ISSN: 0975-833X

Available online at http://www.journalcra.com

International Journal of Current Research
Vol. 5, Issue, 06, pp.1505-1507, June, 2013

INTERNATIONAL JOURNAL OF CURRENT RESEARCH

RESEARCH ARTICLE

USE OF EVIDENCE THEORY FOR CALCULATING SOFTWARE RELIABILITY

***Ravichandran M. and Ramani A. V.**

Department of Computer Science Sri Ramakrishna Mission Vidyalaya College of Arts and Science Coimbatore, India

ARTICLE INFO

Article History:
Received 03rd March, 2013
Received in revised form
11th April, 2013
Accepted 14th May, 2013
Published online 15th June, 2013

Key words:
Software Reliability, Theory of Evidence, Belief measure and plausibility measure.

ABSTRACT

The goal of software Engineering is to produce high quality software. Software reliability is one of the most important factors in software quality evaluation. Now the applications are developed in web oriented environment. Since these applications are developed in different softwares and in different platforms, software reliability measurement plays an important role in software quality measurement. This paper discusses using evidence theory for evaluation of reliability in web oriented software applications.

INTRODUCTION

The IEEE defines reliability as "The ability of a system or component to perform its required functions under stated conditions for a specified period of time". The reliability is equated to correctness to most project and software development managers. The reliability of the delivered code is related to the quality of all of the processes and products of software development, the requirements documentation, the code, test plans, and testing. Software reliability is not as well defined as hardware reliability [8]. Software reliability is defined as the probability of failure-free operation of a computer program for a specified time in a specified environment. A common approach for measuring software reliability is by using an analytical model whose parameters are generally estimated from available data on software failures [11]. Hence, software reliability is a key factor in software development process. Software reliability growth model (SRGM) is a mathematical expression of the software error occurrence and the removal process. Since the early 1970's, many SRGMs have been proposed [7] and [10]. A non homogeneous Poisson process (NHPP) as the stochastic process has been widely used in SRGM. In the past years, several SRGMs based on NHPP which incorporates the testing effort function (TEF) have been proposed by many authors [11]. Software reliability evaluation is playing an important role in software reliability engineering, which can give information taken as the reference or accordance to guide the software's design, analysis and testing and so on. It will provide the quantitative estimation result for the issued software product. In recent years, software reliability evaluation based on failure data has been deeply developed, as the main means of software reliability estimation, lots of software reliability growth models have been proposed [9]. But with the shortcoming of not very good evaluation quality, many new models and technique were proposed to effectively improve the reliability estimation performance, such as Neural-Network-based model presented by N.Karunanithi, chaos deduce model, Bayes networks model, fuzzy theory model and so on. New technologies are also proposed, such as the failure data trend analysis and prediction quality improvement [9].

Corresponding author: ravichandransrkv@gmail.com

Literature Review

Musa [10] defined software reliability as the probability of failure-free operation of a software component or system in a specified environment for a specified time. A failure is defined as an unacceptable departure of program operation from requirements. A fault is the software defect that causes a failure. The term error is used to indicate human actions that results in a fault. Goel [3] classified a number of analytical models proposed to address the problem of software reliability measurement into four types according to the nature of the failure process: time between failures models, failure count models, fault seeding models and input domain based models. Goel also pointed out that an assessed value of the software reliability measures is always relative to a given use environment [12]. At present, for the mainstream method of software reliability research is to use a software failure data obtained by testing as the reliability analysis and prediction of the software. By predicting the number of the failure data in the next phase of running software, the reliability of the software can be largely assessed [13]. The failure data alone cannot evaluate and characterize the software reliability, unless the workload is constant [7,8]. Consequently, both web workload and failure information are needed to collect for reliability analysis. The web software reliability can also be analyzed by existing models such as the GO model or Nelson model. However, these models are so simple that they cannot depict the relationship between web workload and calendar time [5].

