ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 2, Issue 1, January 2014

A Quantitative Analysis of NHPP Based Software Reliability Growth Models

Bijoyeta Roy[1], Santanu Kr. Misra[2], Aradhana Basak[3], Aparupa Roy[4], Deyasini Hazra[5]

Assistant Professor, Department of CSE, Sikkim Manipal Institute of Technology, Rangpo, India[1]

Associate Professor, Department of, Sikkim Manipal Institute of Technology, Rangpo, India[2]

UG Students, Department of, Sikkim Manipal Institute of Technology, Rangpo, India[3,4,5]

ABSTRACT: The application of computer systems is so widely spread that it can be found in almost all complex equipments. So, reliability of software becomes a major concern as unreliable software may lead to huge economic loss to the organization. A software reliability growth model (SRGM) basically predicts the fault detection coverage in software testing phase. The general problem that is encountered is to minimize the number of remaining faults for a given fixed amount of testing effort and reliability objective. In this paper a general framework of two non homogeneous poisson processes based SRGM models namely Goel okumoto and Delayed S Shaped model are presented. The main objective of these two models is to estimate the faults or failures remaining in the system.

Keywords: Software Reliability, SRGM, NHPP, imperfect debugging.

I. INTRODUCTION

Now a day’s large and complex software systems are developed by integrating a number of small and independent modules. A system basically consists of hardware and software. Software is essentially an instrument for transforming a discrete set of input to output. Software testing begins at component level in software development phase and is usually very expensive and lengthy process as most of the commercial software products are complex systems consisting of a number of modules. So it becomes very important for the project managers to allocate specified testing resources among all the modules and develop quality software with high reliability. According to ANSI definition software reliability is defined as the probability of failure free operation for a specified period of time in a specified environment. In general it can be defined as how well the software functions to meet the requirement of customers. To predict the reliability of software many SRGM have been developed during 1970-2000. An SRGM describes failures as random process and is based on NHPP. An NHPP is a Poisson’s process with rate parameter $\chi(t)$ such that rate parameter is a function of time. It is assumed that software reliability can somehow be measured and therefore the question is that what purpose it serves. Software reliability is a useful measure in planning and controlling resources during the development process so that high quality software can be developed. It is seen that the assessed value of the reliability measure is always relative to a given user environment. In this paper two traditional SRGM models are analyzed using various parameters such as imperfect debugging and concepts of NHPP [3, 6, 7].

II. LITERATURE SURVEY

In the past literature on software reliability Chin-Yu Huang in 2001 proposed an SRGM with generalized logistic TEF and elaborated a unified approach for developing software reliability growth models in the presence of imperfect debugging and error generation reliability. Dr. Ajay Gupta, Dr. Digvijay Choudhary and Dr. Sunit Saxena in 2006 discussed the software reliability estimation using delayed s-shaped model under imperfect debugging and proposed a model based testing considering cost, reliability and software quality. Xue Yang, Nan Lang & Hang Lei proposed improved NHPP model with time varying fault removing delay. Michael Grottke in 2007 analysed the software reliability model study by implementing with debugging parameters. Jang Jubhu gave an elaborate introduction to software reliability growth models using various case studies in 2008. Bruce R. Maxim in 2010 calculated the reliability of DSS model using mean time value function and some other parameters. In this paper focus was given on S shaped SRGM using a flexible modelling approach. Basically problems associated with manual operations are huge time consumption, more prone to faults and High failure rate. The proposed SRGM will replace the manual operation of analysis and complex computations with computerized one to increase the reliability.

III. NOTATIONS

t	: time.
χ (t)	: failure rate intensity.
μ(t)	: mean value function.
N	: expected total number of detected failures.
B	: occurrence of rate of failure, proportionality constant.
L	: reliability measure.
f_o	: expected number of initial faults.
f_i	: total number of independent faults.
f_d	: total number of dependent faults.
r	: fault detection rate of independent faults.
θ	: fault detection rate of dependent faults.
p	: proportion of independent faults.
φ(t)	: delay effect factor.
ψ	: Inflection factor.
m(t)	: mean value function of expected number of faults detected.
m_d(t)	: mean value function of expected number of dependent faults detected.
m_i(t)	: mean value function of expected number of independent faults detected.
b_i	: independent fault detection rate[11].

