# COST OPTIMIZATION AND RELIABILITY GROWTH MODELS WITH IMPERFECT DEBUGGING AND CHANGE POINTS

Madhu Jain, Department of Mathematics, IIT Roorkee, Roorkee, Hardwar- 247 667 (India)
(madhufma@iitr.ac.in, drmadhujain.iitr@gmail.com)

**ABSTRACT**

The number of faults in the software (S/w) can be reduced by debugging process which includes the observing or locating the faults and then put appropriate efforts for the removal of the faults. Most of the reliability growth models study the perfect debugging process; however this is not the case in real world scenarios due to fact that during the debugging process some new fault may also come on the surface. In this investigation, reliability modelling of the software system with imperfect debugging (ID) is presented. The software reliability has been examined by the assessment of fault contents in the S/w and associated total system testing costs. To develop the generalized SRGM, the fault reduction, ID and change point are included. Furthermore, testing effort function (TEF) is also considered to solve the optimization problem subject to reliability constraint. The numerical simulation and sensitivity analysis have been carried out to determine the optimal time to release the S/w.

**Keywords**: Software reliability, NHPP, Testing effort, Imperfect debugging, Change points, Cost model,S/w release policy.

## INTRODUCTION

Most of the reliability growth models available in literature are developed by considering the perfect process for debugging; however this is not the case in real world scenarios due to fact that some faults may enter during the debugging process also. For the prediction of the software reliability indices, an imperfect debugging process should be taken into account while developing the NHPP models for the software.

The software reliability literature is mainly concerned with the analysis of mean value function in the perfect debugging environment during the testing process. There is a need of developing software reliability model which will be suited for the imperfect debugging environment. In recent past, a few papers on the software reliability growth have appeared which studied the imperfect testing process (Xia *et al*., 1993; Kapur et al., 1994;Kapur and Younes; Zeephongsekul, 1995). Chatterjee *et al*. (2004) have provided reliability indices by considering the imperfect debugging phenomenon to analyze the NHPP model for the N-version software system. The imperfect debugging and learning phenomenon were taken into consideration by Pham (2007) while developing the NHPP model for the assessment of reliability metrics ofS/w. Jain and Jain (2013) employed the quasi renewal process to explore the S/w testing and imperfect debugging.

The NHPP for determining the software reliability release policy (SRP) using the testing effort functions can be modeled as Exponential, Rayleigh and Weibull (Catuneanu *et al*., 1991). Kapur and Bhalla (1992) discussed various aspects of SRP by developing the NHPP model for reliability prediction. Recently, Cortellessa *et al*.(2015) have managed the evolution of the software architecture to minimize the costs while keeping the reliability constraints within certain threshold limits. For controlling the reliability growth of the concerned S/w, change points and fault reduction factor play important role. Fault reduction factor may also be affected by the environment. Some factors like imperfect debugging, time lag etc. can be incorporated to analyze the effect of environmental factors on the SRP (Huang and Hung, 2010, Pachauri *et al*., 2015; Jain et al., 2016).

## MODEL DESCRIPTION

To describe the SRGM, we consider the initial number of faults and fault detection rate. The fault detection process depends on many factors including the software testability. The fault detection is assumed to change at specific points known as change points. The testing efforts consumed are assumed to be Weibull type to examine how effectively the faults are identified and subsequently removed from the S/w.

For modeling the NHPP model for the assessment of reliability growth,the following notations have been used:

$a(\tau)$ Total number of faults detected including the newly introduced fault sat instant $\tau$ .
a Number of faults detected.
β Fault introduction rate.
η(γ) Scale (Shape) parameter.
K Average total testing effort expenditures required for the software testing.
$d(\tau)$ Fault detection rate function at instant $\tau$ .
$b_i$ Fault detection rate per unit testing effort for $i^{th}$ stage of change point, $1 \le i \le n$
$\mu(\tau)$ Fault reduction factor at instant $\tau$ .
$w(\tau)$ Mean instantaneous testing effort expenditure at instant $\tau$ .
$m(\tau)$ Mean number of faults detected in the given time interval $(0,\tau]$ .

