Software Reliability Using Growth Models

[a]B.Indira, [b]V Aruna
[a]Professor, Dpt of IT,Sreenidhi Institute of Science & Technology,Ghatkesar, India
[b]Professor, Dpt of CSE, Sreenidhi Institute of Science & Technology, Ghakesar, India

Abstract

The Software Reliabilty of a system is a safety-critical sytem, fractionth error may result in hazards.In this paper a practice to study some problems for assessing software reliability using Non-Homogeneous Poisson Process (NHPP) based Log power model as mean value function is done. Control mechanisms to assess whether process is under control or not is observed. Order Statistics are used to detect outliers.Sequential Probability Ratio Test is used for continuous monitoring of the data with an aim to monitor the failure process and detect any change of the intensity parameter.

KEYWORDS— Statistical Process Control, Order statistics, Sequential Probability Ratio Test (SPRT)

I. Introduction

Software Reliability is the probability of failure free software operation for a specified period of time in a specified environment. Software Reliability is also an important factor affecting system reliability.The overall reliability of a system depends on the hardware reliability, the software reliability and the reliability of the system operators. To achieve some required level of reliability, Software reliability assessment is important to evaluate and predict the reliability and performance of software system. 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. Software quality and reliability can be achieved by eliminating the causes or improving the software process or its operating procedures. 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 the quality of the final product which will conform to predefined standards. In any process, regardless of how carefully it is maintained, a certain amount of natural variability will always exist.

Control charts can be classified into several categories, according to several distinct criteria. Control charts should be capable to create an alarm when a shift in the level of one or more parameters of the underlying distribution or a non-random behavior occurs. 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, insights for process improvements, communication of value and results of processes, and active and visible involvement. 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.

The control limits for the chart are defined in such a manner that the process is considered to be out of control when the time to observe exactly one failure is less than LCL or greater than UCL. Our aim is to monitor the failure process and detect any change of the intensity parameter. When the process is in control, there is a chance for this to happen and it is commonly known as false alarm. The traditional false alarm probability is to set to be 0.27% although any other false alarm probability can be used. The actual acceptable false alarm probability should in fact depend on the actual product or process (Xie et al, 2002).

Software reliability is the application of statistical techniques to data collected during system development and operation to state, forecast, estimate, and assess the reliability of software-based systems (DAC, 2006). The research in software reliability modelling has been developing for over four decades. Software reliability modelling is to describe fault-related behaviours of software testing process. Many models have been developed with different mathematical techniques to adapt to different testing environments (Lyu, 1996). The reliability measurements are crucial for the management to make decisions, such as testing-resource allocation (Yamada *et al*., 1995), test stopping decision (Littlewood and Wright, 1997) and cost analysis (Xie and Yang, 2003). Generally these models can be categorized into two groups: data-driven software reliability models and analytical software reliability models (Gokhale and Trivedi, 1999).

Parameter estimation is of primary importance in software reliability prediction. Once the analytical solution for $m(t)$ is known for a given model, parameter estimation is achieved by applying a technique of Maximum Likelihood Estimate (MLE). The idea behind maximum likelihood parameter estimation is to determine the parameters that maximize the probability (likelihood) of the sample data. The method of maximum likelihood is considered to be more robust and yields estimators with good statistical properties. Assuming that the data are given for the occurrence times of the failures or the times of successive failures, s_j for $j = 1, 2, ..., n.$ Given that the data provide n successive times of observed failures s_j for $0 \ s_1 \le s_2 \le \cdots \le s_n$, we can convert these data into the time between failures t_i where $t_i = s_i - s_{i-1}$ for $i = 1, 2, ..., n.$ Given the recorded data on the time of failures, the Log Likelihood Function (LLF) takes on the following form (Pham, 2006):

$$LLF = \sum_{i=1}^{n} \log[\lambda(t_i)] - m(t_n)$$

'$\hat{a}$' and '$\hat{b}$' are Maximum Likely hood Estimates (MLEs) of parameters and the values can be computed using iterative method for the given cumulative time between failures data. Using 'a' and 'b' values we can compute $m(t)$. Now the control limits are calculated by equating the cumulative distribution function to the standard values 0.00135, 0.99865, and 0.5.

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 failure control 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.

