A TESTING-EFFORT BASED SRGM INCORPORATING IMPERFECT DEBUGGING AND CHANGE POINT

Umashankar Samal[1⋆], Shivani Kushwaha[2], Ajay Kumar[3]

•

[1,2,3]Department of Applied Sciences, ABV-IIITM Gwalior, Gwalior-474015, MP, India
[1]umashankar@iiitm.ac.in, [2]shivanik@iiitm.ac.in, [3]ajayfma@iiitm.ac.in
⋆Corresponding Author

Abstract

In this paper, a scheme for constructing software reliability growth model based on Non-Homogeneous Poisson Process is proposed. Here, we consider the software reliability growth model that incorporates with imperfect debugging, change point and testing effort. However, most researchers assume a constant detection rate per fault in deriving their software reliability models. They suppose that all faults have equal probability of being detected during the software testing process, and the rate remains constant over the intervals between fault occurrences. In reality, the fault detection rate strongly depends on the skill of test teams, program size, and software testability. Also in most realistic situations, fault repair has associated with a fault re-introduction rate due to imperfect debugging phenomenon. In this case, the fault detection rate and fault introduction rate will be changed during the software development process. Therefore, here we incorporate both generalized logistic testing-effort function, change-point parameter into software reliability modelling. The Least Square Estimation approach is used to estimate the unknown parameters of the new model. So in our new proposed model we collect software testing data from real application and utilize it to illustrate the proposed model. Experimental results show that the proposed framework to incorporate both testing-effort and change-point for Imperfect-Debugging SRGM has a fairly accurate prediction capability.

Keywords: Stochastic software reliability model , Non-homogeneous Poisson process, Imperfect debugging, Testing effort, Change point.

Acronyms and Notations

NHPP	nonhomogeneous Poisson process
LSE	least squares estimation
SRGM	software reliability growth model
TEF	testing effort function
ID	imperfect debugging
$m(t)$	mean value function, i.e., the expected number of software failures by time t
$a(t)$	error content function
$b(t)$	error detection rate per error at time t
$R(t)$	reliability function
$R(x/t)$	conditional software reliability
β	fault introduction rate
$w(t)$	current testing-effort estimated by a logistic testing effort function
$W(t)$	cumulative testing effort estimated by a logistic testing effort function
τ	change point
$\lambda(t)$	failure density function

1. Introduction

There are many reasons that a software fails, but usually a software fails because of design issue. Other failures occur when the code is written, or when changes are introduced to a working system. For the past several decades, various statistical models have been proposed to assess the software reliability. So, we have reviewed many previous well known models. The NHPP based software reliability growth models are proved quite successful in practical software reliability engineering [1] . The main issue of the NHPP model is to determine an appropriate mean value function to denote the expected number of failures experienced up to a certain time point. Model parameters can be estimated by using maximum likelihood or least square estimate. Once the mean value function is determined the software reliability and the related measurements can be easily derived.

We have found that most SRGMS use calendar time as a unit to fault detection and removal [2, 3, 5, 8, 16], but very few use human power, number of test case runs, or CPU time as a unit [12]. Researchers have proposed SRGMs that incorporates the concept of TEF into an NHPP model to get a better description on the software fault phenomenon [4, 6, 7, 9, 10, 11, 13, 15, 17, 18, 19, 20]. TEF has the advantage of relating the work profile more directly to the natural structure of software development. Many researchers assume that all faults have equal probability of being detected during the software testing process, and the rate remains constant over the intervals between fault occurrences, constant detection rate per fault, but we, in our model, assume that the rate changes before and after a fixed point. Most authors also assume that the once the fault is removed there will be no new faults introduced but we assume that even during debugging new faults will be introduced, and also introduction rate may not be same during overall testing. So in our model, there will be a imperfect debugging with change point and testing effort scenario.

2. Related work

In this section, we reviewe the well-known NHPP SRGMs and then introduce our new general model that incorporates both the imperfect debugging, change-point problem and testing effort. So we first start with some well known SRGMs.

