HYBRID AND DYNAMIC STATIC CRITERIA MODELS FOR TEST CASE PRIORITIZATION OF WEB APPLICATION REGRESSION TESTING

MOJTABA RAEISI NEJAD

A thesis submitted in fulfilment of the
requirements for the award of the degree of
Doctor of Philosophy (Computer Science)

School of Computing
Faculty of Engineering
Universiti Teknologi Malaysia

OCTOBER 2018

This dissertation is dedicated to my family for their endless affection, support and encouragement.

ACKNOWLEDGEMENT

I bestow all praise to, acclamation and appreciation to Almighty Allah, The Most Merciful and Compassionate, The Most Gracious and the Beneficent, whose bounteous blessings enabled me to pursue and perceive higher ideals of life, who bestowed me good health, courage and knowledge to carry out and complete my work. Special thanks to His Holy Prophet Muhammad (SAWW) who enabled me to recognize my lord and Creator.

I consider it proud privileged to express my deepest gratitude and deep sense of obligation to my reverend supervisor **Assoc. Prof. Dr. Dayang Norhayati Abang Jawawi** that she kept my morale high by their kind suggestions and appreciations. Her motivation leads me to this success and without their sincere and cooperative nature and precious guidance I could never have been able to complete this task.

ABSTRACT

In software testing domain, different techniques and approaches are used to support the process of regression testing in an effective way. The main approaches include test case minimization, test case selection, and test case prioritization. Test case prioritization techniques improve the performance of regression testing by arranging test cases in such a way that maximize fault detection could be achieved in a shorter time. However, the problems for web testing are the timing for executing test cases and the number of fault detected. The aim of this study is to increase the effectiveness of test case prioritization by proposing an approach that could detect faults earlier at a shorter execution time. This research proposed an approach comprising two models: Hybrid Static Criteria Model (HSCM) and Dynamic Weighting Static Criteria Model (DWSCM). Each model applied three criteria: most common HTTP requests in pages, length of HTTP request chains, and dependency of HTTP requests. These criteria are used to prioritize test cases for web application regression testing. The proposed HSCM utilized clustering technique to group test cases. A hybridized technique was proposed to prioritize test cases by relying on assigned test case priorities from the combination of aforementioned criteria. A dynamic weighting scheme of criteria for prioritizing test cases was used to increase fault detection rate. The findings revealed that, the models comprising enhanced of Average Percentage Fault Detection (APFD), yielded the highest APFD of 98% in DWSCM and 87% in HSCM, which have led to improve effectiveness prioritization models. The findings confirmed the ability of the proposed techniques in improving web application regression testing.

ABSTRAK

Dalam bidang ujian perisian, teknik dan pendekatan yang berbeza digunakan untuk menyokong proses ujian regresi dengan cara yang berkesan. Pendekatan utama adalah meminimumkan kes ujian, pemilihan kes ujian, dan mengutamakan kes ujian. Teknik mengutamakan kes ujian meningkatkan prestasi ujian regresi dengan mengatur kes-kes ujian sedemikian rupa sehingga memaksimumkan pengesanan kesalahan dalam masa yang lebih singkat. Walau bagaimanapun, masalah untuk ujian web terletak pada masa untuk melaksanakan kes ujian dan bilangan pengesanan kesalahan. Tujuan kajian ini adalah untuk meningkatkan keberkesanan mengutamakan kes ujian dengan mencadangkan suatu pendekatan yang dapat mengesan kesalahan terdahulu dalam masa pelaksanaan yang lebih singkat. Kajian ini mencadangkan pendekatan yang terdiri daripada dua model: Model Kriteria Statik Hibrid (HSCM) dan Model Kriteria Statik Berpemberat Dinamik (DWSCM). Setiap model menggunakan tiga kriteria: permintaan HTTP yang paling kerap di halaman, panjang rangkaian permintaan HTTP, dan kebergantungan permintaan HTTP. Kriteria ini digunakan bagi mengutamakan kes ujian untuk ujian regresi aplikasi web. HSCM yang dicadangkan menggunakan teknik kluster untuk mengumpulkan kes ujian. Teknik hibrid telah dicadangkan untuk mengutamakan kes-kes ujian yang bergantung kepada keutamaan ujian kes yang diberikan dari gabungan kriteria yang disebutkan di atas. Kriteria skima berpemberat dinamik yang mengutamakan kes ujian digunakan untuk meningkatkan kadar pengesanan kesalahan. Dapatan menunjukkan bahawa model yang terdiri daripada peningkatan Purata Peratus Pengesanan Kesalahan (APFD), menghasilkan APFD tertinggi sebanyak 98% bagi DWSCM dan 87% bagi HSCM, yang telah membawa kepada peningkatan keberkesanan model keutamaan. Dapatan itu mengesahkan keupayaan teknik yang dicadangkan untuk meningkatkan ujian regresi aplikasi web.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ANN	-	Artificial Neural Network
APFD	-	Average Percentage Fault Detection
DWSCM	-	Dynamic Weighting Static Criteria Model
FDR	-	Fault Detection Rate
GA	-	Genetic Algorithm
HSCM	-	Hybrid Static Criteria Model
HTTP	-	Hypertext Transfer Protocol
ICA	-	Imperialist Competitive Algorithm
OTRS	-	Open Technology Real Services
PSO	-	Particle Swarm Optimization
RMC	-	Research Management Center
SOM	-	Self Organizing Map
TCP	-	Test Case Prioritization
UTM	-	Universiti Teknologi Malaysia
XML	-	Extensible Markup Language