Now we shall develop the software reliability growth model (SRGM) by incorporating some more realistic features such as (i)fault reduction factor (FRF) (ii) imperfect debugging (iii) testing effort of Weibull type(iv) multiple change points. The software reliability growth model is formulated by considering the NHPPand under certain assumptions which are stated below:

I. NHPP is considered for the fault removal process.
II. The average faults detected in $(\tau,\tau+\Delta\tau]$ is affected by the faults already present in the S/w system.
III. The fault removal process is imperfect i.e. during the fault removal process, there is possibility of introducing some new faults.
IV. Testing effort consumption is assumed to be governed by Weibull distribution.
V. The concept of multiple change points is considered.

**GOVERNING EQUATIONS AND ANALYSIS**

The cumulative testing efforts consumed without change points in time interval $(0,\tau]$ is determined using

$$F(\tau) = K\left(1-\exp\left[-\eta\tau^{\gamma}\right]\right) \quad \ldots(1)$$

The current testing effort expenditure is obtained using

$$f(\tau) = \frac{d}{d\tau}F(\tau) \quad \ldots(2)$$

Due to imperfect debugging, there may be the possibility of introducing some new faults with introduction rate $\beta(\tau)$ . Thus we have

$$\frac{da(\tau)}{d\tau} = \beta(\tau)\frac{dm(\tau)}{d\tau} \quad \ldots(3)$$

In the present investigation, we assume that in the time interval $(0,\tau]$ , the fault detection rate depends on several factors and may change during fault debugging process at some points of time (say $\tau_j$; j=1,2,...,n), which are called the change points. At any instant of time $\tau$ , the fault detection rate is defined by

$$b(\tau) = \begin{cases} b_{1,} & for\ \tau\ (0,\tau_1] \\ b_j, & for\ \tau_{j-1} < \tau \le \tau_j, j = 2,3,\ldots,n \\ b_{n+1}, & for\ \tau > \tau_n \end{cases} \quad \ldots(4)$$

Also, we assume that the fault introduction rate $\beta(\tau)$ changes at the change point $\tau_j$; (j=1,2,...,n), where $\tau_j$ represents the time epoch of the $j^{th}$ change. Also denote $\tau_0$=0. We define the fault introduction rate with multiple change points during the testing of the software is by

$$\beta(\tau)=\begin{cases}\beta_{1,} & for\ 0\le\tau\le\tau_1\\ \beta_j, & for\ \tau_{j-1}<\tau\le\tau_j, j=2,3,...,n\\ \beta_{n+1}, & for\ \tau>\tau_n\end{cases} \quad \ldots(5)$$

The total faults contents by the time t is given by

$$a(\tau)=\begin{cases}a_1(\tau)=a+\beta_1 m(\tau), & for\ 0\le\tau\le\tau_1\\ a_j(\tau)=a+\sum_{i=1}^{j-1}(\beta_i-\beta_{i+1})m(\tau_i)+\beta_j m(\tau), & for\ \tau_{j-1}<\tau\le\tau_j, j=1,2,...,n\\ a_n(\tau)=a+\sum_{i=1}^{n}(\beta_i-\beta_{i+1})m(\tau_i)+\beta_{n+1}m(\tau), & for\ \tau>\tau_n\end{cases} \quad \ldots(6)$$

During the software-testing, Weibull-type testing effort function is assumed tobe changed and is defined by

$$F(\tau)=\begin{cases}F_1^*(\tau)=K\left(1-\exp\left[-\eta_1\tau^{\gamma_1}\right]\right), & 0\le\tau\le\tau_1\\ F_j^*(\tau)=K\left(1-g_1+g_2-\exp\left[-\eta_2\tau^{\gamma_2}\right]\right), & \tau_{j-1}<\tau\le\tau_j, j=1,2,...n\\ F_{n+1}^*(\tau)=K\left(g_{n+1}-\exp\left[-\eta_{(n+1)}\tau^{\gamma_{(n+1)}}\right]+\sum_{m=1}^{n}g_m-\exp\left[-\eta_m\tau_m^{\gamma_m}\right]\right), & \tau_n<\tau\end{cases} \quad \ldots(7)$$

where $g_i=\exp[-\eta_i\tau_{i-1}^{\gamma_i}]$.

