Assessing Pareto Type II Software Reliability using SPC

R.Satya Prasad
Associate Professor
Department of CSE
Acharya Nagarjuna University
Guntur, Andhra Pradesh,
India.

G. Sridevi
Associate Professor
Department of CSE
Nimra Women's College of Engg
Vijayawada, Andhra Pradesh,
India.

K.Sita Kumari
Associate Professor
Department of IT
V.R.Siddhartha Engg. College
Kanuru, Vijayawada, A.P.
India.

ABSTRACT

Software reliability is one of the most important characteristics of software quality. Its measurement and management technologies employed during software life cycle are essential for producing and maintaining quality/reliable software systems. It can also be monitored efficiently using Statistical Process Control (SPC). It assists the software development team to identify and actions to be taken during software failure process and hence, assures better software reliability. In this paper we propose a control mechanism based on the cumulative observations of Interval domain data using mean value function of Pareto type II distribution, which is based on Non-Homogenous Poisson Process (NHPP). The maximum likelihood estimation approach is used to estimate the unknown parameters of the model. We also present an analysis of failure data sets at a particular point.

General Terms

Software failure data, Mean value function.

Keywords

Software Reliability, NHPP, Pareto type II distribution, Parameter Estimation, Interval Domain data, ML Estimation, Statistical Process Control, Mean value function, Control charts.

1. INTRODUCTION

Software Reliability is the most dynamic quality characteristic which can measure and predict the operational quality of the software system during its intended life cycle. Software Reliability is the probability of failure free operation of software in a specified environment during specified duration [Musa 1998], Wood[1996], Satya Prasad[2007]. To identify and eliminate human errors in software development process and also to improve software reliability, the Statistical Process Control concepts and methods are the best choice.

The NHPP based models are the most important models because of their simplicity, convenience and compatibility. The NHPP based software reliability growth models are proved quite successful in practical software reliability engineering [Musa et al., 1987].

Here SPC concepts and methods are used to monitor the performance of a software process over time in order to verify that the process remains in the state of Statistical control. It helps in finding assignable causes, long term improvements in the software process. Software quality and reliability can be achieved by eliminating the causes or improving the software process or its operating procedures.

The main issue in 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 estimate (MLE).Parameter values can be obtained using Newton Raphson Method. SPC is a powerful tool to optimize the amount of information needed for use in making management decisions. Statistical techniques provide an understanding of the business baselines, insides for process improvements, communication of value and results of processes, and active and visible involvement.

The most popular technique for maintaining process control is control charting. The control chart is one of the seven tools for quality control. Software process control is used to secure, that the quality of the final product will conform to predefined standards.

A process is said to be statistically "in-control" when it operates with only chance causes of variation. On the other hand, when assignable causes are present, then we say that the process is statistically "out-of-control".

SPC provides a real time analysis to establish controllable process baselines; learn, set and dynamically improve process capabilities; and focus business areas needing improvement. The early detection of software failures will improve the software reliability. The selection of proper SPC charts is essential to effective statistical process control implementation and use. The SPC chart selection is based on data, situation and need.

This paper presents Pareto type II model to analyse the reliability of a software system using interval domain data. The layout of the paper is as follows: Section2 describes the formulation and interpretation of the model for the underlying NHPP, Section 3 describes the proposed Pareto type II software reliability growth model., Section 4 discusses parameter estimation of Pareto type II model based on interval domain data. Section 5 describes the techniques used for software failure data analysis for a live data and

Section 6 Conclusion.

2. MODEL FORMULATION

There are numerous software reliability growth models available for use according to probabilistic assumptions. The Non Homogenous Poisson Process (NHPP) based software reliability growth models are proved to be quite successful in practical software reliability engineering. NHPP model formulation is described in the following lines.

A software system is subjected to failures at random times caused by errors present in the system.

