International Journal of Advance Research in Science and Engineering
Volume No.07, Special Issue No.04, March 2018
www.ijarse.com
IJARSE
ISSN: 2319-8354

Studies of Software Fault Detection and Correction Process from the Perspective of Debugging Time Lag and Fault Dependency

Iqra Saraf, Javaid Iqbal

Deptt. of Computer Sciences,University of Kashmir, Srinagar, India

ABSTRACT

Software reliability can be described as the likeliness that software will operate successfully in the interval from time 0 to t. Software reliability growth models act as tools to measure reliability of software.They are software formulations which can be used as an indication of number of failures as they act as a measure to predict failure behavior of software in future from its certain or presumed features.Reliability is dependent on fault detection and correction process. Much work has been done on Software reliability models so that work on reliability estimation can be carried.These models aid us in detecting faults and correcting the same. Detection and Correction models can be derived in perspective of imperfect debugging,effort, error generation, change point, debugging lag, fault dependency etc.Here, we study about detection and correction in view of debugging time lag and fault dependency.It is often presumed that there is no gap between these two processes which is not a realistic. Thus detection and correction is taken as single process/distribution. But, in actual, time lag always exists between these two operations which is non-negligible. Further, it is discussed about mean value functions incorporating both debugging lag and fault dependency has been reviewed.

Keywords: Fault Correction (FCP), Fault Dependency, Fault Detection Process (FDP), Mean Value Function (MVF), Non Homogeneous Poisson Process (NHPP), Software Reliability Growth Model (SRGM)

Notations:

a	sum total of detected faults
v	failure occurrence rate per fault/ fault detection rate in case of independent faults.
c	Fault detection rate in case of dependent faults.
q	Proportion of independent faults
$m_d(t), m_c(t)$	Expected number of faults detected and corrected in time(0,t) respectively.
Δt	Debugging time lag
$\lambda(t)$	Intensity function

$F(t)$	Distribution function
f_0	Expected initial fault count.
f_i	Independent fault count
f_d	Dependent fault count.
$m_{df}(t)$	MVF of the expected count of dependent faults which have been detected in time $(0,t)$
$m_{if}(t)$	MVF of expected count of independent faults which have been detected in time $(0,t)$
a_1, a_2	Sum total of independent and dependent faults respectively.

I Introduction

Since computers are facing markedly new changes, human beings are feeling the need of software and hardware systems an unavoidable requisite. As a result, reliable performance of software has turned out to be an essence because reliability is the prominent feature of caliber software. To depict the fault associated demeanor like detecting, removal and introduction is the goal of reliability modeling. Most of the models believe that there exists only detection process of faults and the correction is perfect, with the result there is no process known as debugging time. But the same is not true as the fault which is detected needs to be rectified [1]. To assure complete reliability, both fault detection and correction needs to be addressed. Once the faults are detected, faults corrected can be modeled with a mapping having similar form as fault detection function along with a time lag. The rate by which faults are corrected is pro rata to that of fault detection [2] i.e. both go hand in hand excluding the delay. The fault correction process has to take into consideration various elements like the delay factor while correcting the faults, number and proportion of faults which are rectified, remaining number of faults, dependency between faults etc.

Considering the case of software, many growth models have been put forth to evaluate its reliability and amend it. Modeling of fault correction processes was first put forth by Schneidewind wherein it was proposed that the all faults which are detected are corrected after a constant time delay in a testing phase which is a two level process [3]. Then this model was stretched forth by Xie and Zhao (1992) to another version by replacement of constant delay function with a time dependent one. They concluded that it becomes difficult to correct detected Faults while test phase is in progression. Then, Schneidewind took delay as a random entity which follows an experiential/exponent form of distribution [2].A general framework for modeling both the processes was then presented [4]. Xie et al. focused more on correction process of faults wherein they represented it as delay in detection process, the delay being random or deterministic [5]. Kapur et al. by integrating/mixing failure observation, fault detection and removal processes developed a practical SRGM wherein a delay among these processes was demonstrated [6]. Jia and others developed a fault correction process through a reliability model

grounded on Markov process [7]. Then a unified framework was developed where fault observation and removal are separate testing processes and when both are single testing processes [8]. Vijay Kumar and Ranita Sahni proposed a fault detection and correction model and presumed that both are concurrent activities. They modelled FDP and FCP as independent but simultaneous activities with different budgetary constraints [9]. They concluded that more is the delay; the more complex is correction of fault.