The fault reduction factor is also altered at change point. Thus at time t fault reduction factor μ(t) is defined by

$$\mu(\tau)=\begin{cases}\mu_{1,} & for\ 0\le\tau\le\tau_1\\ \mu_j, & for\ \tau_{j-1}<\tau\le\tau_j, j=2,3,...,n\\ \mu_{n+1}, & for\ \tau>\tau_n\end{cases} \quad \ldots(8)$$

To develop the generalized software reliability growth model, we formulate the equation for the mean value function (MVF) in time interval $(0,\tau)$ as-

$$\frac{dm(\tau)}{d\tau}\frac{1}{f(\tau)}=b(\tau)\mu(\tau)\left[a(\tau)-m(\tau)\right] \quad \ldots(9)$$

Solving (9) by using the (4)-(8), the modified mean value function is obtained as-

$$m(\tau)=\begin{cases}m_1(\tau)=\frac{a}{(1-\beta_1).}\left[1-e^{-(1-\beta_1)b_1\mu_1[F_1{}^*(\tau)]}\right], & for\ 0\le\tau\le\tau_1\\ m_j(\tau)=\frac{1}{(1-\beta_j)}\left[a+\sum_{i=1}^{j-1}(\beta_i-\beta_{i+1})m(\tau_i)\right]\left\{1-e^{-b_j\mu_j(1-\beta_j)F_j^*(\tau)}\right\}+m(\tau_{j-1})e^{-b_j\mu_j(1-\beta_j)F_j^*(\tau)}, & for\ \tau_{j-1}<\tau\le\tau_j, j=2,3,...,n\\ m_{n+1}(\tau)=\frac{1}{(1-\beta_{n+1})}[a+\sum_{i=1}^{n}(\beta_i-\beta_{i+1})m(\tau_i)]\left\{1-\mathrm{e}^{-b_{n+1}\mu_{n+1}(1-\beta_{n+1})F_{n+1}^*(\tau)}\right\}+m(\tau_n)\mathrm{e}^{-b_{n+1}\mu_{n+1}(1-\beta_{n+1})F_{n+1}^*(\tau)}, & for\ \tau>\tau_n\end{cases} \quad \ldots(10)$$

The generalized failure intensity function $\Lambda(\tau)$ with multiple change points by incorporating imperfect debugging and fault reduction factor is obtained using

$$\Lambda(\tau) = \frac{dm(\tau)}{d\tau} \quad \ldots(11)$$

**RELIABILITY INDICES**

The software reliability during the testing phase is given by

$$R(x/T) = \exp[-\{\Delta m(T)\}]$$

$$= \begin{cases} \exp\left[\frac{a}{(1-\beta_1)}\exp\{-(1-\beta_1)b_1\mu_1 K\}\left\{\exp\{(1-\beta_1)b_1\mu_1 K\exp[-\eta_1(T+x)^{\gamma_1}]\}\right.\right. \\ \qquad \left.\left. -\exp\{(1-\beta_1)b_1\mu_1 K\exp[-\eta_1 T^{\gamma_1}]\}\right\} - \right], \quad 0 \le T \le \tau_1 \\ \exp\left[(1-\beta_2)^{-1}\{a+(\beta_1-\beta_2)m(\tau_1)\}\exp\{-b_2\mu_2(1-\beta_2)\}K\left(1-\exp[-\eta_1\tau_1^{\gamma_1}]+g_2\right)\right]. \\ \left\{\exp\{-b_2\mu_2(1-\beta_2)K\left(-\exp[-\eta_2(T+x)^{\gamma_2}]\right)\} - \exp\{-b_2\mu_2(1-\beta_2)\left[K\left(-\exp[-\eta_2(T^{\gamma_2})]\right)\right]\}\right\} \\ +m(\tau_1)\exp\{-b_2\mu_2(1-\beta_2)\left[K\left(1-\exp[-\eta_1\tau_1^{\gamma_1}]+g_2\right)\right]\} \\ \times\left\{\exp\left[-b_2\mu_2(1-\beta_2)\left[K\left(-\exp[-\eta_2(T^{\gamma_2})]\right)\right]\right] - \exp\{-b_2\mu_2(1-\beta_2)\left[K\left(-\exp[-\eta_2(T+x)^{\gamma_2}]\right)\right]\}\right\}\Big], \\ \qquad \tau_1 < T \le \tau_2 \end{cases} \quad \ldots(12)$$

The cost of software depends on the length of testing, consumption of testing resources and types of testing methodologies. These factors affect the quality of the software. The expected total cost is determined by

$$E\{C(F(T))\} = \sum_{i=1}^{n} C_{1i} m_i(F_i^*\ (T)) + \sum_{i=1}^{n} C_{2i}(a_i - m_i(F_i^*\ (T))) + C_3^* \sum_{i=1}^{n} F_i^*(T) \ldots(13)$$

where $C_{1i}$ ( $C_{2i}$ ) and$C^*_3$ are the costs of fixing a fault during the testing(operational) phase for the $i^{th}$ stage of change point and cost of per unit Weibull type testing effort respectively. Here $F^*_i$ denotes the testingefforts for $i^{th}$ stage during testing time T.

