A COMPARATIVE STUDY OF FAULT DENSITY-BASED SOFTWARE RELIABILITY ESTIMATION TECHNIQUES UTILIZING FUZZY NETWORKS

Chintalapally Sandeep Kumar
Research Scholar
Arni University
Himachal Pradesh.

Dr. Regonda Nagaraju
Professor
Malla Reddy University
Hyderabad.

Dr. Prasadu Peddi
Professor
Arni University
Himachal Pradesh.

ABSTRACT

Software dependability pertains to the probability of software continually delivering correct service within a certain duration and under particular circumstances. Identifying commonly recurring problems throughout the development process is becoming more critical in many software industries. Identifying software defects in software project modules is a complex and intrinsically uncertain process. Although there are several intricate machine learning and deep learning models available for predicting problems, it is essential to create a simple model that integrates the knowledge of domain experts and successfully handles uncertainty in feature measurements. We have developed a software fault prediction model using the Mamdani Fuzzy Logic approach. This model has the capability to use conventional membership functions like triangle and trapezoidal, together with custom membership functions created by domain specialists, in order to produce accurate predictions about software faults.

***Key words**: software fault prediction model using the Mamdani Fuzzy Logic approach*

INTRODUCTION

Currently, the majority of individuals rely on software either directly or indirectly. The reliance of governments and humans on software has significantly grown during the last three decades. Software dependability and quality modeling are crucial due to the widespread usage of software in many application domains. Several historical occurrences demonstrate the impact of software failures and vulnerabilities that were present in. Therefore, software engineers have a significant task of constructing dependable software with few errors. The prediction of software dependability is very significant. Dependability models are only useful for obtaining an accurate evaluation of dependability during the latter stages of software development. The current reliability models are only useful during the final phases of development, providing assistance to developers either towards the conclusion of the coding process or during the testing phase. It is now too late for the developers to implement remedial measures to enhance the dependability of the product. Nevertheless, the prediction of software dependability throughout the early stages of the Software Development Life Cycle (SDLC) is a crucial problem due to the goals of cost-effectiveness and efficient resource management. Computer programs or applications Reliability refers to the likelihood that a system can consistently provide its intended functionality and quality within a certain timeframe and specific circumstances, beginning from the start of this timeframe. The primary approach for assessing software quality is via the identification of software defects, often measured using the software defect density metric. The software defect density is calculated by dividing the total number of faults by the size of the program. Predicting software fault density plays a crucial role in ensuring the reliability of software. To accomplish the goal of defect estimation, it is necessary to anticipate the defect density indicator in each step of the Software Development Life Cycle (SDLC) starting from the early stages, particularly at the conclusion of each phase. Several models have been suggested for the estimate and prediction of software dependability during the last thirty years. Traditional methods used to estimate the defect density at the conclusion of the testing phase in software development life cycle (SDLC) are not consistently accurate in projecting software dependability and are not easily understandable to users. The lack of information at the first stages of the Software Development Life Cycle (SDLC) may be attributed to a deficiency of expert knowledge, which can be measured using software metrics.

LITERATURE REVIEW

Ini John Umoeka [2023] the reliability of software product is seen as critical quality factor that cannot be overemphasized. Since real world application is loaded with high amount of uncertainty, such as applicable to software reliability, there should be a technique of dealing with such uncertainty. This paper presents a reliability model to effectively handle uncertainty in software data to enhance reliability prediction of software at the early (requirements and design) stages of Software Development Life Cycle (SDLC).

T. Ravi Kumar [2017] Software Quality analysis is one of the significant criteria required to explore the software life and additionally software reliability. Software quality is been characterized under various parameters. Software risk analysis is one such basis required to distinguish the software reliability. At the point when software is arranged or being created by the sort of software and in addition the endeavors required to build up the software by and large characterizes the software hazard.

Vivek Kumar Singh [2016] today, the influence of information technology has been spreading exponentially, from high level research going on in top labs of the world to the home appliances. Such a huge demand is compelling developers to develop more software to meet the user expectations. As a result reliability has come up as a critical quality factor that cannot be compromised. Therefore, researchers are continuously making efforts to meet this challenge. With this spirit, authors of the paper have proposed a highly structured framework that guides the process of quantifying software reliability, before the coding of the software start.

Gopal Prasad Jaiswal [2015] Software reliability means the probability of the uninterrupted operation of a software system for a specified period of time in a specified environment. Software Reliability Modeling has been one of the much-attracted research domains in Software Reliability Engineering. Day by day software applications are growing more complex and with more emphasis on reuse. Component Based Software (CBSS) applications have developed. We concentrate in this paper is to provide an model for the Component Based Systems reliability estimation.