Let$\{N(t), t > 0\}$be a counting process representing the cumulative number of failures by time t. Since there are no failures at t=0 we have

$$N(0) = 0$$

It is reasonable to assume that the number of software failures during non-overlapping time intervals do not affect each other. In other words, for any finite collection of times $t_1 < t_2 < \cdots t_n$. The n random variables $(t_1), \{N(t_2) - N(t_1)\}, \ldots \{N(t_n - N(t_{n-1}))\}$ are independent. This implies that the counting process {N(t), t>0} has independent increments.

Let m(t) represent the expected number of software failures by time's'. The mean value function m(t) is finite valued, non-decreasing, non-negative and bounded with the boundary conditions.

$$m(t) = 0, t = 0$$
$$= a,\ t \to \infty$$

Where a is the expected number of software errors to be eventually detected.

Suppose N (t) is known to have a Poisson probability mass function with parameters m(t) i.e.,

$$P\{N(t) = n\} = \frac{[m(t)]^n . e^{-m(t)}}{n!}, n = 0,1,2 \ldots \infty$$

Then N(t) is called an NHPP. Thus the stochastic behaviour of software failure phenomena can be described through the N(t) process. Various time domain models have appeared in the literature (Kantam and Subbarao, 2009) which describe the stochastic failure process by an NHPP which differ in the mean value functions m (t).

3. THE PROPOSED PARETO TYPE II SRGM

In this paper we consider m(t) as given by

$$m(t) \ = a\,[1 \ - \frac{c^b}{(t+c)^b}] \qquad (3.1)$$

Where [m (t)/a] is the cumulative distribution function of Pareto type II distribution (Johnson et al, 2004) for the present choice.

$$P\{N(t) = n\} = \frac{[m(t)]^n . e^{-m(t)}}{n!}$$

$$\lim_{n\to\infty} P\,\{N(t) = n\} = \frac{e^{-a}.a^n}{n!}$$

This is also a Poisson model with mean 'a'.

Let N (t) be the number of errors remaining in the system at time't'

$$N(t) = N(\infty) - N(t)$$
$$E[N(t)] \ = E[N(\infty)] - E[N(t)]$$
$$= a - m(t)$$
$$= a - \ a\left[1 - \frac{c^b}{(t+c)^b}\right]$$
$$= \ \frac{ac^b}{(t+c)^b}$$

4 PARAMETER ESTIMATION BASED ON INTERVAL DOMAIN DATA

In this section we develop expressions to estimate the parameters of the Pareto type II model based on interval domain data. Parameter estimation is of primary importance in software reliability prediction.

A set of failure data is usually collected in one of two common ways, time domain data and interval domain data. In this paper parameters are estimated from the interval domain data.

The mean value function of Pareto type II model is given by

$$m(t) = a\left[1 - \frac{c^b}{(t+c)^b}\right], \qquad t \geq 0 \qquad (4.1)$$

In order to have an assessment of the software reliability, a, b and c are to be known or they are to be estimated from software failure data. Expressions are now delivered for estimating 'a', 'b' and 'c' for the Pareto type II model.

Assuming the given data are given for the cumulative number of detected errors ni in a given time interval (0, ti) where i=1,2, ….. n and 0 < t1< t2< …tn, then the logarithmic likelihood function (LLF) for interval domain data [8] is given by

$$Log\ L = \sum_{i=1}^{k}(n_i - \ n_{i-1}).\log[m(t_i) - m(t_{i-1})] - m(t_k) \qquad (4.2)$$

$$Log\ L = \ \sum_{i=1}^{k}(n_i - \ n_{i-1}).\log\left[a\ \left[1 - \frac{c^b}{(t_i + c\,)^b}\right] - a\left[1 - \frac{c^b}{(t_{i-1} + c)^b}\right]\right] - a\left[1 - \frac{c^b}{(t_k \ + c\,)^b}\right]$$

(4.3)

$$\log L = \sum_{i=1}^{k}(n_i \ - n_{i-1}\,)[\log a + b\log c + \log[(t_i \ + c)^b \ - (t_{i-1} \ + c)^b] - b\log(t_{i-1} \ + c) - b\log(t_i \ + c)] - a + \frac{ac^b}{(t_k + c)^b}$$