**NUMERICAL RESULTS AND CONCLUSION**

To perform the numerical simulation, we consider the software reliability for two change point and default parameters as a=50, $b_1$=0.5, $b_2$=0.06, K=8, $\beta_1$=0.9, $\beta_2$=0.6, $\mu_1$=0.8, $\mu_2$ =0.4, $\gamma_1$=0.9, $\gamma_2$=0.6, $\eta_1$=0.5, $\eta_2$=0.6, x=0.5 and $\tau_1$=2. The Figures 1(i-vi) reveal the numerical results for the reliability of the software with respect to testing time 'T' for different values of initial fault contents (a), fault detection rate for two change points ($b_1$), testing effort expenditure parameter (K), scale parameter ($\eta_1$), shape parameter ($\gamma_1$,), fault introduction rate factor ($\beta_1$), fault reduction factor ($\mu_1$ ) for both change points, respectively. The numerical results can be concluded with the remarks that the reliability of the S/w is significantly improved by increasing the testing time up to a certain point; after that it does not change. Also software reliability is lower for one change point in comparison to second stage change point but this trend continues only for lower values of T.

In this investigation, multiple change points and Weibull testing efforts included to examine the software testing efficiency may be helpful to design robust and cost effective software. Software reliability obtained may provide valuable insights to the S/w manufacturer for the releasing of the S/w at appropriate time. The expected maintenance cost during software testing phase reveals the economic aspects for every stage of the change point.

