Fault dependency SRGM with testing effort using learning function

Asit Dwivedi*
Mewar University,
Gangrar, Chittorgarh,
Rajasthan, 312901, India
Email: asitdwivedi@rediffmail.com
*Corresponding author

Deepak Kumar
Amity University,
Uttar Pradesh,
Noida, 201301, India
Email: deepakgupta_du@rediffmail.com

Abstract: Software engineering is the procedural application with integration of various techniques, methods and tools, which help in the production of high quality software system. SRGM are mathematical models that observe past failure history and give an estimate of the future failure behaviour. The rapidity of testing of approximately all the software development is governed by the testing effort. The testing effort primarily includes manpower and computer time. Faults remaining in software system are mainly of two types: Firstly, mutually independent faults that are isolated and removed directly; secondly communally dependent faults that are isolated only by the removal of leading faults. SRGM based on fault dependency with the integration of testing efforts and learning is proposed by the authors in this paper. Furthermore, the proposed model is compared with existing models on real life data sets and it is numerically illustrated.

Keywords: software engineering; software reliability growth model; SRGM; testing effort; fault dependency; learning function; mean value function; MVF; non-homogeneous poisson process; NHPP; software management concept.


Biographical notes: Asit Dwivedi is an Assistant Professor at the Amity University Uttar Pradesh, Noida, India and he is pursuing his PhD from the Mewar University, Gangrar, Chittorgarh, Rajasthan, India. He has published/presented more than ten research papers in various international and national journals/conferences.

Deepak Kumar is an Associate Professor at Amity University Uttar Pradesh, Noida, India. He is an MTech from BITS, Mesra, India and PhD from Delhi University, Delhi, India. He has delivered various talks/guest lectures in University of Maryland, USA, Fayetteville State University (A Constituent Institution of the University of North Carolina), USA, San Jose State
1 Introduction

In the modernised era of today’s world it is not possible for human beings to do any work without the aid of software controlled computers, but the reliability of software is important both for its user as well as the computer system. In the layman’s definition computer can be defined as the device that can be programmed not only to solve arithmetical but logical operation as well. The wide use of the software based system across the various domains of life such as hospitals, in means of transport such as airlines and railways have made it simpler of human generation to carry out the task, but what if the software of this system fails. The failures of these types of software are not only in terms of the money and resources but human loss as well that is why it is necessary that the software system that the users are operating must be reliable (Kumar and Dwivedi, 2015). Reliability of the software can be described as the probability of the software that it will work in proper manner in a specific amount of running time in a given environment (Khatri et al., 2012).

Modern advances in information technologies have made the real time and embedded systems indispensable in every aspect of our life. More and more critical and complex systems like medical transplant equipment, banking system, air traffic control system are being developed every day. Reliable functioning of these systems is of utmost importance to the millions of users using these systems every day. Software development follows a process when developing a software product. It is often referred to as software development life cycle (SDLC). SDLC have many phases (Dwivedi and Kumar, 2016). The life cycle of the software characteristically includes a phase of requirement, a phase of design, a phase of implementation, a phase of testing, installation and maintenance. Testing of software is the major factor in quality control phase. Reliability of software is one of the key attributes of software quality. Reliability of software is described as probability of an operation of software without developing a failure during a specified time in a given specified environment (Kapur et al., 1999). Also, reliable software management is explained as the process involved in the optimising software reliability with the help of a program that is mainly concerned with prevention of error in the software, isolation of faults in software and their removal and using the measurement to increase its reliability to the maximum, keeping in view the constraints of the projects which include resources in developing the software, schedule and software performance (Lyu, 1996). Thus, software reliability comprises of three major events such as prevention of error, detection of its fault and removal, methods to increase its reliability to maximum (Kumar et al., 2012).

A software reliability growth model (SRGM) is a major characteristic feature of reliable software engineering. During the last 40 years, various SRGM have been projected as we try to understand the reason behind how and why failure of software
Fault dependency SRGM with testing effort using learning function

It occurs. It is a common assumption in SRGM that faults once detected are completely removed immediately, but this assumption is not realistic (Kumar, 2010).

There occurs a substantial delay of time in detection of faults and their removal (Dai et al., 2005). The required time for removal of fault is highly dependent on its complication, skills, experience and the number of members in debugging team, the methodology applied, etc. The difference in time that occurs between detection of faults and removal of faults must be negligible. Also, the faults can be mainly classified into two major types. The first are those faults which are mutually independent and second are those which are mutually dependent. The faults which are detected and removed directly are termed as mutually independent faults whereas, the faults which are removed only after removal of leading faults are termed as mutually dependent faults (Huang et al., 2004). The paper focuses on fault detection and its removal and subsequently, finding the mean value function (MVF) by using learning function.