(4.4)

Accordingly parameters 'a','b' and 'c' would be solutions of the equations

$$\frac{\partial \log L}{\partial a} = 0$$

$$a = \sum_{i=1}^{k}(n_i - n_{i-1}) \ \frac{(t_k + c)^b}{(t_k \ + c)^b \ - c^b} \qquad (4.5)$$

The parameter ‘b’ is estimated by iterative Newton Raphson Method using

$$b_{n+1} = b_n - \frac{g(b_{n)}}{g'(_{bn)}}$$, where g(b) and g’(b) are expressed as follows.

$$g(b) = \frac{\partial \log L}{\partial b} = 0$$

$$g(b) = \sum_{i=1}^{k} (n_i - n_{i-1}) \left[-\log(t_{i-1} + 1) - \log(t_i + 1) + \frac{(t_i + 1)^b \log(t_i + 1) - (t_{i-1} + 1)^b \log(t_{i-1} + 1)}{(t_i + 1)^b - (t_{i-1} + 1)^b} \right] + \sum_{i=1}^{k} (n_i - n_{i-1}) \log \frac{1}{(t_k + 1)} \cdot \frac{1}{(t_k + 1)^b - 1} = 0$$

(4.6)

$$g^1\ (b) = \frac{\partial^2 \operatorname{Log} L}{\partial b^2} = 0$$

$$g^1(b) = \sum_{i=1}^{k} (n_{i-}\ n_{i-1}). \frac{2\,(t_{i-1}+1)^b (t_{i+1})^b \log(t_i+1) \log\frac{(t_{i-1}+1)}{(t_i+1)}}{\left[(t_i+1)^b - (t_{i-1}+1)^b\right]^2} + \sum_{i=}^{k} (n_i - n_{i-1}) \log(t_k + 1) \left[\frac{(t_k + 1)^b \log(t_k + 1)}{[(t_k + 1)^b - 1]^2} \right]$$

(4.7)

The parameter ‘c’ is estimated by iterative Newton Raphson Method using

$$c_{n+1} = C_n - \frac{g(c_{n)}}{g'(c_{n)}}$$

Where g (c) and g1(c) are expressed as follows.

$$g(c) = \frac{\partial LogL}{\partial c} = 0$$

$$g\ (c) = \sum_{i=1}^{k} (n_i - n_{i-1}) \left[\frac{1}{c} - \frac{1}{(t_{i-1} + c)} - \frac{1}{(t_i + c)} \right] + \sum_{i=1}^{k} (n_i - n_{i-1}) \frac{1}{(t_k + c)} = 0 \qquad (4.8)$$

$$g'(c) = \frac{\partial^2 LogL}{\partial c^2} = 0$$

$$g'(c) = \sum_{i=1}^{k} (n_i - n_{i-1}) \left[-\frac{1}{c^2} + \frac{1}{(t_{i-1}+c)^2} + \frac{1}{(t_i + c)^2} \right] - \sum_{i=1}^{k} (n_i - n_{i-1}) \frac{1}{(t_k+c)^2} \qquad (4.9)$$

The values of ‘b’ and ‘c’ in the above equations can be obtained using Newton Raphson Method. Solving the above equations, simultaneously yields the point estimates of the parameters b and c. These equations are to be solved iteratively and their solutions in turn when substituted in equation (4.5) gives value of ‘a’.

5. DATA ANALYSIS

In this section, we present the analysis of one software failure data set. The set of software errors analysed here is borrowed from a real software development project as published in Pham (2005), which in turn referred to Pham(2005) as Zhang et al.,(2000).The data are named as Release 2 test data. The Release 2 test data is summarized in the below table.

