Search Based Techniques for Software Fault Prediction: Current Trends and Future Directions

Ruchika Malhotra
Department of Software Engineering, Delhi Technological University, Delhi-110042, India
+91 99 10 290445
ruchikamalhotra2004@yahoo.com

ABSTRACT
The effective allocation of the resources is crucial and essential in the testing phase of the software development life cycle so that the weak areas in the software can be verified and validated efficiently. The prediction of fault prone classes in the early phases of software development can help software developers to focus the limited available resources on those portions of software, which are more prone to fault. Recently, the search based techniques have been successfully applied in the software engineering domain. In this study, we analyze the position of search based techniques for use in software fault prediction by collecting relevant studies from the literature which were conducted during the period January 1991 to October 2013. We further summarize current trends by assessing the performance capability of the search based techniques in the existing research and suggest future directions.

Categories and Subject Descriptors
D.2 [**Software Engineering**]: D.2.8 Metrics, D.4.8 Performance

General Terms
Measurement, Performance, Reliability, Verification.

Keywords
Software Quality, Search Based Techniques, Software Fault proneness.

1. INTRODUCTION
Software testing is the combination of verification and validation methods in order to assess the quality of a software product. It involves discovery of hidden defects, inconsistencies and the identification of classes, which deviate from the intended functionality by providing undesired outputs. Testing activities consume a large percentage of resources allocated to a project. However, there is always a shortage of resources for effective testing. Thus, we need to optimize the testing resources by efficient and effective planning. One possible method for planning resource usage is to develop quality models, which predict fault proneness, attribute of a particular class. Researchers have successfully found relationship between various Object-Oriented metrics of a class and its fault proneness attribute. These models can effectively predict faulty classes in the early phases of software development life cycle. We can thus employ the testing resources on the weak portions to produce low cost and good quality software.

Over the years, various statistical and Machine Learning (ML) techniques have been used to develop models for predicting faulty classes. Harman et al. recommend the use of the Search Based Techniques (SBT) for developing prediction models in software engineering domain [6-7]. However, only a few studies have been conducted which explore the predictive capability of the SBT in developing the Software Fault Prediction (SFP) models. This paper is an attempt to analyze the current position of SBT for developing the SFP models. The procedure followed by us is similar to that of a review study as discussed by Kitchenham et al. [8]. This paper focuses on various research questions, which identify the current trends by collecting all the relevant studies and extracting and synthesizing data from them. We further report answers to the following research questions: (1) Which SBT have been used for developing the SFP Models? (2) Which are the most widely used datasets, performance measures, and metrics using the SBT for developing the SFP Models? (3) What is the overall performance of the SBT for developing the SFP Models? We also identify future directions to provide guidelines to the practitioners for conducting studies using the SBT for the SFP.

2. RESEARCH RESULTS
The data collection procedure followed by us yielded eight primary studies [1-4, 9-12]. The answers pertaining to specific research questions are discussed below. According to the eight selected primary studies, six SBT have been used till date for developing SFP models. These techniques include Ant Colony Optimization (ACO) [2, 12], Multi-Objective Particle Swarm Optimization (MOPSO) [3-4], Genetic Algorithms (GA) [9], Genetic Programming (GP) [1, 9], Artificial Immune Recognition Systems (AIRS) [1] and Simulated Annealing- Probabilistic Neural Network (SA-PNN) [11]. Out of the eight selected studies, five studies [3-4, 10-12] use NASA data sets which are publically available at the PROMISE repository, two studies uses other open source projects [2, 9] while only one study uses an industrial data set collected from a telecommunication industry [1]. Thus, the most widely used datasets for SFP are the NASA data sets. The most widely used metrics for the SFP were found to be the Object-Oriented metrics proposed by Chidamber and Kemerer (CK). These metrics were used by five studies out of eight. Other metrics include Halstead metrics, complexity metrics, and fault slip through metric. In order to assess the performance of the predicted models for the SFP, the selected studies used a number of performance measures like accuracy, Area under the Curve (AUC) using ROC (Receiver Operating Characteristic), recall, specificity, precision and f-measure. However, the accuracy and AUC were the most frequently used performance measures. The overall performance of the SBT techniques was assessed by calculating the average accuracy measure (80.75%) and average

SBST'14, June 2 – June 3, 2014, Hyderabad, India
ACM 978-1-4503-2852-4/14/06
http://dx.doi.org/10.1145/2593833.2593842

AUC measure (0.816) of the selected studies. Figure 1 shows the accuracy (five studies) and AUC values in percentage (four studies) of the SFP models predicted using the SBT. The results show that the SBT yield high accuracy and AUC values.