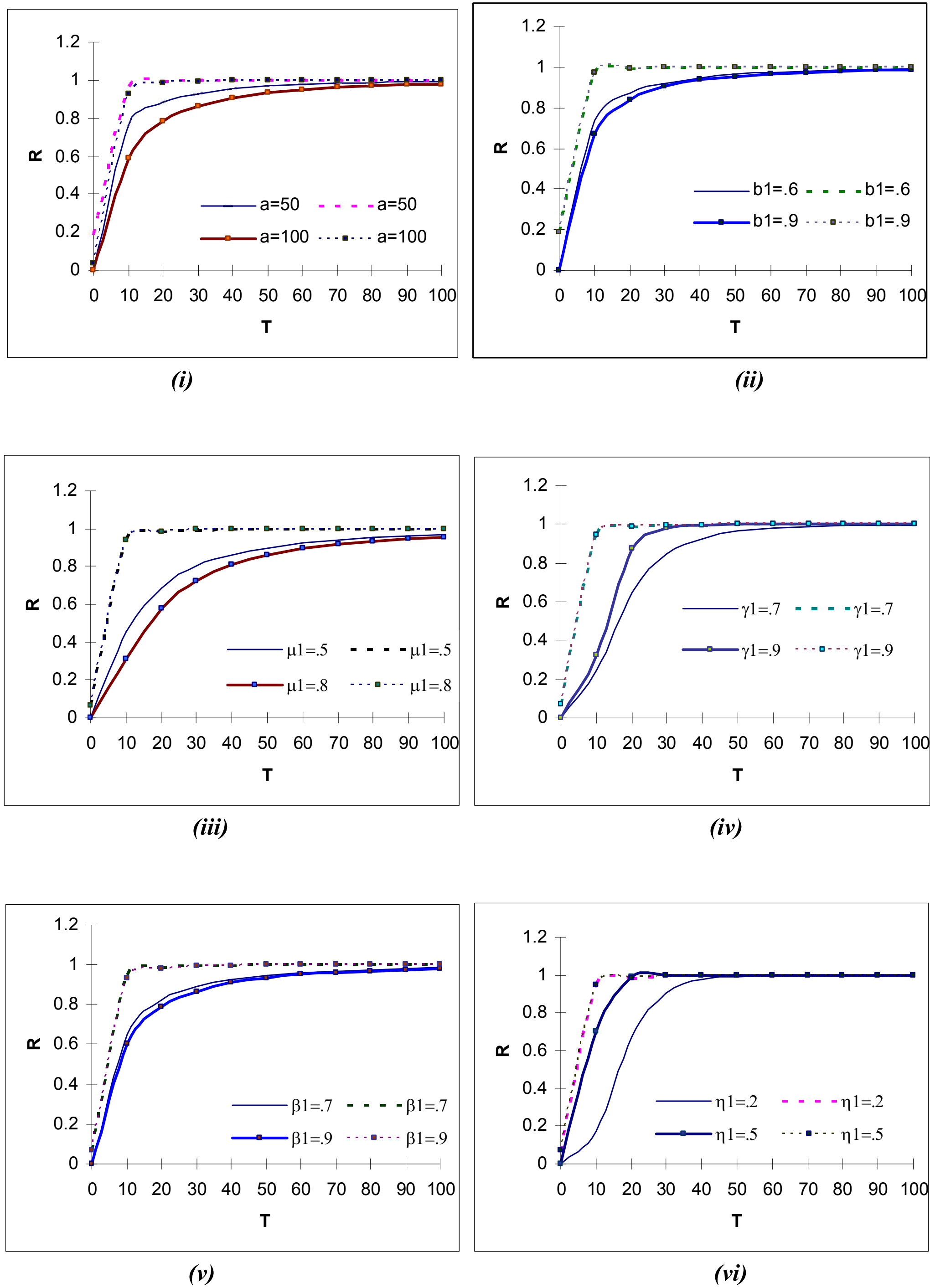


**Figure 1(i-vi):** Reliability vs testing time by varying (i) a (ii) $b_1$ (iii) $\mu_1$ (iv) $\gamma_1$ (v) $\beta_1$ and (vi) $\eta_1$

**REFERENCES**

Catuneanu, V. M., Moldovan, C., Popentiu, Fl., & Popovivi, D. (1991). Software reliability release policy with testing effort, *Microelectronics Reliability*, 31, 895-899.

Chatterjee, S. Mishra, R. B., & Alam, S. S. (2004). N-version programming with imperfect debugging, *Computer Electrical Engineering*, 30, 453-463.

Cortellessa, V., Mirandola, R., & Potena, P. (2015). Managing the evolution of a software architecture at minimal cost under performance and reliability constraints, *Science of Computer Programming*, 98, 439-463.

Huang, C., & Hung (2010). Software reliability analysis and assessment using queueing models with multiple change point, *Computers Mathematics with Applications*, 60, 2015-2030.

Jain, M.,& Jain, A. (2013). Quasi renewal analysis of software reliability growth model incorporating testing efforts and time delay, *International Journal of Mathematics in Operational Research,* 5, 721-742.

Jain, M., Manjula, T., & Gulati, T.R. (2016). Optimal release policies of delayed discrete reliability growth model with imperfect debugging, *Journal of Testing and Evaluation*, 44, 1376-1382.

Kapur, P. K., & Bhalla, V. K. (1992). Optimal release policies for a flexible software reliability growth model, *Reliability Engineering and System Safety*, 35, 49-54.

Kapur, P. K., & Younes, S. (1995). Modelling an imperfect debugging phenomenon in software reliability, *Microelectronics Reliability,* 36, 645-650.

Kapur, P. K., Aggrawal, S., Younes, S., & Sinha, A. (1994). On a general imperfect debugging software reliability growth model, *Microelectronics Reliability,* 34, 1397-1403.

Pachauri, B., Dhar, J., & Kumar, A. (2015). Incorporating inflection S-shaped fault reduction factor to enhance software reliability growth, *Applied Mathematical Modelling,* 39, 1463-1469.

Pham, H. (2007). An imperfect-debugging fault detection dependent-parameter software, *International Journal of Automation and Computing,* 4, 325-328.

Wang, J., Wu, Z., Shu, Y., & Zhang, Z. (2015). An imperfect software debugging model considering log-logistic distribution fault content function, *Journal of Systems and Software*, 100, 167-181.

Xia, G. Zeephongsekul, P., & Kumar, S. (1993). Optimal software release policy with a learning factor for imperfect debugging, *Microelectronics Reliability*, 33, 81-86.

Zeephongsekul, P. (1995). Reliability growth of a software model under imperfect debugging and generation of errors, *Microelectronics Reliability*, 36, 1475-1482.