VOL. 11, NO. 1, JANUARY 2016 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

www.arpnjournals.com

A COMPARATIVE STUDY ON SELECTING AND RANKING THE TEST CASES IN SOFTWARE TESTING

Rajesh Kaluri, Kuruva Lakshmanna, Thippa Reddy, Sudheer Karnam and Srinivas Koppu
School of Information Technology and Engineering, VIT University, Vellore, Tamil Nadu, India
E-Mail: rajesh.kaluri@vit.ac.in

ABSTRACT

Software testing is usually measured by executing the 'n' number of test cases and those test cases has to be designed using the familiar test case design techniques. The aim of Test case design technique is to ensure the detection of typical bugs, systematic coverage and redundant testing. Executing and checking the test cases is not possible all the time in the manual testing. Therefore selecting a test case and ranking is important. The objective of the test case selection is to have a better test case from a pool of test cases and assigning the rank to each test case will leads the software asan error free and which gives a good efficiency. Ranking of test cases is especially useful if a system is a having large number of test cases. Hence selecting and ranking a test case plays an important role in the software testing. In this paper, importance of selection and ranking test cases are analysed and its methods are discussed.

Keywords: average percentage rate of faults detection (APFD), fault-exposing-potential (FEP), genetic algorithm (GA), regression test technique (RTT), regression test selection (RTS), regression test prioritization (RTP), test case selection (TCS).

1. INTRODUCTION

In the Software Industry, maintaining the software products is an expensive phase and which costs around 70 to 80 percent of the total product price. Test case is one which will have a test input, test condition (pre, post) and an output with some result (status - pass or fail). Regression testing and Re-testing are the important techniques used in the software maintenance activity [2]. Re-testing enforces the test cases to work in the same process but Regression testing is been used in the software maintenance not to affect any changes in the system. Regression testing makes the tester to be confident and which will not harm to the existing systems [4]. The approaches for the regression testing are minimizations of test cases, selection and ranking the test cases effectively. Regression Testing Technique (RTT) will provides the service to the tester to select the test cases from the pool of test cases (Test Suite) and henceforth its effectiveness in the system maintenance. Therefore, the various test case selection and ranking techniques are analysed and has shown the techniques that are good over cost and performance of the test cases in the software systems.

The Figure-1 gives the various stages of the Software Testing Life Cycle and each of its stage will have a prominent role in completing the testing process of a software system. It basically starts with the Planning and controlling the necessary requirements to the given software application [10]. Implementation and Execution of the software product system will be purely based on the analysis and design made by the experts. Finally the test closure activity can be done after the successful evaluation of the software product with the minimal conditions.

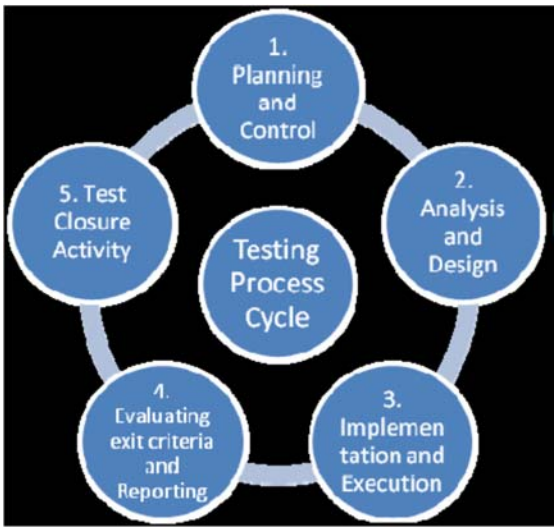

Figure-1. Stages in software testing life cycle.

In order to support for software testing methods, software testing tools have been developed and still they are using to provide the full automation over the methods [1, 2]. The strength of Testing mainly depends on three things; one is test case selection or generation, test case execution and test case evaluation. Among the three things, test case selection is the difficult and has to give topmost consideration. A test case is assumed to be having good code coverage if it uncovers/detect maximum number of faults with minimum number of test cases and having high fault detection capability. A software product will be tested by using all the test cases of the test suite. Generally the test effort will starts at the time of requirement collection and it mainly focus on the coding

process. Most of the testers will not systematize their test planning, test design and test estimation among other processes [2]. The below Figure-1 will gives the test automation average for each of the software test processes.

