UNIVERSITI PUTRA MALAYSIA

TEST CASE PRIORITIZATION APPROACH FOR SEQUENCE OF EVENTS USING COMPLEXITY FACTOR

EMYREEMA BINTI JA'AFAR

FSKTM 2018 32

TEST CASE PRIORITIZATION APPROACH FOR SEQUENCE OF EVENTS USING COMPLEXITY FACTOR

By

EMYREEMA BINTI JA'AFAR

Thesis Submitted to School of Graduate Studies,
University Putra Malaysia, in Fulfillment of the
Requirement for the Degree of Master of Computer Science

January, 2018

DEDICATION

Alhamdulillah, my grateful and praises to the Almighty of Allah who has inspired, strengthened, guided and ease the implementation of this project. I dedicate this dissertation especially to my mum, Amnah Binti Md Salleh for her prayers and supervisor for their countless supports through this journey to finish up the thesis as part of requirement fulfillment for Master of Computer Science. I also would like to express appreciation from my deepest heart to all my family members, lecturers and friends who have assisted directly or indirectly and supported me during this postgraduate study.

ABSTRACT

Test case prioritization (TCP) is a method to prioritize and schedule test cases. The technique is developed in order to run test cases of higher priority for minimizing the time, cost and effort during software testing phase. Complexity is one of the factors that affect the effectiveness of the TCP. However, the existing techniques for measuring complexity have some limitations. This is due to inaccuracy in finding the weightage value for complexity as the value will be used to determine the test case prioritization. This study aims on proposing a TCP approach using complexity factor in order to get a better accuracy in prioritizing the test case for event sequences. The study use a Branch Coverage Expectation (BCE) complexity measurement that been proved empirically in the previous research. In this study, an automate tool is developed to calculate the BCE value using a Visual Basic and Microsoft Access. Average Percentage Fault Detection metric is used to evaluate the proposed approach. The fault matrix was build based on the testing done in Junit Eclipse. Based on the results, it shows that by only using complexity factor solely to determine the test case prioritization has does not improve the effectiveness of TCP approach. It is suggests that the proposed approach need to be combined with other factor(s) in order to improve the effectiveness of TCP.

ABSTRAK

Keutamaan Kes Uji ialah salah satu kaedah untuk mencari dan menjadualkan sesuatu kes uji yang penting mengikut keutamaan. Teknik ini dibangunkan untuk menguji lari kes uji yang mempunyai keutamaan yang tinggi untuk meminimumkan masa, kos dan tenaga semasa fasa pengujian sistem. Kerumitan sistem merupakan salah satu faktor yang menentukan samada Keutamaan Kes Uji adalah efektif. Walaubagaimanapun, pendekatan sedia ada untuk mengukur kerumitan sistem mempunyai beberapa kekangan. Ini adalah disebabkan ketidaktepatan mencari nilai pemberat bagi kerumitan program dan nilai ini akan digunakan untuk mengenalpasti keutamaan kes uji. Projek ini bertujuan untuk mencadangkan pendekatan keutamaan kes uji dengan hanya menggunakan satu faktor sahaja iaitu faktor kerumitan program di dalam mendapatkan ketepatan yang lebih baik di dalam memberi keutamaan pada kes ujian bagi fungsi acara berjujukan. Kerumitan program diukur di dalam projek ini menggunakan pengukuran *Branch Coverage Expectation* yang telah dibuktikan secara empirical di dalam kajian sebelum ini. Di dalam projek ini, sebuah aplikasi automatik dibangunkan dengan menggunakan perisian *Visual Basic* untuk mengira nilai *Branch Coverage Expectation*. Metrik *Average Percentage Fault Detection* digunakan untuk membuat penilaian keberkesanan ke atas pendekatan yang dicadangkan. Perisian Eclipse (Junit) digunakan bagi menjalankan pengujian dan membina *fault matrix*. Berdasarkan kepada keputusan yang diperolehi di akhir projek, didapati nilai bagi kes ujian yang telah disusun mengikut keutamaan adalah lebih rendah dari kes ujian yang tidak disusun mengikut keutamaan. Ini menunjukkan bahawa pendekatan yang dicadangkan dengan hanya menggunakan satu faktor tidak

dapat membuktikan bahawa ianya lebih berkesan dalam pendekatan keutamaan kes ujian. Pendekatan yang dicadangkan hendaklah digabungkan bersama faktor-faktor lain supaya keutamaan kes ujian adalah lebih berkesan.

ACKNOWLEDGEMENT

I would like to express my immense gratitude to Almighty of Allah for the wellness and mercies for allowing me to complete this project within the timeline. I also would like to thank my supervisor **Dr. Sa'adah Binti Hassan** for her guidance, motivation and patience during my research. Your good deeds will always be in my thoughts and prayers.