There are various software strategies and methods which helps programmers in eliminating faults efficaciously. There are many data sets of Failure data available in literature which has been put down in a systematic and organized manner. These are used to figure out or predict the reliability of software. But dependency between faults is not always easily understood from these datasets [10]. Ohba put forth an S-shaped inflection model which explained the process of occurring of software failure with mutual dependency between faults. He conceived the idea that faults can be classified into two categories namely mutually independent and mutually dependent faults [11]. At the start, it was Kapur and Younes who proposed that model of fault dependency model is a double staged process consisting of leading and dependent faults. Herein, it is presumed that a detected fault which is dependent in nature may not be eliminated at once, thus it interims the detection process by a debugging time lag [12]. Many SRGMs were presented which while modeling both types of faults incorporate many lag functions [10].Singh et al. while modeling the same used power function of testing time[13]. Recently, Peng and Zhai put forth a framework for modeling the reliability wherein faults were categorized in above mentioned two types and the test process was took as a paired fault detection and correction [14]. The upcoming sections are organized as follows: Section II introduces fault detection and correction process, Section III and IV discusses about debugging time lag and dependency of faults, Section V presents the conclusion.

II Fault Detection and Fault Correction Processes in SRGMs

Most of SRGMs presume that fault detection process adopts Non Homogenous based Poisson Process with Mean value function whose rate of change decreases exponentially [15].In software fault detection process, the intensity function denoted by $\lambda_d(t)$ is dependent on attribute time. If $\lambda_d(t)$ is given, the mean value function $m_d(t)$, a prominent feature of NHPP model, fulfills the criteria that:

$$m_d(t) = \int_0^t \lambda_d(t)\,dt \qquad (1)$$ or,

$$\lambda_d(t) = \frac{d}{dt}\left(m_d(t)\right) \qquad (2)$$

Generally, the MVFs are written as:

$$m_d(t) = aF(t) \qquad (3)$$

If $F(t)$ takes the value of experiental distribution, it becomes the famous Goel Okumoto model:

$$m_d(t) = a.(1-\exp(-vt)) \qquad (4)$$

v depicts failure occurrence rate per fault (Goel Okumoto), $a, v > 0$.

$F(t)$ can take any distribution like weibull, gamma, erlang and so on. We can model Correction Process of Faults as delayed Fault Detection as we can correct fault only once it has been detected. The difference between fault detection and correction lies in the time used to rectify the detected fault i.e. time delay $\Delta(t)$. From $m_d(t)$ and time delay i.e. $\Delta(t)$, FCP models are derived. i.e. in case of GO model:

$$m_c(t) == a(1-\exp(-vt+v\Delta)) \qquad (5)$$

The models based on fault dependency and time lag follow these basic assumptions:

1. At any random time, software can fail because of the leftover errors in it.
2. Removal of faults is done by a phenomenon which is modeled by non homogeneous poisson process (NHPP).
3. Faults are eliminated without injecting more faults during the fault elimination process.
4. At any point of time, faults removed are directly proportional to remaining faults.
5. Faults are finite and mutually independent.
6. Given inputs are compared with the obtained outputs for desired results for occurrence of success or failure.
7. $\Delta(t)$ is the time delay between elimination of leading and dependent fault.
8. The mean number of leading faults detected/spotted in the time interval $(t, t+\Delta t)$ is proportional to the mean number of left over leading faults in the system.
9. The mean number of dependent faults detected/spotted in the interval $(t, t+\Delta t)$ is pro rata to the mean number of left over dependent faults and to the proportion of leading faults eliminated at time 't' and the sum total of faults.

III Considering Various Forms of Debugging Time Lag

The below mentioned differential equation is derived by considering the basic assumptions:

$$\frac{dm(t)}{dt} = v(t)\left(a(t) - m(t)\right) \qquad (6)$$

If, $v(t) = v$, $a(t) = a$

Then, at $m(0) = 0$ equation (6) becomes

$$m_d(t) = a\left(1 - e^{-vt}\right) \quad (7)$$

From assumption 7, the new MVF [15] becomes:

$$m_c(t) = m_d(t - \Delta t) \quad (8)$$

If $\Delta(t) = 0$, (detection rate equals correction rate) then,

$$m_r(t) = a\left(1 - \exp(-vt)\right) \quad (9)$$

which is a Goel Okumoto model.