2.1. A general NHPP model

Let $N(t)$ be a counting process representing the cumulative number of software failure by time t. The counting process is shown to be a NHPP with a mean value function $m(t)$. Mean value function represents the expected number of failures by time t. Goel and Okumoto [2] assume that number of software failures is time independent and software failure density is proportional to residual fault content. Thus $m(t)$ can be solved by solving the following differential equation.

$$\lambda(t) = \frac{dm(t)}{dt} = b(a - m(t)) \tag{1}$$

Where a denotes the initial number of faults contained in a program and b represents the fault detection rate. The result shows that

$$m(t) = a(1 - e^{-b\,t}) \tag{2}$$

The conditional software reliability, $R(x|t)$, is defined as the probability that there is no failure observed in the time period $(t, t + x)$, given that the last failure occurred at a time point $t(t \geq 0; x > 0)$. Given the mean value function $m(t)$, the conditional software reliability can be shown as

$$R(x|t) = e^{-[m(t+x)-m(t)]} = exp[-a(e^{-bt} - e^{-b(t+x)}] \tag{3}$$

2.2. Imperfect software debugging models

In the general NHPP model, a constant fault content function implies the perfect debugging assumption,i.e., no new faults are introduced during the debugging process. Pham introduced an NHPP SRGM in imperfect debugging environment. He assumed if detected faults are removed, then there is a possibility of introduction of new faults with a constant rate β. Let $a(t)$ be the number of faults to be eventually detected (denoted by a) plus the number of new faults introduced to the program by time t, the mean value function $m(t)$ can be given as the solution of the following system of differential equations:

$$\frac{\partial m(t)}{\partial t} = b[a(t) - m(t)], \frac{\partial a(t)}{\partial t} = \beta \frac{\partial m(t)}{\partial t} \tag{4}$$

Solving the above equations, we can obtain the mean value function and conditional software reliability, respectively, as follows

$$m(t) = \frac{a}{1-\beta}\left[1 - e^{-(1-\beta)bt}\right], R(x|t) = exp\left(-m(x)e^{-(1-\beta)bt}\right) \tag{5}$$

2.3. An NHPP model with change-point

Many SRGMs suppose the fault detection rate is a constant, or a monotonically increasing function. The failure intensity is expected to be a continuous function of time. An increasing fault detection rate function represents the debugging process with the learning phenomenon. But the fault detection rate can be affected by many factors such as the testing strategy and resources allocation. During a software testing process, there is a possibility that the underlying fault detection rate function is changed at some time moments τ, called as change-point.

Chang [7] considered the change-point problems in the NHPP SRGMs. The parameters of the NHPP with change-point models are estimated by the weighted least square method. Let the parameter τ be the change point that is considered unknown and is to be estimated from the data. The fault detection rate function is defined as :

$$b(t) = \begin{cases} b_1 & when \quad 0 \le t \le \tau, \\ b_2 & when \quad t > \tau \end{cases}$$

On solving for $m(t)$

$$m(t) = \begin{cases} a(1 - e^{-b_1 t}) & when \quad 0 \le t \le \tau, \\ a(1 - e^{-b_1\tau - b_2(t-\tau)}) & when \quad t > \tau \end{cases}$$

Reliability function will be

$$R(x|t) = \begin{cases} exp\left\{-a\left(e^{-b_1 t} - e^{-b_1(t+x)}\right)\right\} & when \quad t \le t+x \le \tau, \\ exp\left\{-a\left(e^{-b_1 t} - e^{-b_1\tau - b_2(t+x-\tau)}\right)\right\} & when \quad t \le \tau \le t+x \\ exp\left\{-a\left(e^{-b_1\tau - b_2(t-\tau)} - e^{-b_1(t+x)}\right)\right\} & when \quad \tau < t \end{cases}$$

2.4. Imperfect software debugging model with change point

To consider the NHPP SRGM that integrates imperfect debugging with change-point problem, the following assumptions are made:

(a) When detected faults are removed at time t, there is a possibility of introduction new faults at a rate $\beta(t)$.

$$\beta(t) = \begin{cases} \beta_1 & when \quad 0 \le t \le \tau, \\ \beta_2 & when \quad t > \tau \end{cases}$$

(b) The fault detection rate represented as following is a step function

$$b(t) = \begin{cases} b_1 & when \quad 0 \le t \le \tau, \\ b_2 & when \quad t > \tau \end{cases}$$

(c) A NHPP model of fault detection phenomenon in the software system.