Not to forget to my first supervisor **Dr. Salmi Binti Baharom** for teaching me and give me an idea of the software testing and especially on Test Case Prioritization.

My appreciation goes to UPM lecturers, facilitators and staffs who have been involved directly or indirectly in this project. Besides that I would like to thank my colleagues and my family for endless support in completing this project. Last but not least, I am indebted to Public Service Department (JPA) and Government of Malaysia for giving me an opportunity to further postgraduate study (master level) in Faculty Science Computer and Information Technology, University of Putra Malaysia.

DECLARATION

I hereby confirm that:

- This thesis is my original work.
- Quotations, illustrations and citations have been duly referenced.
- This thesis has not been submitted previously or concurrently for any other degree at any other institutions.
- Intellectual property from the thesis and copyright of thesis are fully-owned by University Putra Malaysia, as according to the University Putra Malaysia (Research) Rules 2012; written permission must be obtained from supervisor and the office of Deputy
- Vice-Chancellor (Research and Innovation) before thesis is published (in the form of written, printed or in electronic form) including books, journals, modules, proceedings, popular writings, seminar papers, manuscripts, posters, reports, lecture notes, learning modules or any other materials as stated in the University Putra Malaysia (Research) Rules 2012.
- There is no plagiarism or data falsification/fabrication in the thesis, and scholarly integrity is upheld as according to the University Putra Malaysia
- (Graduate Studies) Rules 2003 (Revision 2012-2013) and the University Putra
- Malaysia (Research) Rules 2012. The thesis has undergone plagiarism detection software.

Signature: .....
Date:
Name and Matric No: EMYREEMA BINTI JA'AFAR, GS45409

Signature: .....
DR SA'ADAH BINTI HASSAN (Supervisor)

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

Abbreviations	Meaning
TCP	Test Case Prioritization
TC	Test Case
BCE	Branch Coverage Expectation
APFD	Average Percentage Fault Detection
NP	Non-Prioritization
P	Prioritization

CHAPTER 1

INTRODUCTION

1.1 Background

With the number of software users expanding geometrically on daily basis and the proliferation of more advanced software functionalities in business, industry, administration and communications to mention but a few, software engineering development has become more complex and large. Most of the organizations are taking software testing into consideration as an important task that needs to be implemented in the development process. Many researchers have proven that software testing is a critical phase in software development cycle and consumes significant resources in terms of cost, time and effort. The objectives of software testing are to validate the software product, authenticate quality of software product and to discover as much errors as possible in estimated time.

Test case prioritization (TCP) is one of the research areas in software testing. According to (Duggal, 2008), TCP prioritize the test cases so as to increase a test suite's rate of fault detection. Test case prioritization techniques schedule test cases so that the test cases that are higher in priority are executed before the test cases that have a lesser priority. TCP techniques

also can reduce time, cost and detecting more fault. This indeed has created the need to innovate and produce a new or improving algorithm in finding the best solution of TCP so that it can been implemented faster and cost efficient with quality. In recent years, there has been an increasing amount of literature on TCP (Catal & Mishra, 2013) that shows the interest of the researchers in TCP and how much of enhancement of TCP can be research and explored. In TCP there are factors that affect the effectiveness of the TCP technique. The factors are fault, redundancy, complexity, frequency, requirements, time, distance, cost, permutation and others. In this study, we only focus on the complexity factor.

Generally, a complex structure of a system is always a main challenge in software development. This is because the complexity factor always be one of the important factor in determined the cost, time and effort in the software development. It is known, the more complex the system, the greater number of defects will be found. Numerous complexity metrics have been proposed and published in the previous researches in different area of software development. The common complexity measure that been used by the industry and researchers are Mc Cabe's Cyclomatic Complexity (Marchetto, Islam, Asghar, Susi, & Scanniello, 2016), Control Flow Graph (CFG) (C. Y. Huang, Chang, & Chang, 2010) and function point (FP) (Briand, L.C., Morasca, S., Basili, V.R, 1996). These are several complexities metric that can be used in measuring the complexity of the software(Ahmad & Baharom, 2017) .

1.2 Problem Statement

From the previous study (Ahmad & Baharom, 2017), a comparison of software complexity in measuring the complexity of event sequences been made and it is to determine the best metric. The result from the study showed that Unique Complexity Metric (UCM) was the best metric in measuring the complexity of event sequences. However, the researchers mentioned that even though UCM is the best metric, UCM still have some restriction where UCM does not assign the upper and lower bound complexity values in which it will lead to inaccuracy in finding the weightage value for complexity. Since this value will be used for determining the test case prioritization, hence, it must be as accurate as it can. Thus, a different complexity metric measure is proposed to determine the weightage value.