If $\Delta(t) = \left(\frac{1}{v}\right)\ln(1 + vt)$, [15] then,

$$m_r(t) = a\left(1 - (1 + vt)\exp(-vt)\right) \quad (10)$$

which is Yamada delayed S-shaped model.

If $\Delta(t) = \left(\frac{1}{v}\right)\ln\left(\frac{(\Psi + 1)}{(1 + \Psi e^{-vt})}\right)$, then[10],

$$m_c(t) = a\frac{1 - e^{-vt}}{1 + \Psi e^{-vt}} \quad (11)$$

which is MVF for Inflected S-shaped model. If $\Delta(t) = \frac{1}{v}\ln\left(1 + vt + \frac{v^2t^2}{2}\right)$, then,

$$m_c(t) = a\left[1 - \left(1 + vt + \frac{v^2t^2}{2}\right)\exp(-vt)\right] \quad (12)$$

which is a 2-stage Erlang model.

IV Considering Fault Dependency

SRGMs assume that when models are developed, the faults detected are corrected immediately. But it is not a realistic assumption because time to get rid of detected fault is dependent on many things like team skill, fault complexity i.e. whether it is simple, hard or complex fault etc. Faults are of two types. Mutually independent and mutually dependent faults. In the former, there is no time delay between the two processes i.e. in between detection and correction.Whereas faults cannot be eliminated instantaneously in mutually dependent faults [11]. Various forms of $\Delta(t)$ can be incorporated in this dependency model. The total faults which can be detected in time $(0, t)$ are written as [16] :

$$f_0 = f_i + f_d \quad (13)$$

or we can assume that

$$m(t) = m_{df}(t) + m_{if}(t) \quad (14)$$

Where, $m(t)$ is the Mean Value Function (MVF) of the expected count of faults which have been detected in time $(0,t)$.

Also, $a = a_1 + a_2$ (15)

If, $m_{if}(t) = a_1\left(1-e^{-vt}\right)$

From assumption 7, we get [15],

$$\frac{d\left(m_{df}(t)\right)}{dt} = c\,x\left[a_2 - m_{df}(t)\right]x\frac{m_{if}(t-\Delta t)}{a_1} \quad (16)$$

Here, if $a_1 = qa$ and $a_2 = (1-q)a$, $0 \le q \le 1$

Using equation (14, 16), various forms of debugging lag discussed above and boundary condition $m_{df}(0) = 0$, t=0, we get $m(t)$ as [18]:

1. If $\Delta t = 0$ then,

$$m(t) = a\left(1 - qe^{-vt} - (1-q)e^{\left(\frac{c\left(1-e^{-vt}\right)}{v} - tc\right)}\right) \quad (17)$$

2. If $\Delta t = \frac{1}{v}\ln(1+vt)$ then,

$$m(t) = a\left(1 - qe^{-vt} - (1-q)e^{\left(\frac{2c\left(1-e^{-vt}\right)}{v} - tc\left(1+e^{-vt}\right)\right)}\right) \quad (18)$$

3. If $\Delta t = \frac{1}{v}\ln\left(1+vt+\frac{v^2t^2}{2}\right)$ then,

$$m(t) = a\left(1 - qe^{-vt} - (1-q)e^{\left(\frac{3c\left(1-(1+vt)e^{-vt}\right)}{v} - tc\left(1-\left(1-\frac{vt}{2}\right)e^{-vt}\right)\right)}\right) \quad (19)$$

Similarly, $m(t)$ for other SRGMs can be derived [18].

V Parameter Estimation and Comparison Criteria

Model selection is followed by estimation of its parameters. Parameters of a model can be estimated by Maximum Likelihood method, Least Squares or by Non Linear Regression Module of Statistical Package for Social Sciences (SPSS). After estimation, models are quantitatively evaluated to check their effectiveness. Various criteria used are Mean Square error, Bias, Predictive Ratio Risk, Sum of Squared
Errors, Noise, Akaike Information Criteria and so on[17].