Reliability & Failure

Reliability refers to the likelihood that a system will successfully carry out its intended operation within a certain timeframe and under specific operating conditions. Reliability, from a qualitative perspective, refers to the item's capacity to maintain its functionality. Reliability, in quantitative terms, refers to the likelihood that there will be no operational disruptions during a certain time period. However, it is important to note that while redundant pieces might fail, they can still be fixed without causing any disruption to the overall functioning of the item or system.

Fuzzy set operations

A set with items that fluctuate in their degree of membership in the set is called a fuzzy set. The construction of a conceptual framework that is similar to but more general than the framework used in the case of ordinary sets particularly in the areas of pattern classification and information processing can benefit greatly from the use of fuzzy sets, which serve as a suitable starting point.

Hardware and Software Reliability

Reliability is a characteristic of a system that pertains to its consistency and dependability in meeting the needs of the end-user. Typically, a product may be considered dependable if it fulfills its commitments. Once engineering items have undergone the necessary stages of completion, testing, and sale, it is anticipated that the products will function consistently and dependably.. Hardware and software are interdependent.

Importance of Software Reliability Modelling

Software reliability models are very effective for predicting, managing, and evaluating the dependability of software. As the program became larger and more complicated, management problems started to take precedence. Several problems that prompted the development of software engineering.

Types of Software Reliability Models

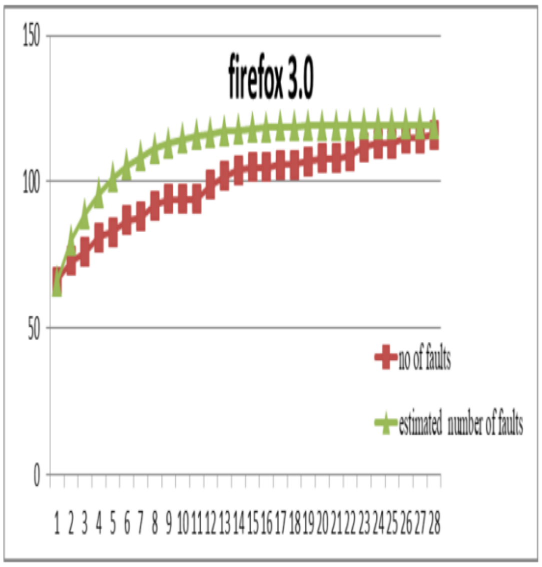
A software reliability model is a mathematical model that predicts the rise in software dependability over time when errors are detected and corrected. The technique is predicated on the concept that a program inherently has flaws, which manifest as observable failures during the testing phase.

METHODOLOGY

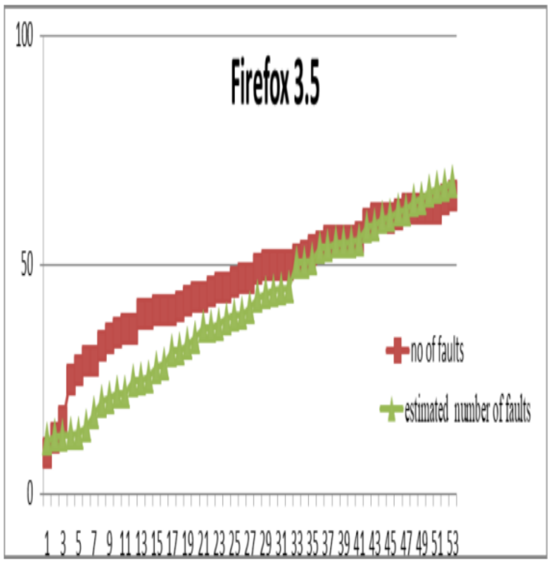
There should be mechanisms in place to prevent software failures in order to minimize significant losses. Prior to any software deployment, it is essential for software developers to implement a degree of quality assurance. This entails having a clear understanding of the likelihood of failure in the real-world context. There is a significant need for accurately assessing the dependability of rapidly deployed open-source software (OSS) systems. Researchers have created several models for estimating software dependability. However, the emergence of new software development environments, advancements in technology, and diverse techniques of software development have necessitated the creation of a new model for estimating software reliability.