1.3 Objectives and Scope

The objectives of this study are as below:

a) To propose a technique to prioritize test case using complexity factor;
b) To implement the proposed technique;
c) To evaluate and measure the effectiveness of the proposed technique.

The main scope of this project is focusing on determining the test case prioritization technique in the area of event sequences. The factor used to

prioritize the test case is complexity factor. Besides that, this project only uses a java programming codes as a case study.

The evaluation part of this project will use an Average Percentage Fault Detected (APFD) metric in order to prove the effectiveness of the proposed approach.

1.4 Contributions

The proposed approach is expected to give a more accurate measurement of complexity value of the program. This study also emphasis on facilitating software testing team to elicit test case requirements towards producing a reliable software product.

1.6 Thesis Organization

This thesis is organized into six (6) chapters that including this chapter which covers the backgrounds of the study, problem statement, objectives, scope of the research, contribution and thesis structure. Chapter 2 present a literature review by covering existing study on test case prioritization, factors and complexity metric used in the research and also the evaluation metric that been implemented in the research. Chapters 3 present the methodology that explains on theoretical approaches and experimental design that were used

to achieve the research objectives. Proper planning to carry out this study is important to reduce unforeseen problem in the future. Meanwhile, in Chapter 4, the implementation of the proposed approach and prototype development will be covered. It is followed by evaluation and its results, which will be elaborated in Chapter 5. Finally, the last chapter, Chapter 6 summarizes the thesis finding and work that can be done in future.

REFERENCES

Ahmad, J., & Baharom, S. (2017). Comparison of Software Complexity Metrics in Measuring the Complexity of Event Sequences, Information Science and Applications 2017, *424*. https://doi.org/10.1007/978-981-10-4154-9

Chaurasia, G., Agarwal, S., & Gautam, S. S. (2015). Clustering based novel test case prioritization technique. *2015 IEEE Students Conference on Engineering and Systems (SCES)*, 1–5. https://doi.org/10.1109/SCES.2015.7506447

Demarko, T. (1982). *Controlling software projects: management, measurement and estimation*. Controlling software projects Management measurement and estimation Book N.Y.: Yourdon Press1982

Ferrer, J., Chicano, F., & Alba, E. (2013). Estimating software testing complexity. *Information and Software Technology*, *55*(12), 2125–2139. https://doi.org/10.1016/j.infsof.2013.07.007

Henry, S., & Kafura, D. (1981). Software Structure Metrics Based on Information Flow. *IEEE Transactions on Software Engineering*, *SE-7*(5), 510–518. https://doi.org/10.1109/TSE.1981.231113

Hettiarachchi, C., Do, H., & Choi, B. (2016). Risk-based test case prioritization using a fuzzy expert system. *Information and Software Technology*, *69*, 1–15. https://doi.org/10.1016/j.infsof.2015.08.008

Huang, C. Y., Chang, J. R., & Chang, Y. H. (2010). Design and analysis of GUI test-case prioritization using weight-based methods. *Journal of Systems and Software*, *83*(4), 646–659. https://doi.org/10.1016/j.jss.2009.11.703

Huang, R., Chen, J., Towey, D., Chan, A. T. S., & Lu, Y. (2015). Aggregate-strength

interaction test suite prioritization. *Journal of Systems and Software*, *99*, 36–51. https://doi.org/10.1016/j.jss.2014.09.002

Krishnamoorthi, R., & Sahaaya Arul Mary, S. A. (2009). Factor oriented requirement coverage based system test case prioritization of new and regression test cases. *Information and Software Technology*, *51*(4), 799–808. https://doi.org/10.1016/j.infsof.2008.08.007

Kumar, H., & Chauhan, N. (2016). A Novel Approach for Selecting an Effective, 1122–1125. 2016 International Conference on Computing for Sustainable Global Development (INDIACom) .

Marchetto, A., Islam, M. M., Asghar, W., Susi, A., & Scanniello, G. (2016). A Multi-Objective Technique to Prioritize Test Cases. *IEEE Transactions on Software Engineering*, *42*(10), 918–940. https://doi.org/10.1109/TSE.2015.2510633

Nayak, S., Kumar, C., & Tripathi, S. (2016). Effectiveness of prioritization of test cases based on Faults. *2016 3rd International Conference on Recent Advances in Information Technology, RAIT 2016*, 657–662. https://doi.org/10.1109/RAIT.2016.7507977

Noor, T. Bin, & Hemmati, H. (2015). A similarity-based approach for test case prioritization using historical failure data. *2015 IEEE 26th International Symposium on Software Reliability Engineering (ISSRE)*, 58–68. https://doi.org/10.1109/ISSRE.2015.7381799

Srivastava, P. R., & Science, C. (2008). Test case prioritization, 178–181.