In this paper, Section 2 describes notation for development of fault dependency SRGM with testing effort using learning function. In Section 3, we discuss literature review of software reliability growth developed under different set of assumptions by different researchers. Section 4 discusses the assumptions and proposed model with fault dependency with testing effort using learning function. In Section 5, we describe comparison criteria, real data set and description of tables, parameter results and goodness of fit curves. Section 6 concludes the paper.

2 Notations

To propose SRGM we have used the notations, which are as follows:

$m_1(t)$ MVF for the probable no. of leading faults

$m_2(t)$ MVF for the probable no. of dependent faults

$m(t)$ MVF, i.e., the expected no. of failures in software in the time (0, t)

$a$ estimated number of initial faults

$a_1$ total no. of leading faults

$a_2$ total no. of dependent faults

$p$ proportion of the leading faults

$\beta$ a constant in the learning function

$b(t)$ fault detection rate of dependent faults

$\gamma$ rate of error generation.
3 Review of literature for some existing models

Computers cannot exist without software, but the software must be reliable (Branch and Rocchi, 2015). Numerous SRGM have been proposed in the literature to measure the quality of software and to release the software at minimum cost (Kumar and Dwivedi, 2015). In this paper, we are going to propose a model based on fault dependency with testing effort.

We are reviewing the two basic models in the following section and comparing the result with the proposed model. Goel and Okumoto (GO) with testing effort is the first testing effort model in software reliability (Yamada et al., 1986) and Yamada with testing effort is the first model with the time lag function (Khatri et al., 2012).

3.1 GO with testing (M1)

Model given is based on the supposition that the errors are removed in an instant as they are detected, the process of removal of faults can be defined in a model as a single stage process. This can be given as

\[
\frac{dm_r}{dt} \times \frac{1}{w(t)} = b\left(a - m_r(t)\right)
\]

For \(m_r(t = 0) = 0\) and \(w(0) = 0\). On solving the equation given above, we found

\[
m_r(t) = a\left(1 - e^{-bW(t)}\right)
\]

The model is given by Yamada et al. (1986). This model is also discussed in Kumar and Dwivedi (2015).

3.2 Yamada with testing (M2)

In addition to assumptions (1–5), it is believed that removal of faults occur perfectly. The following model was given by Kapur and Garg (1991). This model is also discussed in Khatri et al. (2012). Model given below is the modification of NHPP to achieve graphical curve of S-shaped for increase in detection of fault, in such a way that the rate of failure of software increases at the beginning and decreases or decays later. S-shaped graphical curve can be considered as the process of learning which is defined by the process of software fault detection because skills of the software tester will gradually increase and improve with time.

This model of the fault removal phenomenon with testing effort can be specified by the equation given below

\[
\frac{dm_r(t)}{dt} \times \frac{1}{w(t)} = b(t)\left(a - m_r(t)\right)
\]

where \(b(t) = \left(\frac{b^2W(t)}{1+bW(t)}\right)\)

To get MVF, solving the above equation, for \(m_r(t = 0) = 0\) and \(W(t = 0) = 0\), we found.
Fault dependency SRGM with testing effort using learning function

\[ m_r(t) = a\left[1 - \left(1 + bW(t)e^{-bw(t)}\right)\right] \]  (3)

4 Proposed model

The SRGM proposed in this paper is based on fault dependency with assumption that the dependent faults can be isolated only when leading faults of dependent faults already removed. To propose our model we have assumed that SRGM is based on non-homogeneous poisson process. After developing model based on assumptions, we will verify and validate our model with the help of some of the statistical parameters.

Proposed SRGM is based on given assumption:
1 NHPP is followed by the removal of faults and process of detection
2 due to appearance of already remaining faults in the system, the system sometimes encounters failures
3 no. of faults are finite, broadly categorised in leading faults and dependent faults
4 in the given time interval \((t + \Delta t)\), the leading faults are directly proportional to the number of left over fault in system
5 in the given time interval \((t + \Delta t)\), the no. of dependent faults is directly proportional to the faults left behind in system
6 removal of detected dependent faults lags its detection by delay effect factor
7 throughout fault removal process, no new additional fault is generated.