IV. GOEL OKUMOTO MODEL

The primary objective of Software reliability model is to predict the failure behavior of software. Goel Okumoto model provides the analytical framework for describing the software failure phenomenon during testing.

A. Basic assumptions of Goel Okumoto Model

The model proposed by Goel Okumoto is based on the following assumptions:-

i) The execution time between the failures is exponentially distributed.
ii) The cumulative number of failures by time t follows a Poisson process with mean value function µ (t).
iii) The number of software failures that occur in (t, t+$\wedge$t) with $\wedge$t → 0 is proportional to the expected number of undetected faults, N- µ (t). The constant of proportionality is φ.
iv) For any finite collection of times $t_1<t_2<t_3<\ldots<t_n$, the number of failures occurring in each of the disjoint intervals (0, t_1),(t_1, t_2),(t_2, t_3)…(t_{n-1}, t_n) is independent.
v) Fault causing failure is corrected immediately otherwise reoccurrence of that failure is not counted [8].

Since each fault is perfectly repaired after it has caused a failure, the number of inherent faults in the software at the beginning of testing is equal to the number of failures that will have occurred after an indefinite amount of testing.

B. Reliability Analysis

In this model, Goel and Okumoto assumed that software is subject to failures at random times caused by faults present in the system.

Let N (t) be the cumulative number of failures observed by time t and they proposed that N(t) can be modeled as a non homogeneous Poisson process that is a poisson process with a time dependent failure rate.

The proposed form of the model is as follows:

$$P\{N(t)=y\} = (m(t))^{y} e^{-m(t)}/y!\ ,\ y=0, 1, 2.. \qquad (1)$$

Where $m(t) = a(1-e^{-bt})$ and $\chi(t) = m't = abe^{-bt}$

Here m (t) is the mean value function or the expected number of failures observed by time t and χ (t) is the failure intensity rate [8].

A typical plot of m (t) function for the Goel Okumoto model with a= 175 and b= 0.05 is shown in figure 1.

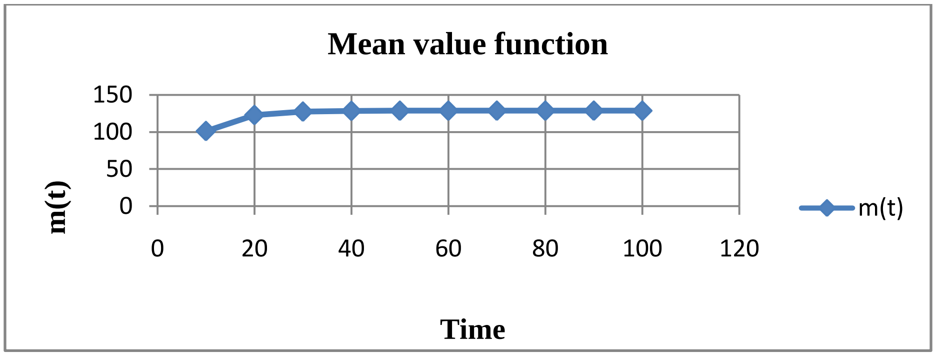

Fig. 1: Mean value function with respect to time

In this model 'a' is the expected number of failures to be observed eventually and 'b' is the fault detection rate per fault. Model estimation and reliability prediction can be done using

$$L = \prod_{i=1}^{n}[m(t_i) - m(t_i - 1)]^{f_i}\ \exp\ [m(t_i) - m(t_i - 1)]/f_i \qquad (2)$$

Where fi is the failure count in each of the testing intervals and t_i is the completion time of each period that the software is under observation. Plot of reliability prediction in Goel Okumoto model for a= 175 and b= 0.05 is shown below.