II. SPC ON UNGROUPED DATA

Software reliability is the probability of failure-free operation of software in a specified environment during specified time duration. Relatively little research work is, however, available on their use to monitor failure process of software. The control scheme adopted for our study is based on the inter failure cumulative data between observations of failure. It is proposed that the said control scheme can be easily and fruitfully applied to monitor the software failure process for Log Power model based Non-Homogeneous Poisson Process (NHPP).

The monitoring of Software reliability process is a far from simple activity. In recent years, several authors have recommended the use of SPC for software process monitoring. The main thrust of the thesis is to formalize and present an array of guidelines in a disciplined process with a view to helping the practitioner in putting SPC to correct use during software process monitoring.

Our effort is to apply SPC in the software development process so as to improve software reliability and quality. It is reported that SPC can be successfully applied to several processes for software development, including software reliability process. The utilization of SPC for software reliability has been the subject of study of several researchers. A few of these studies are based on reliability process improvement models.

It is accepted on all hands that SPC acts as a powerful tool for bringing about improvement of quality and is particularly relevant to software development also. Applying SPC for assessing software reliability with Log Power model based on NHPP, is the main aim of this chapter. The Maximum Likelihood Estimation (MLE) method is used for parameter estimation.

III. SPC- AN ORDER STATISTICS APPROACH

There are many software reliability models that are based on the times of occurrences of errors in the debugging of software. It is shown that it is possible to do asymptotic likelihood inference for software reliability models based on order statistics or non-homogeneous Poisson processes, with asymptotic confidence levels for interval estimates of parameters. In particular, interval estimates from these models are obtained for the conditional failure rate of the software, given the data from the debugging process. The data can be grouped or ungrouped. For someone making a decision about when to market software, the conditional failure rate is an important parameter.

Order statistics are used in a wide variety of practical situations. Their use in characterization problems, detection of outliers, linear estimation, study of system reliability, life-testing, survival analysis, data compression and many other fields can be seen from the many books on the topics, e.g., the recent books (Bala Krishnan.N. *et al.,* 1991),(Arnold.B.C. *et al.,* 1989)

In this chapter we propose a control mechanism based on order statistics of cumulative quantity between observations of time domain failure data using mean value function of Log Power model based on NHPP.

Order statistics deals with properties and applications of ordered random variables and of functions of these variables. The use of order statistics is significant when failures are frequent or inter failure time is less. Let 'A' denote a continuous random variable with probability density function (pdf) f(x) and cumulative distribution function (cdf) F(x), and let (A_1 , A_2 , …, A_n) denote a random sample of size n drawn on 'A'. The original sample observations may be unordered with respect to magnitude. A transformation is required to produce a corresponding ordered sample. Let ($A_{(1)}$, $A_{(2)}$, …, $A_{(n)}$) denote the ordered random sample such that $A_{(1)} < A_{(2)} < \ldots < A_{(n)}$; then ($A_{(1)}$, $A_{(2)}$, …, $A_{(n)}$) are collectively known as the order statistics derived from the parent of A. The various distributional characteristics can be known from Balakrishnan and Cohen model (1991).

The inter-failure time data represent the time lapse between every two consecutive failures. On the other hand if a reasonable waiting time for failures is not a serious problem, we can group the inter-failure time data into non overlapping successive sub groups of size 4 or 5 and add the failure times with in each sub group. For instance if a data of 100 inter-failure times are available we can group them into 20 disjoint subgroups of size 5. The sum total in each subgroup would denote the time lapse between every 5^{th} failure and it would be 5^{th} order statistic in a sample of size 5.

In general for inter-failure data of size 'n', if r (any natural number less than 'n') and preferably a factor of n, we can conveniently divide the data into 'k' disjoint subgroups (k=n/r) and the cumulative total in each subgroup indicate the time between every r^{th} failure. The probability distribution of such a time lapse would be that of the r^{th} ordered statistic in a subgroup of size r, which would be equal to r^{th} power of the distribution function of the original variable m(t) (David. H. A. *et al.,* 2003).

The whole process involves the mathematical model of the mean value function and knowledge about its parameters. If the parameters are known they can be taken as they are, for the further analysis. If the parameters are not known they have to be estimated using a sample data by any admissible, efficient method of estimation. This is essential because the control limits depend on mean value function, which inturn depends on the parameters. If software failures are quite frequent keeping track of inter-failures is a tedacious task. If failures are more frequent order statistics are preferable.