Wang, Y., Zhao, X., & Ding, X. (2015). An effective test case prioritization method based on fault severity. *Software Engineering and Service*. Retrieved from

http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=7339162

Chen, T. Y., Kuo, F.-C., Liu, H., & Wong, W. E. (2013). Code Coverage of Adaptive Random Testing. *IEEE Transactions on Reliability*, *62*(1), 226–237. http://doi.org/10.1109/TR.2013.2240898

Eroglu S., Toprak S., Urgan O, MD, Ozge E. Onur, MD, Arzu Denizbasi, MD, Haldun Akoglu, MD, Cigdem Ozpolat, MD, Ebru Akoglu, M. (2012). Test Case Prioritization Incorporating Ordered Sequence of Program Elements. *Saudi Med J*, *33*, 3–8. http://doi.org/10.1073/pnas.0703993104

Ghandehari, L. S. G., Bourazjany, M. N., Lei, Y., Kacker, R. N., & Kuhn, D. R. (2013). Applying Combinatorial Testing to the Siemens Suite. *2013 IEEE Sixth International Conference on Software Testing, Verification and Validation Workshops*, 362–371. http://doi.org/10.1109/ICSTW.2013.47

He, Z., & Bai, C.-G. (2015). GUI Test Case Prioritization by State-coverage Criterion. *2015 IEEE/ACM 10th International Workshop on Automation of Software Test*. http://doi.org/10.1109/AST.2015.11

Islam, M. M., Marchetto, A., Susi, A., & Scanniello, G. (2012). A Multi-Objective Technique to Prioritize Test Cases Based on Latent Semantic Indexing. *2012 16th European Conference on Software Maintenance and Reengineering*, (3), 21–30. http://doi.org/10.1109/CSMR.2012.13

Jiang, B., Zhang, Z., Chan, W. K., & Tse, T. H. (2009). Adaptive Random Test Case Prioritization. *2009 IEEE/ACM International Conference on Automated Software Engineering*, 233–244. http://doi.org/10.1109/ASE.2009.77

Karambir, K., & Kaur, K. (2013). Survey of Software Test Case Generation Techniques, *3*(6), 937–942. Retrieved from http://www.ijarcsse.com/

Kosindrdecha, N., & Daengdej, J. (2010). A Test Case Generation Technique and Process. Proceedings of the First International Workshop on Evolution Support for Model-Based Development and Testing (EMDT'10), 59–66. Retrieved from http://ceur-ws.org/Vol-646/DERIS2010_EMDT2010Proceedings.pdf#page=59

Lu, S., Li, Z., Qin, F., Tan, L., Zhou, P., & Zhou, Y. (2005). BugBench: Benchmarks for Evaluating Bug Detection Tools. Proc of the Workshop on the Evaluation of Software Defect Detection Tools, (3), 1–5. Retrieved from http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Bugbench:+Benchmarks+for+evaluating+bug+detection+tools#0

Ma, Z., & Zhao, J. (2008). Test Case Prioritization Based on Analysis of Program Structure. 2008 15th Asia-Pacific Software Engineering Conference, 471–478. http://doi.org/10.1109/APSEC.2008.63

Mariani, L., Pezzè, M., Riganelli, O., & Santoro, M. (2011). AutoBlackTest: a tool for automatic black-box testing. Software Engineering (ICSE), 2011 33rd International Conference on, 1013–1015.

Petticrew, M., & Roberts, H. (2006). Systematic Reviews in the Social Sciences: A Practical Guide. Cebma.Org. http://doi.org/10.1027/1016-9040.11.3.244

Roongruangsuwan, S., & Daengdej, J. (2010). Test case prioritization techniques. Journal of Theoretical and Applied Information Technology, 45–60.

Saha, R. K. (2015). An Information Retrieval Approach for Regression Test Prioritization Based on Program Changes. Proceeding of the 37th International Conference on Software Engineering (ICSE 2015), 268–279. http://doi.org/10.1109/ICSE.2015.47

Briand, L.C., Morasca, S., Basili, V.R.: Property-based software engineering measurement. IEEE Trans. Softw. Eng. 22(1), 68–86 (1996). doi:10.1109/32.481535

Shete, N. (2014). An Empirical Study of Test Cases in Software Testing, (1978).