Using assumption 3rd, it is known that the no. of faults that occur in software can be given by:

\[ a = a_1 + a_2 \]

We can suppose:

\[ m(t) = m_1(t) + m_2(t) \]

Considering the rate of fault detection for leading faults to be constant, the equation can be specified as

\[ \frac{dm_1(t)}{dt} \times \frac{1}{w(t)} = b\left(a_1 - m_1(t)\right) \quad a_1 > 0, 0 < b < 1 \]

For \(m_1(0) = 0\), on solving the above equation we get

\[ m(t) = a(1 - e^{-bw(t)}) \]  (4)

Assuming the supposition (5) and (6), we can get

\[ \frac{dm_2(t)}{dt} \times \frac{1}{w(t)} = b\left(a_2 - m_2(t)\right) \times \frac{m_1(t - \Delta t)}{a} \]
Considering negligible amount of time delay ($\Delta t = 0$) and considering fault removal rate for dependent fault $b(t)$ as learning function $b(t) = \frac{b}{1 + \beta \exp(-bW(t))}$, we can rewrite the above equation as:

$$\frac{dm_2(t)}{dt} = \frac{1}{w(t)} \frac{b}{1 + \beta \exp(-bW(t))} (a_2 - m_2(t)) \frac{m_1(t - \Delta t) - m_2(t)}{a}$$

After finding the solution of the differential equation given above under the boundary condition $m_2(0) = 0$ and $w(0) = 0$, we found the MVF of dependent faults as given below:

$$m_2(t) = a_2(1 - \exp\left(-\frac{a}{a_2}\left(\frac{\beta + 1}{\beta}\log\left(\frac{1 + \beta \exp(-bW(t))}{w(t)}\right) + bW(t)\right)\right))$$

The MVF $m(t)$ can be achieved from above equations as:

$$m(t) = m_1(t) + m_2(t)$$

And considering $a_1 = Pa$ and $a_2 = (1 - P)a$, $0 \leq P \leq 1$,

$$m(t) = \begin{cases} 
1 - \exp\left(-P\left(\frac{\beta + 1}{\beta}\log\left(\frac{1 + \beta \exp(-bW(t))}{1 + \beta}\right) + bW(t)\right)\right) \\
+ Pa \left(\exp\left(-P\left(\frac{\beta + 1}{\beta}\log\left(\frac{1 + \beta \exp(-bW(t))}{1 + \beta}\right) + bW(t)\right)\right) - \exp(-bW(t))\right) \end{cases}$$

5 Estimation of parameters and criteria of comparison

5.1 Criteria of comparison: goodness of fit

The capability of a model (6) can be examined by its capacity to replicate the experimental performance of the software. The comparisons criteria (Kapur et al., 1999) that are used are in this paper are:

1. mean square fitting error (MSE) (Kapur et al., 1999)
2. bias (Lyu, 1996)
3. variation (Lyu, 1996)
4. root mean square prediction error (Lyu, 1996)
5. coefficient of multiple determination ($R^2$) (Kapur et al., 1999).

5.2 Data analysis and model comparison

Performance of the proposed model has been surveyed on two continuous data sets and received results are evaluated against the current models.
5.2.1 Data set I (DS-1)

Primary data set (DS-1) used to test the efficiency of the model, had been gathered amid 35 months of the testing of a radar arrangement in which 1,301 deficiencies were found in the size of 124 kilo lines of code, amid testing. This data set was presented by Brooks and Motley (1980).

5.2.2 For DS-1

Estimation results for the effort function and the proposed SRGM are given in Table 1 and Table 2 respectively. Weibull describes the best fitted curve on this effort data set. Parameter estimation of SRGM is performed w.r.t. Weibull effort function. Statistical parameters of the proposed models are estimated and compared with various models. The comparison criteria are shown in Table 3, it can be seen that the proposed models provide a better fit to this data set. The fitting of the models is illustrated graphically in Figure 1.