IV. SPRT: LOG-POWER MODEL

In Classical Hypothesis testing volumes of data is to be collected and then the conclusions are drawn, which may need more time. But, Sequential Analysis of Statistical science could be adopted in order to decide upon the reliability / unreliability of the developed software very quickly. The procedure adopted for this is, Sequential Probability Ratio Test (SPRT). It is designed for continuous monitoring. The likelihood based SPRT proposed by Wald is very general and it can be used for many different probability distributions.

Sequential Probability Ratio Test (SPRT) is an ongoing statistical analysis repeatedly conducted as data is collected. It is used in anomaly detection and decision making for electronics, structures and process controls. The data are repeatedly reassessed and a

decision is made either to: Reject the null hypothesis and stop collecting data, Fail to reject the null hypothesis and stop collecting data or Continue collecting data until a decision regarding the null hypothesis can be reached. The SPRT sets threshold boundaries, which take the form of parallel lines, one of which represents the expected outcome and the other a significantly different outcome. When the value of the calculated test statistic falls outside of these threshold boundaries, a conclusion can be drawn and data collection stops. The parameters are estimated using Maximum Likelihood Estimation method. In the present Chapter, the Log Power model is applied on five sets of existing software reliability data and analyzed the results.

The respective brief contents of these three problems are given in the “introduction”. The numerical calculations and subsequent tables are provided at appropriate places in the respective chapters. The reprints of some of our findings in published form are appended towards the end of the thesis. List of references arranged alphabetically is also provided towards the end of the thesis.

IV. CONCLUSION

Sequential Probability Ratio Test (SPRT) is an ongoing statistical analysis repeatedly conducted as data is collected. It is used in anomaly detection and decision making for electronics, structures and process controls. The data are repeatedly reassessed and a decision is made either to: Reject the null hypothesis and stop collecting data, Fail to reject the null hypothesis and stop collecting data or Continue collecting data until a decision regarding the null hypothesis can be reached. When the value of the calculated test statistic falls outside of these threshold boundaries, a conclusion can be drawn and data collection stops. The parameters are estimated using Maximum Likelihood Estimation method. Therefore, we may conclude that, applying SPRT on data sets we can come to an early conclusion of reliability / unreliability of software.

REFERENCES

1. Arnold B.C., And Balakrisha N.,(1989), “Relations, Bounds And Approximations For Order Statistics”, Lecture Notes In Statistics No-53, Springer-Verlag.
2. Balakrishna N., And Cohen A.C.,(1991), “ Order Statistics And Inference: Estimation Methods”, Academic Press.
3. Dac,(2006). Http://Www.Thedacs.Com/Databases.
4. David, H.A. And Nagaraja, H.N., (2003). “Order Statistics”, 3rd Edition, John Wiley & Sons.27-32.
5. Gokhale, S.S And Trivedi, K.S., (1999). “A Time/Structure Based Software Reliability Model”, Annals Of Software Engineering, 8, Pp.85-121.
6. Littlewood, B. And Wright, D. (1997). “Some Conservative Stopping Rules For The Operational Testing Of Safety-Critical Software”, Ieee Transactions On Software Engineering 23(11), 673–683.
7. Lyu. M.R., (1996). “Handbook Of Software Reliability Engineering”, Data Dictionary Ieee Computer Society And Mcgraw-Hill, New York. Www.Cse.Cuhk.Edu/~Lyu/Book/Reliability/Data/

8. Pham. H., (2006). “System Software Reliability”, Springer.
9. Xie, M., Goh, T. N. And Ranjan.P., (2002). “Some Effective Control Chart Procedures For Reliability Monitoring”, Reliability Engineering And System Safety, 77, 143 -150.
10. Xie, M. And Yang, B., (2003). “A Study Of The Effect Of Imperfect Debugging On Software Development Cost”, Ieee Trans. Software Engineering, 29(5), 471-473.
11. Yamada, S., Ichimori, T. And Nishiwaki, M. (1995). “Optimal Allocation Policies For Testing-Resource Based On A Software Reliability Growth Model”, Mathematical And Computer Modelling, 22, 259-301.
12. Software Engineering,By Ian Sommerville. Ninth Edition