VI CONCLUSION

For ensuring complete reliability so as to have quality of caliber, both detection and correction need to go in parallel. If not, then reliability of a system will be understated. The factors like time delay and dependency of faults adds to it. Considering the fault dependency and the delay factor overall improves the detection and correction process, thus increasing the reliability. In this paper, perfect debugging has been taken into consideration. Presuming that new faults get introduced while rectifying an observed fault is known as imperfect debugging. When new faults get injected, it leads to change in testing environment. This change is called as change point. It is proposed to extend the approach by integrating delay, fault severity, imperfect debugging and change point into a single model. To ease the estimation process, parameters may be estimated by Statistical Package for Social Sciences. This model may be validated by closed and open data sets and a comparison between the two may also be presented.

REFERENCES

[1] R.Peng and F.R.Shahrzad,Simulation of Software Fault Detection and Correction Processes Considering Different Skill Levels of Debugger, *IEEE symposium*,2014,

[2] N.F.Schneidewind, Modelling the Fault Correction Process, in the proceedings of the 12th International Symposium on Software Reliability Engineering, IEEE Computer Society press: Loss Alarmitcs, CA, 2001,185-190.

[3] N.F.Schneidewind, Analysis of Error Process in Computer Software, *Sigplan Notices*, 1975; *10(6):* 337-346.

[4] Lo and Huang ,An integration of fault detection and correction process in software reliability analysis, *Journal of System and Software*, *79*, 2006,1312-1323.

[5] M.Xie, Q.P.Hu , Y.P.Wu and S.H.Ng ,A Study of the Modeling and Analysis of Software Fault detection and Fault Correction Processes, *Quality and Reliability Engineering International*, 2007, *23*: 459-470.

[6] P.K.Kapur, PC Jha, D.Gupta and K. Yadav, Identification of Different Stages in the Testing Phase of a Software Reliability Growth Model, in the proceedings of Advances in Performance and safety of complex systems (Eds A.K.Verma, P.K.Kapur and S.G. Ghadge) MacMillan India Ltd,2008,850-861.

[7] L.Jia, B.Yang, S.Guo and D. H. Park, Software Reliability Modeling Considering Fault Correction Process, *IEICE*,*E93-D(1)*,2010.

[8] P. K. Kapur, H. Pham, Fellow, IEEE, S.Anand, and K.Yadav, A Unified Approach for Developing Software Reliability Growth Models in the Presence of Imperfect Debugging and Error Generation, *IEEE Transactions on Reliability*,2011, *60(1)*.

[9] V.Kumar and R. Sahni, An effort allocation model considering different budgetary constraint on fault detection process and fault correction process, *Decision Science Letters*, 2016,143-156.

[10] Chin-Yu Huang and Chu-Tin Lin, Software Reliability Analysis by Considering Fault Dependency and Debugging Time Lag,*IEEE Transaction on Reliability*, *55(3)*,2006, 436-450.

[11] M.Ohba, Software Reliability Analysis Models by Ohba,*IBM J. RES. Develop*, *28(4)*,1984.

[12] P.K.Kapur and S.Younes,Software Reliability Growth Model with Error Dependency, *Microelectronics and Reliability*,*35(2)*,1995, 273-278.

[13] V.B.Singh, K.Yadav, R.Kapur and V.S.S Yadavalli,Considering Fault Dependency Concept with Debugging Time Lag in Software Reliability Growth Modeling Using a Power Function of Testing Time, *International Journal of Automation and Computing*, *4(4)*, 2007,359-368.

[14] R. Peng and Q. Zhai, Modeling of software fault detection and correction processes with fault dependency, *Science and Technology*, *Maintenance and Reliability*, *19(3)*,2017.

[15] C.Y.Huang, C.T.Lin, C.C.Sue,Considering Fault Dependency and Debugging Time Lag in Reliability Growth Modeling During Software Testing, IEEE 13th Asian Symposium, 2004.

[16] Dr. A.Gupta,Dr. D.Choudhary and Dr. S. Saxena, Software Reliability Estimation using Yamada Delayed S Shaped Model under Imperfect Debugging and Time Lag, *International Journal of Computer Applications(0975-8887)*,*23(7)*, 2011.

[17] K.Sharma,R.Garg,C.K.Nagpal and R.K.Garg ,Selection of Optimal Software Reliability Growth Models Using a Distance Based Approach, *IEEE Transactions on Reliability*, *59(2)*, 2010.

[18] P.K.Kapur, V.B.Singh and Sameer Anand, Fault Dependency Based Software Reliability Growth Modeling with Debugging Time Lag Functions, *Communications in Dependability and Quality Management anInternational Journal*,*Serbia*,*10(3)*,2007,46-48