A number of studies adopt Markov models to measure the reliability of modular software. Cheung [2] proposed a user oriented reliability model to measure the reliability of service Markov model was formulated based on the knowledge of individual module reliability and inter-module transition probabilities. In Littlewood's reliability model a modular program is treated as transfers of control between modules following a semi-Markov process. There are also many architecture-based approaches for measuring software reliability. Krishnamurthy and Mathur [6] conducted an experiment to evaluate a method, Component Based Reliability Estimation (CBRE), to estimate software reliability using software components. CBRE involves computing path reliability estimates based on the sequence

of components executed for each test input, and the system reliability is the average over all test runs. Yacoub et al. proposed a scenario-based model for distributed component-based software [1].

Fuzzy Logic

The calculations of web softwares are complex and uncertain. The fuzzy logic provides not only with a meaningful and powerful representation of measurement of uncertainties, but also with a meaningful representation of vague concepts expressed in natural languages. A fuzzy set can be defined mathematically by assigning to easy possible individual in the universe of discourse a value representing its grade of membership in the fuzzy set. This grade corresponds to the degree to which that individual is similar or complicate with the concept represented by fuzzy sets [4]. Let the reliability of web software is measured on the parameters x_i , i=1 to n. These parameters denote various type of errors occurred in web applications. Let $p(x_i)$ be the probability for error of xi. The membership function $A(x_i)$ can be defined as: if L is the tolerance limit than

$$A(x_i) = \begin{cases} 1 & \text{if } \sum p(x_i)=0 \\ 1-\sum p(x_i) & \text{if } \sum p(x_i)\le L \\ 0 & \text{if } \sum p(x_i)>L \end{cases}$$

Evidence theory

The evidence theory is based on two measures [4]: Belief measures and plausibility measures. For a finite universal set X, a belief measure is a function

Bel:P(X)→ [0,1]
Such that Bel(φ)=0,Bel(X)=1 and
Bel(A1 ∪A2∪….∪An) ≥ $\sum_i$ Bel(Ai) – $\sum_{i<j}$ Bel(Ai∩Aj)+….+$(-1)^{n+1}$ *
Bel(A1 ∩A2∩……..∩An) for all possible subsets of X.
A plausibility measure is a function

PI: P(X) → [0,1]
such that PI(φ)=0 PI(X)=1 and
P(A1 ∩A2∩……..∩An) ≤$\sum_i$ PI(Ai) -
$\sum_{i<j}$ PI(Ai∪Aj)+……+$(-1)^{n+1}$ *
PI(A1 ∪A2∪……..∪An) for all possible subsets of x.

A belief measure and a plausibility measure are determined [4] for all set Aε P(X) by the formula:

$$Bel(A) = \sum_{B/B\subseteq A} m(B) \quad \ldots.(1)$$

$$PI(A) = \sum_{B/A\cap B=\varphi} m(B) \quad \ldots.(2)$$

For a belief measure the corresponding basic probability assignment M is determined for all Aε P(X) by this formula:

$$m(A) = \sum_{B|B\subseteq A} (-1)^{|A-B|} Bel(B) \quad \ldots.(3)$$

The evidence can be combined in various ways. The standard way of combining evidence is using Dempster's rule expressed by the formula:

$$M1,2(A) = \frac{\sum_{B\cap C=A} m_1(B).m_2(C)}{1-K} \quad \ldots(4)$$

Where $K = \sum_{B\cup C=\varphi} m_1(B).m_2(C)$

Software Reliability Measurement

The evidence theory can be used to measure software reliability. An error log list provides us various errors occurred in the web software. This list is analyzed and various errors are calculated and tabled. The error log list provides only the type of error occurred. Some times series of errors are shown as only one error i.e., this shows the error which occurs first. So more than one expert may be involved to categorize errors in the log error list. The belief measure, plausibility measure and assignment m are calculated using the formula ①,② and③. Since many experts are analyzing the error log list and giving assignment individually, these should be combined to obtain a joint basic assignment. The ④ is used to get a joint basic assignment. Also the analysis are performed on data that are collected at various time intervals. The data collected at one time interval need not be same for other intervals. In these types of situations, only one expert is engaged to analysis the data. The expert calculates belief measure, plausibility measure and assignment m for each interval and these assignments are combined using the ④ to obtain overall measure of reliability. Here an attempt is made to analyze the reliability of a web software. The errors in the web software are classified into three categories: software errors, hardware errors and database errors. The objective is to find out which error occurs most. Let S, H and D denote the set of software errors, hardware errors and database errors respectively. Two experts performed examination of the errors and provide assignments for M1 and M2 specified in Table 1. Let M1 and M2 are the degree of evidence which each expert obtained by the examination and which supports that various claim that the error belong to one of the sets (S, H, D, SUH, HUD, DUS, SUHUD) of our concern. M1(S∪H) is the degree of evidence obtained by the first expert that the error belongs to the software or hardware.