The testing strategy and resource allocation can be tracked all the time during the fault detection process. It may be more reasonable to reconsider that the change-point (τ) is given. According to these assumptions, one can derive the new set of differential equations to obtain the new mean value function.

$$\frac{\partial m(t)}{\partial t} = b[a(t) - m(t)], \frac{\partial a(t)}{\partial t} = \beta \frac{\partial m(t)}{\partial t}, a(0) = a, m(0) = 0 \tag{6}$$

Solving the differential equation (6)

$$m(t) = \begin{cases} \frac{a}{1-\beta_1}[1 - e^{-(1-\beta_1)b_1 t}] & when \quad 0 \le t \le \tau, \\ \frac{a}{1-\beta_2}[1 - e^{-(1-\beta_1)b_1\tau-(1-\beta_2)b_2(t-\tau)}] + \frac{m(\tau)(\beta_1-\beta_2)}{1-\beta_2} & when \quad t > \tau \end{cases}$$

$$\lambda(t) = \frac{\partial m(t)}{\partial t} = \begin{cases} ab_1 e^{-(1-\beta_1)b_1 t} & when \quad 0 \le t \le \tau, \\ ab_2 e^{-(1-\beta_1)b_1\tau-(1-\beta_2)b_2(t-\tau)} & when \quad t > \tau \end{cases}$$

3. Proposed model

In this section, we propose a new Software Reliability Model that incorporates imperfect debugging with change-point and testing effort. Beginning with the necessity of testing in software reliability, we will make some assumptions for our model to construct it. In the beginning of the testing phase, many faults can be discovered by inspection and the fault detection rate depends on the fault discovery efficiency, the fault density, the testing effort, and the inspection rate. Later, the fault detection rate depends on some more additional parameters such as the failure-to-fault relationship, the code expansion factor, the skill of test teams, program size, and software testability.

Here we use a NHPP model with TEF and The following assumptions are made for the same.

(a) The fault removal process follows the Non-Homogeneous Poisson Process (NHPP).

(b) The software system is subject to failures at random times caused by the manifestation of remaining faults in the system.

(c) The mean number of faults detected in the time interval $(t, t + \lambda t)$ by the current testing-effort expenditures is proportional to the mean number of remaining faults in the system.

(d) The consumption curve of testing-effort is modelled by a generalized logistic TEF.

$$W(t) = \frac{N}{1 + A\,exp[-\alpha t]}$$

where

N = total amount of testing-effort to be eventually consumed,

α = consumption rate of testing-effort expenditures,

A = constant

(e) When detected faults are removed at time t, it is possible to introduce new faults with introduction rate $\beta(t)$.

$$\beta(t) = \begin{cases} \beta_1 & when \quad 0 \le t \le \tau, \\ \beta_2 & when \quad t > \tau \end{cases}$$

(f) The fault detection rate represented as following is a step function.

$$b(t) = \begin{cases} b_1 & when \quad 0 \le t \le \tau, \\ b_2 & when \quad t > \tau \end{cases}$$

We can describe an SRGM based on generalized logistic TEF with fault introduction rate and change-point as follow:

$$\begin{gathered} \frac{dm(t)}{dt} * \frac{1}{w(t)} = b(t) * (a - m(t)) \\ \frac{\partial a(t)}{\partial t} = \beta \frac{\partial m(t)}{\partial t} \\ a(0) = a, m(0) = 0 \end{gathered} \tag{7}$$

$W(t)$ is defined as-

$$W(t) = \int_0^t w(\tau)\, d\tau$$

From the above differential equation, the mean value function will be

$$m(t) = \begin{cases} \frac{a}{1-\beta_1}\left[1 - e^{-(1-\beta_1)b_1(W(t)-W(0))}\right] & when \quad 0 \le t \le \tau, \\ \frac{a}{1-\beta_2}\left[1 - e^{-(1-\beta_1)b_1(W(\tau)-W(0))-(1-\beta_2)b_2(W(t)-W(\tau))}\right] + \frac{m(\tau)(\beta_1-\beta_2)}{1-\beta_2} & when \quad t > \tau \end{cases}$$