Table 1 - Release 2 Test Data

Test Week	CPU Hours	Percent CPU Hours	Defects found	Predicted total defects
1	384	-	13	-
2	1186	-	18	-
3	1471	-	26	-
4	2236	-	34	-
5	2772	-	40	-
6	2967	-	48	-
7	3812	-	61	-
8	4880	-	75	-
9	6104	-	84	-
10	6634	65	89	203
11	7229	70	95	192
12	8072	79	100	179
13	8484	83	104	178
14	8847	86	110	184
15	9253	90	112	184
16	9712	95	114	183
17	10083	98	117	182
18	10174	99	118	183
19	10272	100	120	184
20	-	-	-	-

Solving equations in section 4 by Newton Raphson Method(N-R) method for the Release 2 test data, the iterative solutions for MLEs of a, b and c are

$$a = 158.1535$$

$$b = 0.977674$$

$$c = 8.74581$$

Hence, we may accept these three values as MLEs of a, b, c. Using these we can compute $m(t)$. Now the control limits are calculated by the following equations taking the standard values 0.00135, 0.99865 and 0.5.

Table 2. Successive differences of Cumulative mean values

Week	Cumulative Failures	M(t)	Successive Differences
1	13	93.24004	11.89084
2	18	105.1309	11.96898
3	26	117.0999	7.528576
4	34	124.6284	4.040184
5	40	128.6686	4.070694
6	48	132.7393	4.641541
7	61	137.3809	3.401823
8	75	140.7827	1.649872
9	84	142.4325	0.786680
10	89	143.2192	0.844979
11	95	144.0642	0.633677
12	100	144.6979	0.466909
13	104	145.1648	0.642012
14	110	145.8068	0.199979
15	112	146.0068	0.193534
16	114	146.2003	0.278884
17	117	146.4792	0.090060
18	118	146.5693	0.175969
19	120	146.7452	-----

$$T_U = \left(1 - \frac{c^b}{(t+c)^b}\right) = 0.99865$$

$$T_C = \left(1 - \frac{c^b}{(t+c)^b}\right) = 0.5$$

$$T_L = \left(1 - \frac{c^b}{(t+c)^b}\right) = 0.00135$$

These limits are converted to $m(t_U), m(t_C)$ and $m(t_L)$ form. They are used to find whether the software process is in control or not by placing the points in Mean value chart shown in figure 1.

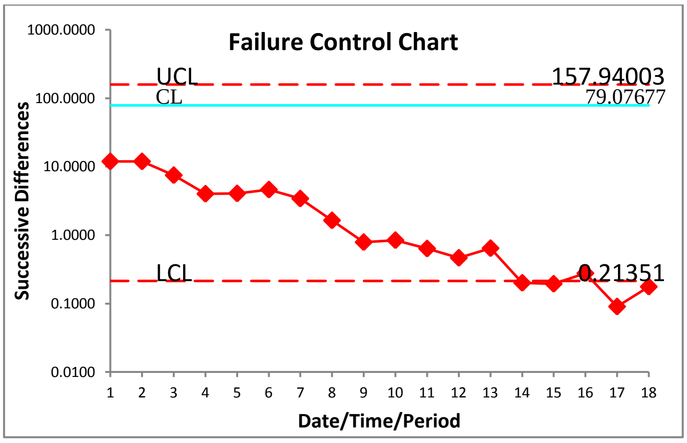

Figure 1. Mean Value chart

A point below the control limit $m(t_L)$ indicates an alarming signal. A point above the control limit $m(t_U)$ indicates better quality. If the points are falling within the control limits it indicates the software process is in stable. The values of control limits are as follows.

$$m(t_U) = 157.940029$$

$$m(t_C) = 79.076768$$

$$m(t_L) = 0.213507$$

By placing the successive differences shown in table 2 on y axis and failure week on x axis and the values of control limits are placed on Mean Value chart, we obtained Figure 1.The Mean Value chart shows that all the successive differences after the 13 th failure data has fallen below $m(t_L)$ which indicates the failure process is identified. It is significantly early detection of failures using Mean Value chart.

6. CONCLUSION

Software Reliability is an important quality measure that quantifies the operational profile of computer systems. In this paper we proposed Pareto type II software reliability growth model. This model is primarily useful in estimating and monitoring software reliability, which is viewed as a measure of software quality. Equations to obtain the maximum likelihood estimates of the parameters based on interval domain data are developed.