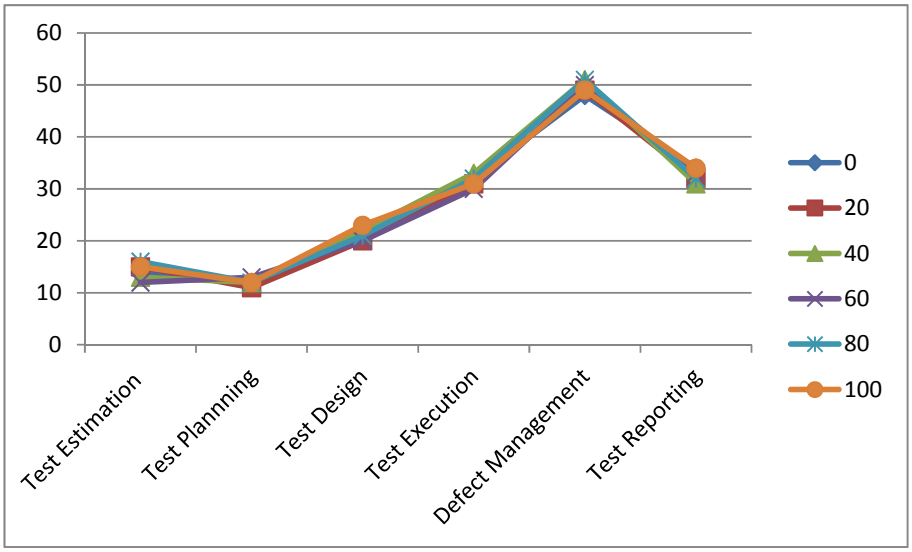

Figure-2. Test automation average for software test process.

2. LITERATURE REVIEW

Test cases can be reduced by using effective algorithms

Software Testing life cycle consists of the test plan, test creation, test design, test case execution, test deliverables, test review etc. Regression testing is a process of repeating the testing with the same test cases or new test cases without affecting the result [3]. In the test execution, regression testing plays a vital role since its algorithm has been used to select the test cases from the pool of test cases. Test case ranking algorithm is very much powerful in finding the best test cases from the test suite and henceforth the comparison of test cases is also quite easy. Especially the test case ranking algorithm can be used for the system or product having bulky test cases.

Genetic algorithm (GA) is a quickly rising area in Artificial Intelligence and it's inspired by Darwin's theory. Principally this process has started with the set of solutions which is basically represented by chromosome known as population. Test cases can be generated by applying Genetic Algorithm and which is a big breakthrough in the software testing process. Random numbers will be generated by the source code and then the GA produces the chromosomes for the initial generation which is the binary representation of every random number taking this a current generation the fitness of every chromosome is evaluated using the fitness function [8].

Search based constrained test case selection: This paper mainly focuses on the test case selection based on the constraints. An algorithm named Binary Constraints Particle Swarm Optimization (BCPSO) is used for test case selection [5]. According to Yoo and Harman, selecting a Test Case has to treat like a search optimization problem. Here, search techniques discover the space of probable solutions and looking for the result that best matches the test objectives. Testing consumes 40 - 50% of the total project development cost. Test Case Selection (TCS) can be done by smoothly reducing the irrelevant test cases from the Test Suite The PSO algorithm states the random population of particle, in which each will have a position in the exploration space.

By using fitness function, the particles will be evaluated and if the particle reaches to the better position, then the algorithm breaks. Hybrid search method was developed and validated. Hybrid implementation of BCPSO was established by linking BCPSO with forward selection and hill-climbing algorithms. Test Case Selection process value will depends upon the effectiveness of the used search technique, constraints imposed by the user and the feature of the test suite at hand. Test Suites investigated and performed on limited only. The test cases are selected only based on the requirements. The quality of the process can be increased only if the proper test case selection is done.

Effective test selection for the end user: The paper entitled on Effective test Selection for the End User enables the tester - how to rank the test cases by following the various prioritization practices. The various practices analysed are Random prioritization, Total fault-exposing - potential (FEP) prioritization, Additional fault-exposing-potential (FEP) prioritization, Total branch coverage

prioritization, Additional branch coverage prioritization, No prioritization. The outcome displays that the fault detection rate of test cases are increased by the prioritization practices. RANDOM selection methods, MOST-RELEVANT coverage and CANONICAL prioritization are measured as baseline approaches and planned CONFIDENCE Prioritization, COS-DIST prioritization and LEAST-RELEVANT prioritization especially to find surprise failure [6]. Hence, the three proposed approaches use changed methods to order the test case and prioritize it. Test case selection efficiency can be increased by the above three methods.

A new code based test case prioritization technique: This paper mainly focuses on the Test Case Prioritization (ranking) technique. Ranking the test cases is an important feature in the software testing environment which leads to the success of the pool of test cases. In general, the test cases will be having more value in the test suite if they cover the maximum part of the code and regression testing technique can be applied to the different level of testing. Even in the regression testing, ranking the test case will give a better performance than with an ordinary approach.

Three techniques are very much used for decreasing the time and costs of regression testing, which are regression test selection, test suite minimization and test case prioritization techniques [7]. Hence these three techniques will tries to reduce the regression testing cost by choosing suitable test cases. New code based test case prioritization technique will suits to the test suite with minimal test cases.

3. CONCLUSIONS

Many researchers has clearly quantified on the test case selection and ranking with their methods. The theories of Regression testing tell that it is expensive and which has power algorithms and which gives a novel methodology for test case selection and ranking of test cases in the test pool. If an original program has 'n' number of test cases and some of the test cases have been changed and the entire program can't run properly because of the changed test cases, so the selection and ranking of test cases is important in any of the program. Hence, the ranking and selection of test cases are very much important for the larger systems.