Failure density function is

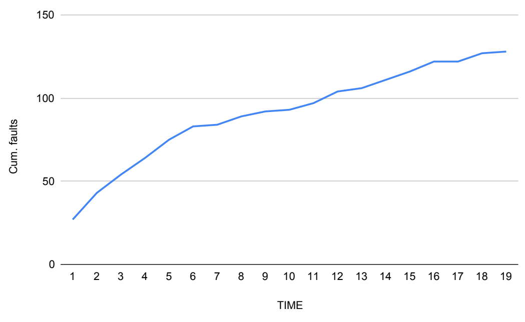
$$\lambda(t) = \frac{\partial m(t)}{\partial t} = \begin{cases} ab_1 w(t) e^{-(1-\beta_1)b_1(W(t)-w(0))} & when \quad 0 \le t \le \tau, \\ ab_2 w(t) e^{-(1-\beta_1)b_1(W(\tau)-W(0))-(1-\beta_2)b_2((W(t)-W(\tau))} & when \quad t > \tau \end{cases}$$

The results for mean value function and failure intensity function obtained have integrated the imperfect debugging change point problem and testing effort problem into a single NHPP SRGM. The unknown parameters for the above equations can be calculated using LSE.

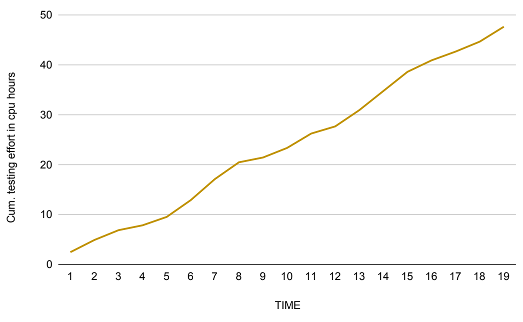
4. Numerical and Data Analysis

4.1. Description of real dataset

To verify the new proposed model and to evaluate the performance of the SRGM, We have taken a dataset from Ohba [14]. The total testing time, cumulative number of software failures and cumulative testing consumption of generalized logistic TEF are recorded.
To assume the change point for the given set of data given in Table (1), graph (1a) of the cumulated number of faults versus time has been considered. It is found that it is not differentiable around 11.2.
During the testing period, 20 Hours of experiments, 47.65 CPU Hours were consumed and about 128 software errors are removed. To find the unknown parameters of the logistic testing effort function, we have used LSE. Using LSE, the unknown parameters are found to be $N = 54.823$, $A = 13.033$, $\alpha = 0.226$.



(a) *Cumulative faults of $N(t)$ process*



(b) *Testing efforts vs time*

Figure 1: *CF and TEF versus time*

Table 1: *Ohba data set*

time	cumulative failure	testing effort consumption	time	cumulative failure	testing effort consumption
1.00	27.00	2.45	11.00	97.00	26.23
2.00	43.00	4.90	12.00	104.00	27.67
3.00	54.00	6.86	13.00	106.00	30.93
4.00	64.00	7.84	14.00	111.00	34.77
5.00	75.00	9.52	15.00	116.00	38.61
6.00	83.00	12.89	16.00	122.00	40.91
7.00	84.00	17.10	17.00	122.00	42.67
8.00	89.00	20.47	18.00	127.00	44.66
9.00	92.00	21.45	19.00	128.00	47.65
10.00	93.00	23.35			

4.2. Model Comparison

In the paper, we have compared the accuracy with Delayed S-shaped model [3] and Huan-Jyh Shyur's imperfect debugging and change point model [1].

No.	**Model**	**m(t)**
1	Delayed S-shaped [3]	$m(t) = a\left(1 - (1+bt)\,e^{-bt}\right)$
2	Huan-Jyh Shyur's model [1]	$m(t) = \begin{cases} \frac{a}{1-\beta_1}\left[1 - e^{-(1-\beta_1)b_1 t}\right] & when \quad 0 \le t \le \tau, \\ \frac{a}{1-\beta_2}\left[1 - e^{-(1-\beta_1)b_1\tau-(1-\beta_2)b_2(t-\tau)}\right] + \frac{m(\tau)(\beta_1-\beta_2)}{1-\beta_2} & when \quad t > \tau \end{cases}$
3	New model	$m(t) = \begin{cases} \frac{a}{1-\beta_1}\left[1 - e^{-(1-\beta_1)b_1(W(t)-W(0))}\right] & when \quad 0 \le t \le \tau, \\ \frac{a}{1-\beta_2}\left[1 - e^{-(1-\beta_1)b_1(W(\tau)-W(0))-(1-\beta_2)b_2(W(t)-W(\tau))}\right] + \frac{m(\tau)(\beta_1-\beta_2)}{1-\beta_2} & when \quad t > \tau \end{cases}$