This analysis shows that the Release 2 data (Figure 1) shows out of control signals i.e., below the LCL. We conclude that our method of estimation and the control chart are giving a +ve recommendation for their use in finding out preferable control process or desirable out of control signal.

By observing the Mean Value Control chart (Figure 1) we identified that the failure situation is detected at 14th point of Table-2. Hence our proposed Mean Value Chart detects out of control situation. This is a simple method for model validation and is very convenient for practitioners of software reliability.

The early detection of software failure will improve the software reliability. The methodology adopted in this paper is better than the methodology adopted by Xie et al[2002]. Therefore, we may conclude that this model is the best choice for an early detection of software failures.

7. ACKNOWLEDGEMENTS

Our thanks to Department of Computer Science and Engineering, Acharya Nagarjuna University for providing necessary details to carry out the research work.

8. REFERENCES

[1] Goel, A.L., Okumoto, K., 1979. Time-dependent error detection rate model for software reliability and other performance measures. IEEE Trans. Reliab. R-28, 206-211.

[2] Musa J.D, Software Reliability Engineering MCGraw-Hill, 1998.

[3] Pham .H (2005) “A Generalized logistic Software Reliability Growth Model”, Opsearch, Vol. 42, No. 4, 332-331.MCGraw-Hill, New York.

[4] Musa J.D., Iannino, A., Okumoto, K., 1987. Software Reliability: Measurement Prediction Application. MC Graw Hill, New York.

[5] Wood, A(1996), “Predicting software Reliability”, IEEE Computer,2253-2264.

[6] Satya Prasad R and Geetha Rani N (2011), “Pareto type II Software Reliability Growth Model”. International Journal of Software Engineering, Volume 2, Issue(4) 81-86.

[7] Satya Prasad R (2007) “Half Logistic Software Reliability Growth Model”, Ph.D Thesis of ANU, India.

[8] Pham, H., “System Software Reliability”, Springer 2006.

[9] Satya Prasad R and Goutham V (2011), “Interval Domain Software Process Control-GoelOkumoto”, International Journal of Research and Reviews in Computer Science, Volume 2, Issue (4) , ISSN:2079-2557

[10] Xie. M, T.N Goh and P.Rajan . “Some effective control chart procedures for reliability monitoring”, Reliability engineering and system safety, 2002. 77, 143-150.

9. AUTHORS PROFILE

Dr.R Satya Prasad received Ph.D.degree in computer science in the faculty of Engineering in 2007 from Acharya Nagarjuna University, Andhra Pradesh, India. He received gold medal from Acharya Nagarjuna University for his outstanding performance in master’s degree. He is currently working as Associate Professor in the department of Computer Science & Engineering, Acharya Nagarjuna University. He has occupied various Academic responsibilities like practical examiner, project adjudicator, external member of board of examiners for various universities and colleges in and around in Andhra Pradesh. His current research is focused on Software engineering, Image processing & Database Management system. He has published several papers in National & International Journals.

Mrs. G. Sridevi received M.Sc. and M.Tech degree from Acharya Nagarjuna University. She is currently pursing Ph.D at Department of Computer Science and Engineering, Acharya Nagarjuna University, Andhra Pradesh, India. She is currently working as a Vice-Principal and Associate professor in the Department of Computer Science, Nimra Women’s College of Engineering, Jupudi, Ibrahimpatnam, Vijayawada, Andhra Pradesh. Her research interests lies in Data Mining and Software Engineering.

Mrs. K.Sita kumari received M.Sc degree from Acharya Nagarjuna University, Guntur and M.Tech degree from Dr.MGR University, Chennai. She is currently pursuing her Ph.D from Department of Computer Science and Engineering, Acharya Nagarjuna University, Andhra Pradesh, India. Presently she is working as an Associate Professor in the department of Information Technology, V.R.Siddhartha Engineering College, Kanuru, Vijayawada.