<table>
<thead>
<tr>
<th>Parameter estimation</th>
<th>Goodness of fit criteria</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>v</td>
</tr>
<tr>
<td>Exponential</td>
<td>240,045.7</td>
</tr>
<tr>
<td>Rayleigh</td>
<td>2,873.408</td>
</tr>
<tr>
<td>Weibull</td>
<td>2,669.913</td>
</tr>
</tbody>
</table>

Table 2 Model parameter estimation results of DS-1

<table>
<thead>
<tr>
<th>Parameters</th>
<th>M1</th>
<th>M2</th>
<th>Proposed SRGM</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>1,546.171</td>
<td>1,298.628</td>
<td>1,356.759</td>
</tr>
<tr>
<td>b</td>
<td>0.0010962</td>
<td>0.0032953</td>
<td>0.00914</td>
</tr>
<tr>
<td>β</td>
<td>-</td>
<td>-</td>
<td>16.78549</td>
</tr>
<tr>
<td>P</td>
<td>-</td>
<td>-</td>
<td>0.215894</td>
</tr>
</tbody>
</table>

Table 3 Results of model comparison for DS-1

<table>
<thead>
<tr>
<th>Comparison criteria</th>
<th>Models under comparisons</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>M1</td>
</tr>
<tr>
<td>MSE</td>
<td>1,127,8161</td>
</tr>
<tr>
<td>BIAS</td>
<td>0.2122847</td>
</tr>
<tr>
<td>VARIATION</td>
<td>33.111073</td>
</tr>
<tr>
<td>R²</td>
<td>0.995</td>
</tr>
<tr>
<td>RMSPE</td>
<td>33.111753</td>
</tr>
</tbody>
</table>
5.2.3 Data set 2 (DS-2)

Secondary data set (DS-2) used to test the efficiency of the model had been gathered amid 19 weeks, 47.65 CPU hrs were frenzied and 328 flaws were identified amid testing. This information is referred to from Obha (1984).

5.2.4 For DS-2

The estimated values of parameters are given in Table 4 and Table 5 for the above data set. The comparison criterion with the existing models has been made in Table 6. The results of proposed model are found to be better than the existing models in terms of $R^2$, MSE, SSE, variation and RMSPE. The fitting of the models is illustrated graphically in Figure 2. It shows that the proposed model is better fit than other existing models.

Table 4 Data estimation on effort function

<table>
<thead>
<tr>
<th>Data set (DS-2)</th>
<th>Parameter estimation</th>
<th>Goodness of fit criteria</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$N$</td>
<td>$v$</td>
</tr>
<tr>
<td>Exponential</td>
<td>2,860.669137</td>
<td>8.65E-04</td>
</tr>
<tr>
<td>Rayleigh</td>
<td>49.31915535</td>
<td>0.013681</td>
</tr>
<tr>
<td>Weibull</td>
<td>799.2015065</td>
<td>2.33E-03</td>
</tr>
</tbody>
</table>

Table 5 Model parameter estimation results of DS-2

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Models under comparisons</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>M1</td>
</tr>
<tr>
<td>a</td>
<td>565.35181</td>
</tr>
<tr>
<td>b</td>
<td>0.019659</td>
</tr>
<tr>
<td>$\beta$</td>
<td>-</td>
</tr>
<tr>
<td>P</td>
<td>-</td>
</tr>
</tbody>
</table>
Table 6  Results of model comparison for DS-2

<table>
<thead>
<tr>
<th>Comparison criteria</th>
<th>Models under comparisons</th>
<th>Proposed SRGM</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSE</td>
<td>116.84349</td>
<td>70.81799861</td>
</tr>
<tr>
<td>BIAS</td>
<td>0.5640708</td>
<td>0.882915022</td>
</tr>
<tr>
<td>VARIATION</td>
<td>11.291195</td>
<td>14.85956</td>
</tr>
<tr>
<td>R²</td>
<td>0.989</td>
<td>0.978</td>
</tr>
<tr>
<td>RMSPE</td>
<td>11.305275</td>
<td>0.882915022</td>
</tr>
</tbody>
</table>

Figure 2  Goodness of fit (see online version for colours)

6  Conclusions

In this paper, we have differentiated between the process of detection of faults and its correction. These two are critical processes for acquiring a high quality software product. Leading and dependent are two kinds of faults can be classified in the software system. Those types of faults which are dependent can be detached only if the errors which are leading to those faults are already removed. We have proposed a model where we have incorporated a learning function in the MVF of the dependent faults and thereafter found the MVF for removing a fault. The proposed model is estimated on two real information sets and evaluated results are weighed against with some existed models for goodness of fit. The results obtained show that the proposed model have better goodness of fit as compared with existing models. In this paper we have considered time lag that exist in actual life between error recognition and its correction. We can also come up with new fault dependency model with more parameters added in the model. We have also considered perfect debugging in the modelling, so we can consider imperfect debugging in fault dependency model in the future. We can also apply change point concept in fault dependency model.
References