RESULTS

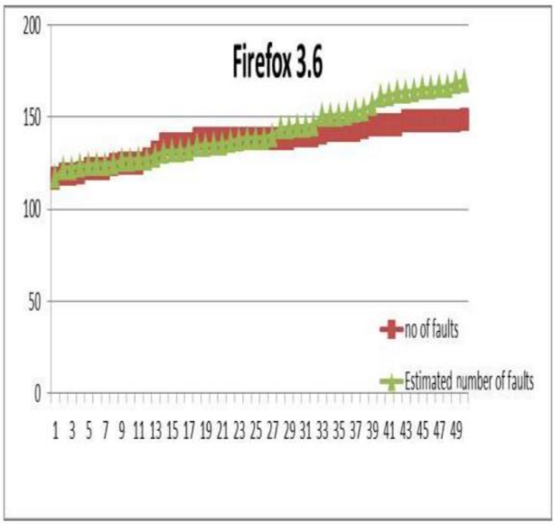
This section examines the suggested model using three sets of failure data from different versions of the open-source program Firefox. The testing workload in each release varies based on the changes in the quantity of faults and the functional complexity of the published version. The goodness of fit measurements indicate that the majority of values are much higher, with the exception of MEOP values where the Inflection s-shaped model showed excellent performance. Additionally, both the Inflection s-shaped model and PTZ model did well in terms of AE values. Overall, the suggested methodology has a 75% success rate in surpassing significant benchmarks. In Firefox 3.6, the suggested model is generally superior to the Inflection s-shaped model, except in terms of the AIC criteria, where the Inflection s-shaped model outperformed the proposed model. Graphs 1, 2, and 3 show the estimated quantity of defects using the suggested model. Overall, the suggested model is outperforming previously utilized models by 87.5%.



Graph. 1 Estimation of the number of faults using Firefox 3.0



Graph. 2 Estimation of the number of faults using Firefox 3.5



Graph .3 Estimation of the number of defects using Firefox 3.6

CONCLUSION

The primary concern in the software development process is to create dependable software at the conclusion of the testing phase. The authors of the study conducted a thorough assessment on the creation of software dependability models. They carefully evaluated how different models are derived from existing models, taking into consideration the assumptions made by these models. Each model has used distinct qualities for the purpose of model building and is based on certain assumptions. Models are classified into twelve categories based on different qualities, assuming certain conditions. These classes are subsequently examined to determine the membership of different models in these categories and to understand their evolutionary development. An in-depth analysis of existing research conducted by scholars is undertaken to thoroughly examine the key challenges in the creation of software dependability models.

Reference

1. *M. Anjum, MA Haque, N. Ahmad, "Analysis and Ranking of Software Reliability Models based on Weighted Criteria Value", Int. Journal of Information Technology and Computer Science, vol.5, no.2, page: 1-14, 2013.*
2. *R. Peng, Y.F. Li, W.J. Zhang, Q.P. Hu, "Testing effort dependent software reliability model for imperfect debugging process considering both detection and correction", Reliability Engineering & System Safety, Vol. 126, pp. 37-43, 2014.*
3. *K. M. S. Faqih, "What is Hampering the Performance of Software Reliability Models? A literature review", In Proceedings of the International MultiConference of Engineers and Computer Scientists, Hong Kong, 2009.*
4. *C. Wohlin, M. Host, P. Runeson and A. Wesslen, "Software Reliability", In Encyclopedia of Physical Sciences and Technology (3rd edition), Vol. 15, 2001, Academic Press.*
5. *H. Pham, M. Pham, "Software Reliability Models for Critical Applications", A report on Software Reliability Research Program, Idaho National Engineering Laboratory, ECG-2663, Dec. 2011.*
6. *U. Sumita and J. G. Shanthikumar, "A Software Reliability Model with Multiple-Error Introduction & Removal," in IEEE Transactions on Reliability, vol. 35, no. 4, pp. 459-462, Oct. 1986, doi: 10.1109/TR.1986.4335507.*
7. *M. Xie, B. Yang, "A study of the effect of imperfect debugging on software development cost", in IEEE Transactions on Software Engineering, vol. 29, no. 5, pp. 471-473, May 2003.*
8. *U. Sumita and J. G. Shanthikumar, "A Software Reliability Model with Multiple-Error Introduction & Removal," in IEEE Transactions on Reliability, vol. 35, no. 4, pp. 459-462, Oct. 1986, doi: 10.1109/TR.1986.4335507.*
9. *N. D. Singpurwalla, and S. P. Wilson. "Software Reliability Modeling." International Statistical Review, Vol. 62, no. 3, pp. 289–317, 1994.https://doi.org/10.2307/1403763.*
10. *V. Almering, M. van Genuchten, G. Cloudt, P. J. M. Sonnemans, "Using Software Reliability Growth Models in Practice," in IEEE Software, vol. 24, no. 6, pp. 82-88, Nov.-Dec. 2007.*