REFERENCES

[1] Abran A., Moore J.W., Bourque P., Dupuis R. 2004. Guide to the software engineering body of knowledge: 2004 version. IEEE Computer. Society.

[2] J. Lee, S. Kang, D. Lee. 2012. Survey on software testing practices. IET Software. 6(3): 275-282.

[3] A.Pravin and Dr. S. Srinivasan. 2013. An Efficient Algorithm for Reducing the Test Cases which is Used for Performing Regression Testing. 2nd International Conference on Computational Techniques and Artificial Intelligence (ICCTAI'2013).

[4] Siavash Mirarab, Soroush Akhlaghi and Ladan Tahvildari. 2012. Size-Constrained Regression Test Case Selection Using Multicriteria Optimization. IEEE Transactions on Software Engineering. 38(4):

[5] Luciano S. deSouza, Ricardo B.C. Prudêncio, Flaviade A. Barros, Eduardo H.daS. Aranha. 203. Search based constrained test case selection using execution effort. ELSEVIER, Expert Systems with Applications. 40: 4887-4896

[6] Muhammad Shahid and Suhaimi Ibrahim. 2014. A New Code Based Test Case Prioritization Technique. International Journal of Software Engineering and Its Applications. 8(6): 31-38.

[7] Samaila Musa, Abu BakarMd Sultan, Abdul Azim Bin AbdGhani, SalmiBaharom. 2014. A Regression Test Case Selection and Prioritization for Object-Oriented Programs using Dependency Graph and Genetic Algorithm. Research Inventy: International Journal of Engineering and Science. 4(7): 54-64 ISSN (e): 2278-4721, ISSN (p): 2319-6483.

[8] Roy P. Pargas, Mary Jean Harrold, Robert R Peck. 1999. Test Data Generation using Genetic Algorithms. Journal of Software Testing, Verification and Reliability.

[9] Harpreetkaur. 203. Comparative Study of Automated Testing Tools: Selenium, Quick Test Professional and Test Complete. Journal of Engineering Research and Applications, ISSN: 2248-9622, 3(5): 1739-1743.

[10] Rishab Jain C and Rajesh Kaluri. 2015. Design of automation scripts execution application for Selenium Webdriver and TestNG Framework. ARPN Journal of Engineering and Applied Sciences. 10(6).

[11] Panigrahi C. and Mall R. 2013. A heuristic-based regression test case prioritization approach for object-oriented programs. Innovations in Systems and Software Engineering (Springer), doi: 10.1007/s11334-013-0221-z, pp. 1-9.

[12] Jin W., Orso A. and Xie T. 2010. Automated Behavioral regression testing. Third International Conference on Software Testing, Verification and Validation, Washington DC, USA. pp. 137-146.

[13] Gupta R. and Yadav A. K. 2013. Study of Test Case Prioritization Technique Using APFD. International Journal of Computers and Technology. 10(3): 1475-1481.

[14] Panigrahi C. and Mall R. 2013. An approach to prioritize regression test cases of object-oriented programs. JCSI Trans on ICT (Springer), doi: 10.1007/s40012-013-0011-7, pp. 1-15.

[15] Frechette N., Badri L., and Badri M. 2013. Regression Test reduction for object-oriented software: A control call graph based technique and associated tool. International Scholarly Research Network Software engineering (ISRN Software Engineering). 2013(ID 420394): 1-10.

[16] Shaveta Malik: 2010. Performance Comparison between Ant Algorithm and Modified Ant Algorithm. International Journal of Computer Science and Applications. 1(4): 42-45.

[17] Ruchika Malhotra, Arvinder Kaur and Yogesh Singh: 2010. A Regression Test Selection and Prioritization Technique. Journal of Information Processing Systems. 6: 235-252.

[18] Y. C. Huang, K. L. Peng and C. Y. Huang. 2012. A history-based cost-cognizant test case prioritization technique in regression testing. Journal of Systems and Software. 85(3): 626-637.

[19] P. K. Chittimalli and M. J. Harrold. 2009. Recomputing coverage information to assist regression testing. Software Engineering. IEEE Transactions. 35(4): 452-469.

[20] S. Yoo and M. Harman. 2012. Regression testing minimization, selection and prioritization: a survey. Softw. Test. Verif. Reliab. 22(2): 67-120.

[21] A. Ngah and K. Gallagher. 2009. Regression test selection by exclusion using decomposition slicing. Proceedings of the European Software Engineering Conference and the ACM SIGSOFT Symposium on the Foundations of Software Engineering (ESEC/FSE), Amsterdam, the Netherlands. pp. 23-24.

[22] Rothermel G. and Harrold M. J. 1997. A safe, efficient regression test selection technique. ACM Transactions on Software Engineering and Methodology. 6(2): 173-210.