4.3. Comparision Criteria

In order to compare the performance of the proposed model with other models, we have used MSE [17]. MSE is defined as:

$$MSE = \sum_{i=1}^{n} \frac{[m(t_i) - m_i]^2}{D}$$

where $m(t_i)$, m_i and D represent estimated values, observed values and degrees of freedom respectively.

5. Results

The unknown parameters in the proposed model are a, b_1, b_2 and the unknown parameters in the testing effort function are α, N, A. We have examined the proposed model with the given dataset,

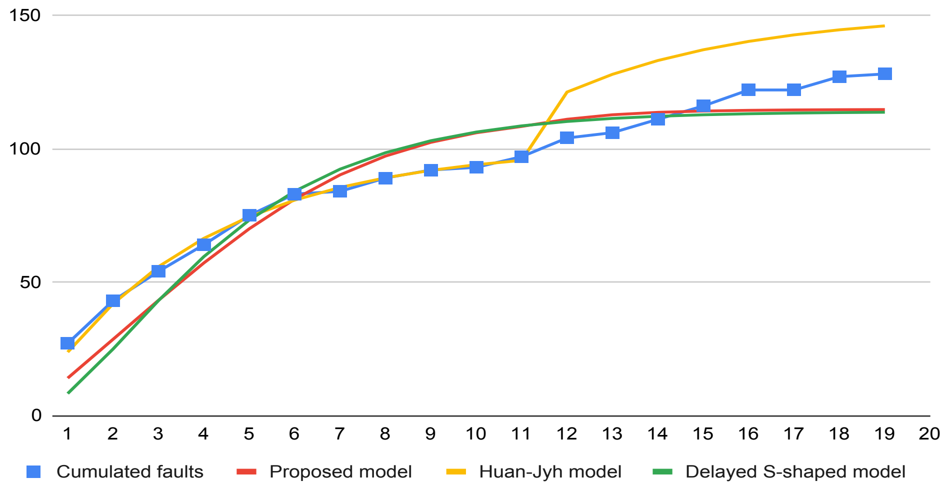


Figure 2: *Mean value functions $m(t)$ of all models for dataset1*

and the results from the TEF i.e. $N = 54.826$, $\alpha = 13.033$, $A = 0.226$. The fault introduction rate before change point β_1 is taken as 0.2 and the fault introduction rate after change point β_2 is taken as 0.5. Using LSE, the values of a, b_1 and b_2 are found to be 90.0059 0.1834 and 0.388 respectively.

Table 2: *Estimation of parameters for datasets*

No.	Model	Parameter	MSE
1	Delayed S-shaped [3]	$a = 113.9062$, $b = 0.438$	127.0564
2	Huan-Jyh Shyur's model [1]	$a = 80.75$, $b_1 = 0.335$, $b_2 = 0.5$	195.4614
3	New model	$a = 90.75$, $b_1 = 0.1834$, $b_2 = 0.3887$	101.8713

We have compared the proposed model with previous two well known models and the result shows that with introduction of testing effort function, performance of the model has increased.

6. Conclusion

In this paper, we present a new change point software reliability model considering the testing effort function based on NHPP and imperfect debugging environment. A generalized logistic testing effort function and effect of change point in a imperfect debugging environment are discussed and the explicit mean value function for the new model is presented. Furthermore, comparisons of this model with several existing change point, imperfect debugging models have also been provided in terms of values of MSE on Ohba data set. Numerical results demonstrate that the proposed model can give a better goodness-of-fit. It seems that this proposed model, though a little more sophisticated but By means of incorporating both ID, testing effort and the effect of change point, provides a more powerful property to model the changing fault detection rate, which describes more realistically actual effects of the real testing process.