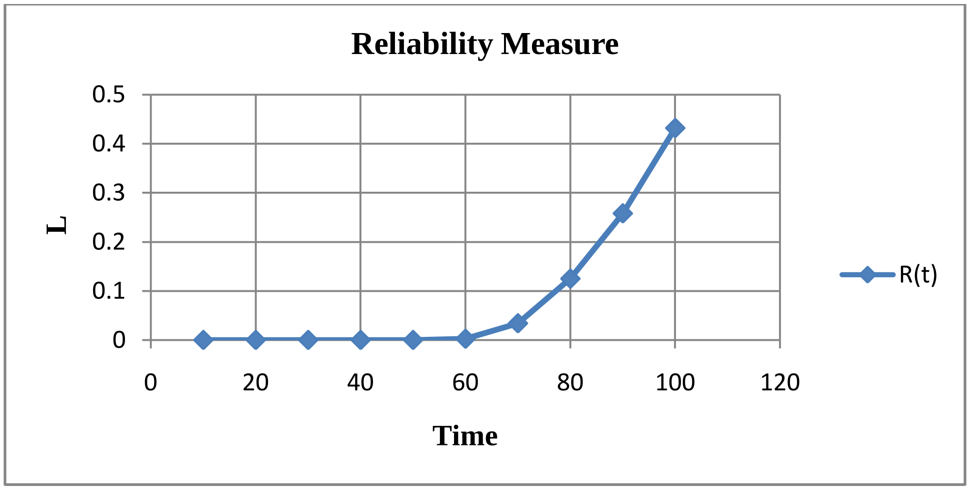

Fig. 2: Reliability measure with respect to time

V. DELAYED S SHAPED MODEL

Reliability of software depends upon fault detection and error correction. The analysis of the problem can be made using dependent and independent faults whether the faults can be detected and corrected or not. The software reliability using this model can be analyzed by involving imperfect debugging and time delay function.

A. Basic assumptions of Delayed S Shaped model

The model proposed by Yamada is based on the following assumptions:

i) All detected faults are either independent that is the faults are detected and corrected immediately and there is no time delay between them or dependent that faults cannot be removed immediately.

ii) The software failure rate at any time is a function of fault detection rate and the number of remaining faults present at that time.

iii) The total number of faults is infinite.

iv) The detected dependent fault may not be removed immediately and it lags the fault detection process by $\emptyset(t)$ [2, 5].

B. Faults in Delayed S Shaped SRGM model

Total detected faults in time (0, t) is given by:

$$\mathrm{m}\ (\mathrm{t}) = \mathrm{m}_d\ (\mathrm{t}) + \mathrm{m}_i\ (\mathrm{t}) \qquad (3)$$

i) Independent faults ($m_i(t)$)

The rate of independent faults detected is proportional to the remaining faults. The differential equation is

$$\frac{\boldsymbol{d}}{\boldsymbol{dt}} m_i(t) = r[f_i - m_i(t)] \quad f_i > 0, \qquad 0<r<1 \qquad (4)$$

The differential equation under imperfect debugging is

$$\frac{d}{dt} m_i(t) = r[f_i + b_1 m_i(t) - m_i(t)] \qquad (5)$$

And using time delay function $\emptyset(t)$ is

$$m_i(t) = \frac{f_i}{(1-b_{1)}}[1 - \exp\{1 - r(1-b_1)(t - \emptyset(t))\}] \qquad (6)$$

ii) Dependent faults ($m_d(t)$)

The rate of dependent faults detected is proportional to the remaining dependent faults in the system and the ratio of independent faults removed at time 't' to the total number of faults.

$$\frac{d}{dt} m_d(t) = \theta[f_d - m_i(t)]\frac{m_i(t)}{f_0} \quad ; 0 < \theta<1 \qquad (7)$$

C. Reliability Analysis

This model describes the S shaped curve for the cumulative number of faults detected such that the failure rate initially increases and later decreases.