Table 1. Belief measure, plausibility measure and joint assignment of focal elements

Focal elements	M1	Bel1	M2	Bel2	M1,2	Bel 1,2	Pl 1,2
S	0.34	0.34	0.50	0.50	0.5	0.5	0.5
H	0.24	0.24	0.22	0.22	0.15	0.15	0.15
D	0.42	0.42	0.28	0.28	0.35	0.35	0.35
SUH	0	0.58	0	0.72	0	0.65	0.65
SUD	0	0.76	0	0.78	0	0.85	0.85
HUD	0	0.66	0	0.50	0	0.50	0.50
SUHUD	0	1.00	0	1.00	0	1.00	1.00

The belief measures are calculated as follows
Bel1 (S)=M1(S)=0.34
Bel1(H)=M1(H)=0.24
Bel1(D)=M1(D)=0.042
Bel1(S∪H)=M1(S)+M1(H)+M1(S∪H) =0.58
Bel1(S∪D)=M1(S)+M1(D)+M1(S∪D) =0.76
Bell(H∪D)=M1(H)+M1(D)+M1(H∪D)=0.66
Bel1(S∪H∪D)=M1(S)+m1(H)+M1(D)+ M1(S∪H) + M1(S∪D)+ M1(H∪D)+M1(S∪H∪D) =1.00

Similarly the belief measures are calculated for the second experts measurements and given in Bel2 column of table 1. The two experts evidence measure are combined and M1,2 is calculated as follows

K=M1(s).M2(h)+M1(S).M2(d)+M1(h).M2(s)+M1(H).M2(D)+
M1(D).M2(S)+M1(D).M2(H)

0.34*0.22+0.34*0.28+0.24*0.50+0.24*0.28+0.42*0.50+0.42*0.22=0.6596
1-K=0.34

M1,2(S)=[M1(S).M2(S)+M1(S).M2(S∪H)+M1(S)M2(S∪D)+
M1(S).M2(S∪H∪D)+M1(S∪H).M2(S)+M1(S∪H).M2(S∪D)
+M1(S∪D).M2(S)+M1(S∪D).M2(S∪H)+M1(S∪H∪D).M2(S)]/
0.34 =0.5

Similarly, M1,2(H) and M1,2(D) are calculated.

M1,2(S∪H)=[M1(S∪H).M2(S∪H)+M1(S∪H)-
M2(S∪H∪D)+M1(S∪H∪D).M2((S∪H)]/0.34= 0
M1,2((S∪H∪D) = [M1(S∪H∪D).M2(S∪H∪D)] /0.34 =0

Similarly, M1, 2(S∪D) and M1, 2(H∪D) are calculated.

The joint basic assignments are used to calculate bel1, 2

Bel 1,2(S)= M1,2(S) =0.5

Bel 1,2(H)= M1,2(H) =0.15

Bel 1,2(D)= M1,2(D) =0.35

Bel 1,2(S∪H)= M1,2(S) + M1,2(H) +M1,2(S∪H)=0.65

Bel 1,2(S∪D)= M1,2(S) + M1,2(D) +M1,2(S∪D)=0.85

Bel 1,2(H∪D)= M1,2(H) + M1,2(D) +M1,2(H∪D)=0.50

Bel 1,2(S∪H∪D)= M1,2(S) + M1,2(H) + M1,2(D) + M1,2(S∪H)+ M1,2(S∪D)+ M1,2(H∪D)+ M1,2(S∪H∪D)=1.0

The plausibility measure is calculated using the formula Pl(A)=1-Bel(Ã) and given in Pl column of Table 1.