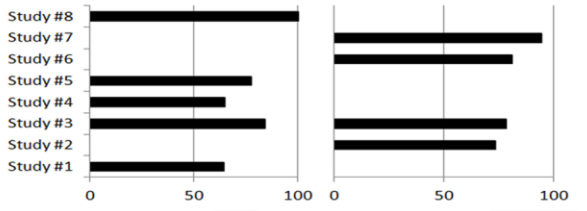

Figure 1: Accuracy & AUC for SFP models using SBT

3. DISCUSSION OF CURRENT TRENDS

Although, a few studies have used SBT for SFP, the initial results obtained from these studies are encouraging. The current trends discussed in the study spark the following questions: **(1) Are the SBT effective for practical implementation taking into account their slow nature and memory consumption?** It is a possibility that software practitioners are cautious in using SBT as these techniques have slow learning potential and require large memory consumption. Thus, it is important for researchers to conduct a number of studies to provide empirical evidence which support the use of SBT for SFP and evaluate the competency of SBT with existing techniques. **(2) Why is there a lack of industrial data sets for empirical validation?** In order to prove the applicability of SBT, researchers need real world industrial data sets which are generally private in nature. Research results in Section 2 indicate only one such study. There is a need for formulating data repositories which share industrial data sets so that researchers can explore and evaluate SBT on these data sets. **(3) Which SBT and fitness functions should be used in a particular scenario?** A vast number of choices are available in terms of SBT and fitness functions for application in a particular scenario. A researcher should carefully select a technique and fitness function by exploring a wide variety of available techniques for SFP [6]. **(4) Is there a scope of applicability of hybrid SBT for SFP?** There are ML techniques that could be used in combination with SBT to allow the researchers to use the advantages of a specific ML technique with a SBT into one single approach. Further, hybridization of SBT with ML technique could lead to better results by improving the quality of candidate solutions, by proper parameter selection of a SBT or by incorporating SBT as a part of a larger system [5]. Certain studies [9-10] have already tried to explore hybrid techniques for SFP.

4. CONCLUSION & FUTURE DIRECTIONS

We found very few (eight) studies that examine the effectiveness of the SBT such as GP, ACO etc. for SFP. The most commonly used datasets for SFP were NASA data sets from the Promise repository while the most commonly used metrics were CK metrics. The AUC measure and the accuracy measure were most commonly used performance measures for SFP. The initial results show that the SBT should be used for developing SFP models as they show high accuracy (80.75%) and high AUC (0.816) values. The future directions are summarized as follows: (1) More studies should be conducted which explore the predictive capability of the SBT and hybrid SBT for developing the SFP models (2) In order to assess the practical usefulness of the SBT, researchers should conduct more studies using industrial data sets. (3) A number of other performance measures like h-measure, g-measure etc. should be evaluated for the SFP models (4) Researchers should perform studies which assess and rank the predictive capabilities of various SBT for developing the SFP models using statistical tests like Friedman (5) Studies should be conducted which evaluate and compare the performance of different SBT with statistical and ML techniques (6) Software practitioners should examine and develop research tools which ease the use of SBT for SFP.

5. REFERENCES

[1] Afzal, W. 2010. Using Faults-Slip-Through Metric as a Predictor of Fault-Proneness, In *Proceedings of Asia Pacific Software. Engineering Conference,* USA, 414-422.

[2] Azar, D. and Vybihal, J. 2011. An ant colony optimization algorithm to improve software quality prediction models: Case of class stability, *Inf. Softw. Technol.,* 53, 4 (April 2011), 388–393.

[3] Carvalho, A. B., Pozo, A.,Vergilio, S. and Lenz, A. 2008. Predicting Fault Proneness of Classes Trough a Multiobjective Particle Swarm Optimization Algorithm. In *Proceedings of 20th IEEE International Conference on Tools with Artificial Intelligence,* USA, 387–394.

[4] Carvalho, A.B., Pozo, A. and Vegilio, S.R. 2010. A symbolic fault-prediction model based on multiobjective particle swarm optimization, *J. Syst. Softw.,* 83, 5 (May 2010), 868-882.

[5] Grosan, C. and Abraham A., 2007. Hybrid Evolutionary Algorithms: Methodologies, Architectures and Reviews, *Studies in Computational Intelligence*, 75, 1-17.

[6] Harman, M., Jones, B. F., 2001. Search based Software Engineering, *Information & Software Technology*, 43, 14 (December 2001), 833-839.

[7] Harman, M., 2010. The relationship between Search Based Software Engineering and Predictive Modelling, In *Proceedings of 6th International Conference on Predictive Models in Software Engineering,* USA, 1-13.

[8] Kitchenham, B. A. and Charters, S. 2007. *Guidelines for performing systematic literature review in Software Engineering.* Technical Report.

[9] Malhotra, R. and Jain, A. 2012. Fault Prediction Using Statistical and Machine Learning Methods for Improving Software Quality, *J. Inf. Process. Syst.,* 8, 2 (June 2012), 241–262.

[10] Martino, S., Ferrucci, F., Gravino, C. and Sarro, F., 2011. A Genetic algorithm to configure Support vetcor machine for predicting Fault prone components, In *Proceedings of 12th International Conference on Product Focus Software Process Improvement,* PROFES'11, 6759, 247-261.

[11] Pendharkar, P. C. 2010. Exhaustive and heuristic search approaches for learning a software defect prediction model, *Eng. Appl. Artif. Intell.,* 23, 1 (February 2010), 34–40.

[12] Vandecruys, O., Martens, D., Baesens, B., Mues, C., Backer, M. D. and Haesen, R. 2008. Mining software repositories for comprehensible software fault prediction models, *J. Syst. Softw.,* 81, 5 (May 2008), 823–839.