$$\emptyset(t) = \frac{1}{r(1-b_{1)}}\ln\{1 + (1-b_1)rt\} \qquad (8)$$

The software reliability is defined as the probability that a software failure does not occur in the time interval (t, t + Δt):

$$R\left(\frac{\Delta t}{t}\right) = \exp[-\{m\ (\mathrm{t} + \Delta\mathrm{t}) - \mathrm{m(t)}\}]\ \mathrm{t} \geq 0, \Delta\mathrm{t} \geq 0. \qquad (9)$$

The number of failures m (t) and software reliability R(t) have been evaluated considering the input as f_0= 400, r= 0.255, θ= 0.0833, ψ= 2.84, p=0.55, b=0.1, f_i=0, 1, 2….

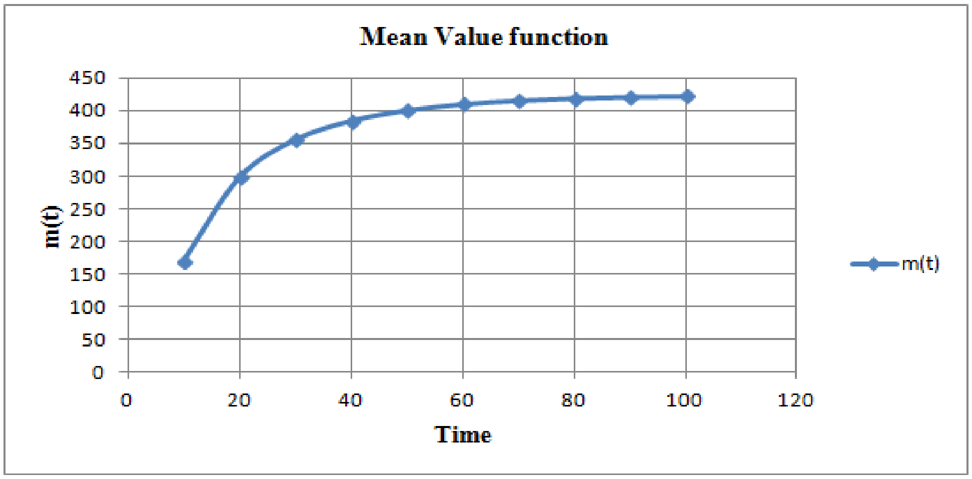

Fig.3: Variation of mean value function with time

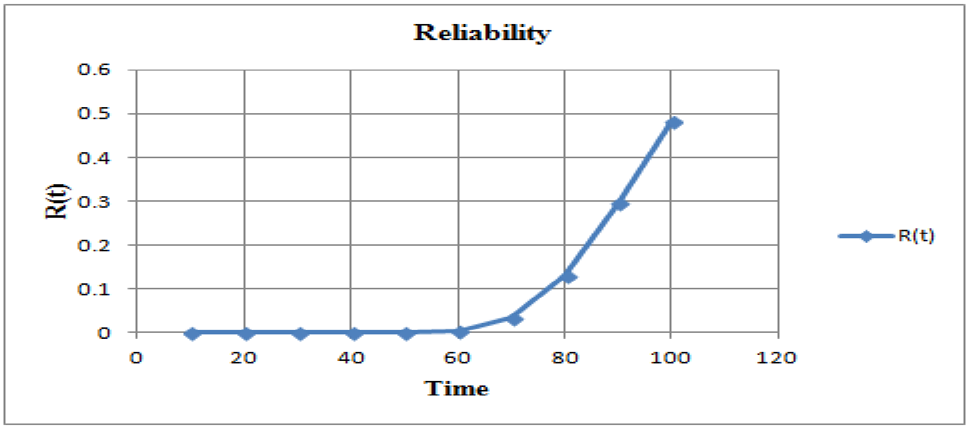

Fig. 4: Variation of Software reliability with time

Fig.3 represents the variation of number of faults detected with respect to time. Initially the faults detected during testing are very high but later on becomes constant. The number of faults debugged under imperfect debugging is higher than that in perfect debugging.

