859</td>
</tr>
<tr>
<td>( p_2 )</td>
<td>-</td>
</tr>
<tr>
<td>( p_3 )</td>
<td>-</td>
</tr>
<tr>
<td>( \alpha(\alpha_1) )</td>
<td>0.010</td>
</tr>
<tr>
<td>( \alpha_2 )</td>
<td>-</td>
</tr>
<tr>
<td>( \alpha_3 )</td>
<td>-</td>
</tr>
<tr>
<td>( \beta(\beta_2) )</td>
<td>1</td>
</tr>
<tr>
<td>( \beta_3 )</td>
<td>-</td>
</tr>
</tbody>
</table>

Data set II: The data was obtained from one of four major releases of software products at Tandem Computers (Woods 1996)[18]. It was tested for 20 weeks. 100 faults were removed during testing. Statistical parameters of the proposed model are estimated. The estimated values of parameters are given in table 3. Goodness of fit measures are given in table 4 which shows better fit as compared to Kapur et al [11] model. The Fitting of the model is illustrated graphically in figure 2.

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>( a )</td>
<td>116.73</td>
</tr>
<tr>
<td>( b(b_1) )</td>
<td>0.162</td>
</tr>
<tr>
<td>( b_2 )</td>
<td>-</td>
</tr>
<tr>
<td>( b_3 )</td>
<td>-</td>
</tr>
<tr>
<td>( r_1 )</td>
<td>-</td>
</tr>
<tr>
<td>( r_2 )</td>
<td>-</td>
</tr>
<tr>
<td>( r_3 )</td>
<td>-</td>
</tr>
<tr>
<td>( p(p_1) )</td>
<td>0.976</td>
</tr>
<tr>
<td>( p_2 )</td>
<td>-</td>
</tr>
<tr>
<td>( p_3 )</td>
<td>-</td>
</tr>
<tr>
<td>( \alpha(\alpha_1) )</td>
<td>0.010</td>
</tr>
<tr>
<td>( \alpha_2 )</td>
<td>-</td>
</tr>
<tr>
<td>( \alpha_3 )</td>
<td>-</td>
</tr>
<tr>
<td>( \beta(\beta_2) )</td>
<td>1.24</td>
</tr>
<tr>
<td>( \beta_3 )</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 2

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>( R^2 )</td>
<td>0.9939</td>
</tr>
<tr>
<td>SSE</td>
<td>858.22</td>
</tr>
<tr>
<td>MSE</td>
<td>22.58</td>
</tr>
<tr>
<td>BIAS</td>
<td>-1.058</td>
</tr>
<tr>
<td>Variation</td>
<td>4.695</td>
</tr>
<tr>
<td>RMSPE</td>
<td>4.813</td>
</tr>
</tbody>
</table>

Table 4

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>( R^2 )</td>
<td>0.9895</td>
</tr>
<tr>
<td>SSE</td>
<td>171.19</td>
</tr>
<tr>
<td>MSE</td>
<td>8.56</td>
</tr>
<tr>
<td>BIAS</td>
<td>-0.4055</td>
</tr>
<tr>
<td>Variation</td>
<td>2.973</td>
</tr>
<tr>
<td>RMSPE</td>
<td>3.000</td>
</tr>
</tbody>
</table>
On The Development of Flexible Discrete SRGM with Two Types of Imperfect Debugging

4. CONCLUSION

In this paper, a discrete SRGM considering faults of different severity/complexity with two types of imperfect debugging has been proposed. The goodness of the fit analysis has been done on two real software failure/removal datasets. The results obtained show better fit and wider applicability of the model to different types of datasets. From the numerical illustrations, we see that the Proposed SRGM provides improved results. In future we would also like to incorporate the concept of change point and testing effort functions in the proposed model. In this paper we have taken constant proportion of the error generation point and testing effort functions in the proposed model. In the future we would also like to incorporate the concept of change point and testing effort functions in the proposed model.

REFERENCES


[19] Woodcock TG and Khoshgoftaar TM; Software Reliability Model Selection: A case study,