References

[1] Shyur, Huan-Jyh. "A stochastic software reliability model with imperfect-debugging and change-point." *Journal of Systems and Software* 66, no. 2 (2003): 135-141.

[2] Goel, A.L. and Okumoto, K., 1979. Time-dependent error-detection rate model for software reliability and other performance measures. *IEEE transactions on Reliability*, 28(3), pp.206-211.

[3] Yamada, Shigeru, Mitsuru Ohba, and Shunji Osaki. "S-shaped software reliability growth models and their applications." *IEEE Transactions on Reliability* 33, no. 4 (1984): 289-292.

[4] Chiu, Kuei-Chen, Yeu-Shiang Huang, and Tzai-Zang Lee. "A study of software reliability growth from the perspective of learning effects." *Reliability Engineering & System Safety* 93, no. 10 (2008): 1410-1421.

[5] Pham, Hoang, and Xuemei Zhang. "NHPP software reliability and cost models with testing coverage." *European Journal of Operational Research* 145, no. 2 (2003): 443-454.

[6] Zhao, Ming. "Change-point problems in software and hardware reliability." *Communications in Statistics-Theory and Methods* 22, no. 3 (1993): 757-768.

[7] Chang, I. P. "An analysis of software reliability with change-point models." Taipei, Taiwan: National Science Council (1997).

[8] Pham, Hoang, Lars Nordmann, and Zuemei Zhang. "A general imperfect-software-debugging model with S-shaped fault-detection rate." *IEEE Transactions on reliability* 48, no. 2 (1999): 169-175.

[9] Huang, Chin-Yu, Sy-Yen Kuo, and Michael R. Lyu. "An assessment of testing-effort dependent software reliability growth models." *IEEE transactions on Reliability* 56, no. 2 (2007): 198-211.

[10] Huang, Chin-Yu. "Performance analysis of software reliability growth models with testing-effort and change-point." *Journal of Systems and Software* 76, no. 2 (2005): 181-194.

[11] Rafi, Shaik Mohammad, and Shaheda Akthar. "Software reliability growth model with logistic-exponential testing effort function and analysis of software release policy." *In Proceedings of international conference on advances in computer science*. 2010.

[12] Yamada, Shigeru, and Shunji Osaki. "Software reliability growth modeling: Models and applications." *IEEE Transactions on software engineering* 12 (1985): 1431-1437.

[13] Pradhan, Sujit K., Anil Kumar, and Vijay Kumar. "An Optimal Resource Allocation Model Considering Two-Phase Software Reliability Growth Model with Testing Effort and Imperfect Debugging." *Reliability: Theory & Applications* SI 2 (64) (2021): 241-255.

[14] Ohba, Mitsuru. "Software reliability analysis models." *IBM Journal of research and Development* 28, no. 4 (1984): 428-443.

[15] Khurshid, Shozab, A. K. Shrivastava, and Javaid Iqbal. "Effort based software reliability model with fault reduction factor, change point and imperfect debugging." *International Journal of Information Technology* 13, no. 1 (2021): 331-340.

[16] Li, Qiuying, and Hoang Pham. "A testing-coverage software reliability model considering fault removal efficiency and error generation." PloS one 12, no. 7 (2017): e0181524.

[17] Shrivastava, Avinash K., and P. K. Kapur. "Change points based software scheduling." *Quality and Reliability Engineering International* 37, no. 8 (2021): 3282-3296.

[18] Sharma, Dinesh K., Deepak Kumar, and Shubhra Gautam. " Flexible software reliability growth models under imperfect debugging and error generation using learning function." *Journal of Management Information and Decision Sciences* 21, no. 1 (2018): 1-12.

[19] Chatterjee, Subhashis, Deepjyoti Saha, and Akhilesh Sharma. "Multi upgradation software reliability growth model with dependency of faults under change point and imperfect debugging."; *Journal of Software: Evolution and Process* 33, no. 6 (2021): e2344.

[20] Panwar, Saurabh, Vivek Kumar, P. K. Kapur, and Ompal Singh. "Software reliability prediction and release time management with coverage." *International Journal of Quality & Reliability Management* (2021).