Fig.4 represents the variation of software reliability with respect to testing time. Software reliability increases rapidly with testing time during initial phase. If we incorporate the factors like faults dependency, debugging time lag and imperfect debugging into model, prediction of software reliability is more realistic and generalized.

VI. CONCLUSION

In this paper a review of software reliability growth models is presented. Two classes of analytical models along with their underlying assumptions were described. It should be noted that the above analytical models are primarily useful in estimating and monitoring software reliability. A generalized framework for software reliability growth modeling is analyzed with respect to testing effort and faults of different severity. Software reliability growth model can provide a good prediction of number of faults at a particular time and can compute the remaining numbers of failures also. When the software is at high risk, the testing effort would be too high and the rate of detection of errors would be high and value of ‘b’ would approach to 1 and hence reliable software can be produced. The present study is based on assumption of independence of failures of modules and in future the dependence of failures from different modules can also be considered and reliability can be studied.

REFERENCES

[1] C.Y Hung, M. R Lyu, and S. Y Kuo, “A unified scheme of some non-homogeneous poison process models for Software reliability estimation”, IEEE transaction on Software Engineering, Vol 29,Issue.3, pp. 261-269,2003.
[2] P.K Kanpu , H. Pham, S. Anand and K .Yadav, “An Unified approach for developing software reliability growth models in the presence of imperfect debugging and error generation reliability”, IEEE transaction ,Vol.6, Issue 1, pp. 331-340,2003.
[3] M.R Lyu, “Handbook of software reliability engineering”, Mc Graw Hill, pp.428-443, 1993.
[4] A. L Goel and K. Okumoto “Time dependent error detection rate model for software reliability and other performance measures” IEEE transaction on reliability, Vol 28, Issue.3, pp 206-211, 1979.
[5] H. Hetaera and S. Yamada, “Optimal allocation and control problems for software testing resources”, IEEE transaction on reliability, Vol.39, Issue. 2, pp 171-176, 1990.
[6] P. Kubat and H.S Koch, “Managing test procedure to achieve to achieve reliable software”, IEEE transaction on reliability, Vol 32, Issue 3, pp-299-303, 1983.
[7] B. Littlewoods, Software reliability growth model for modular programming structure. IEEE transaction on reliability. Vol. 28, Issue 3, PP-35-41, 1989.
[8] P. Nagar and B. Thankachan, “Application of Goel-Okumoto Model in Software Reliability Measurement”, International Journal of Computer Applications, Vol.30, Issue 2, pp 0975 – 8887, 2012.

[9] B.W Boehm, J.R Brown, and M. Lipow, “Quantitative evaluation of software quality”, In proceeding 2nd International conference software engineering, Sanfrancisco, CA, Vol. 26, Issue. 3 PP-592-605, 1986.
[10] C.V Ramamoorthy and F.B Batsani, “Software reliability: status and prospectives”, IEEE transaction on software engineering, Vol. 8,Issue 2, pp-359-371, 1982.
[11] C.Y Huang and M. R. Lyu, “Optimal Testing Resource Allocation, and Sensitivity Analysis in Software Development”, IEEE transactions on reliability, vol. 54, Issue 4, December 2005

BIOGRAPHY

Ms. Bijoyeta Roy is an Assistant Professor in the Department of Computer Science & Engineering at Sikkim Manipal Institute of Technology, Sikkim. She received B.E in Computer Science & Engineering from Assam Engineering College, Gauhati University in 2008 and completed M. tech in Computer Science & Engineering from SMU in 2013. She has around two years of industrial experience and three and half years of experience in teaching. Her research interests include Software Engineering, Project management etc.

Mr. Santanu Kumar Misra is an Associate Professor in the Department of Computer Science & Engineering at Sikkim Manipal Institute of Technology, Sikkim. He received B.E in Computer Science & Engineering from WBUT, M. Tech in Computer Science & Engineering from SMU. He has around six year’s professional experience in teaching. His research interests include Operations management, Project management, applied soft computing etc.