Belief measure measures the strength of evidence in favour of set of propositions. Its ranges from 0 to 1. It is the lower bound for the hypothesis to be true. The plausibility measure is an upper bound for the hypothesis to be true. The hypothesis "Error is due to software" has a belief of 0.5 and plausibility measure of 0.5. This means that there is an evidence for software error with a confidence 0.5. However the evidence contrary to the hypothesis has a confidence of 0.5. In this example belief measure and plausibility measure for all the focal elements are same because all the focal elements are mutually exclusive. If the focal elements are not mutually exclusive, then the plausibility measure will vary. Belief measure states that the strangeness of evidence, one minus plausibility measure gives the contrary to the evidence and in between belief measure and one minus plausibility give the hypothesis may or may not be true.

Conclusion

Reliability measurement is difficult because software is estimated to know how long it will work without giving any problem. Already existing methods of probability and hypothesis testing for measuring reliability give only possibility of working or not working. These results of true or false are referred only negation of the other one. But evidence theory provides three values: Possibility for hypothesis is true, hypothesis is false and hypothesis is true or false. Since the belief measure and plausibility measure act as a lower bound and upper bound for the hypothesis to be true, this measure is better than previous measure for reliability calculations.

REFERENCES

1. Andrew G. Liu, Ewa Musial, Mei-Hwa Chen, Progressive Reliability Forecasting of Service-Oriented Software, IEEE Intrernational Conference on Web services(ICWS),2011.
2. R. C. Cheung. A user-oriented software reliability model. IEEE Transactions on Software Engineering, 6(2):118, pp118-125, March 1980.
3. L. Goel, "Software Reliability Models Assumptions, Limitations, and Applicability", IEEE Trans. Soft. Eng., SE-11, No. 12, pp 1411 -1423, Dec. 1985.
4. George J. Klir and bo yuan, Fuzzy sets and Fuzzy logic theory and applications,PHI Learning private limited, 2009.
5. Jianfeng Yang, Zhouhui Deng, Web Software Reliability Analysis with Yamada Exponential Testing-Effort, 9th International Conference on Reliability,Maintainability and safety(ICRMS), pp 760 – 765, 12-15 June 2011.
6. S. Krishnamurthy and A. P. Mathur. On the estimation of reliability of a software system using reliabilities of its components. Proceedings of Eighth International Symposium on Software Reliability Engineering (ISSRE), pp 146-155, Albuquerque, NM. USA, November. 1997
7. S. Y. Kuo, C. Y. Hung and M. R. Lyu, "Framework for modeling software reliability, using various testing efforts and fault detection rates", IEEE Transactions on Reliability, Vol. 50, no.3, pp 310-320, 2001.
8. Dr. Linda Rosenberg, Ted Hammer and Jack Shaw, SOFTWARE METRICS AND RELIABILITY, ISSRE '98 Best Paper, 1998.
9. Li Qiuying and Li Haifeng, A Software Reliability Evaluation Method, International Conference on Computer and Software Modeling 2011, IPCSIT vol.14, Singapore, 2011.
10. J. D. Musa, A. Iannino and K. Okumoto, Software Reliability: Measurement, Prediction and Application, McGraw-Hill, 1987.
11. O.Naga Raju, Software Reliability Growth Models for the Safety Critical Software with Imperfect Debugging, International journal on Computer Science and Engineering(IJCSE), Vol. 3, No. 8, August 2011.
12. Shi Zhao, Xiaoming Lu, Xianzhong Zhou, Tie Zhang, Jianqiang Xue, A Reliability Model For Web Services, International Conference on Computer Science and service system(CSSS), pp 91 - 94 , 27-29 June 2011.
13. Zhu Xiao-meia, Zhang Qun-yanb, Ren xinc , The Research of Software Reliability Measure Based on Conditional Value at Risk, International Conference on Electronoic and Mechanical Engineering and information Technology (EMEIT), pp 3092 – 3094, 12-14 Aug. 2011.

****