LIST OF APPENDICES

CHAPTER 1

INTRODUCTION

1.1 Overview

Across various businesses, web applications serve as critical tools. Failure of these critical tools means the loss of millions of dollars to the organizations that are using them. Web applications offer an organization a worldwide audience (Wang and Zeng, 2016). Most web applications are required to have reliable uptimes without any interruptions. This requires testers to isolate possible software bugs and software engineers to fix the aforementioned bugs immediately, which subsequently result in the release of new versions. Under such circumstances, regression testing is typically initiated to ensure newly released version does not affect other unaltered, working functionalities. Testing is a process which includes all software lifecycle activities comprising both static and dynamic activities; planning, preparation, and evaluation of software, to determine that the built software products are according to specified requirements, requirements of the built software are aligned with clients' purpose, and above all, to detect defects (Sampath *et al.*, 2008).

Test case is a set of input values, execution pre-conditions, expected results, and execution post-conditions, developed for a particular objective or test condition, such as, exercising a particular program path or verifying compliance with a specific requirement (Graham et al.,2008).

Regression tests are executed when some changes are made to an existing application, in order to ensure that the impacts caused by the changes do not negatively affect the rest of the system or the expected behavior of unaltered parts of the software. Such testing is a complicated process for web applications that are built based on modern architectures and technologies. For example, service-oriented architecture introduces simplicities in software development and complexities in software testing process for services like components, which are black box entities generally developed by third parties (Laranjeiro *et al.*, 2012). Any changes executed on those components may negatively affect the rest of the software. In addition, regression testing typically cannot access and verify the content of third-party components (Kumar *et al.*, 2014). Moreover, today's demand for rapid software development also imposes time constraints on testing process.

Software change impact analysis, offers considerable leverage in understanding and implementing change in the system because it provides a detailed examination of the consequences of changes in software. Impact analysis provides visibility into the potential effects of the proposed changes before the actual changes are implemented. The ability to identify the change impact or potential effect will greatly help a maintainer or management to determine appropriate actions to take with respect to change decision, cost, and resource estimates (Ibrahim et al.,2005).

Even with good planning and control, it is possible that the time and budget allocated for the total test, or for a certain test level, are insufficient for executing all planned test cases. In this case, it is necessary to arrange test cases in a way that could reveal faults earlier. This could subsequently inform testers on test cases that could be omitted upon determining test case arrangement that could successfully reveal all faults earlier without having to cover all test cases exhaustively. However, even with a reduced number of executable test cases determined by such arrangement, it must be assured that most critical faults are found. A common technique to arrange test cases is by utilizing test case prioritization. Ideally, test case prioritization allows best possible ordering of test cases to be achieved. In cases where testing ends prematurely, prioritization could enable best possible test case to be achieved, satisfying a certain level of test adequacy. Prioritization is also typically used to determine, as best as possible, that most important test cases are executed first.

1.2 Research Background

In simplest form, all existing test cases in a test suite can be executed without executing sophisticated approaches. However, cost is a concerned, especially when software size gradually expands due to modifications, causing the size the test suite to expand as well. For a significantly large test suite size, executing the entire test suite would be very expensive. Due to economic reasons, software engineers are in a constant pursuit to deploy efficient test techniques to reduce the effort required for regression testing.

New version of application is developed during the life cycle of an application in results of the bug fixes and requirement modification (Onoma *et al.*, 1998). There is potentially a high number of reusable test cases amassed from several application versions that may be applicable to test subsequent versions of the application. However, running all the test cases may involve a significant amount of time. For example, it may take weeks to execute all test cases of an earlier application version (Rothermel *et al.*, 2001; Yoo and Harman, 2012).

Test Case Prioritization (TCP) helps in ordering an optimal sequence of test cases. While prioritization process is not associated with the selection process of test cases, it is assumed that all test cases must be executed. TCP attempts to arrange test cases in a way that if the test process is interrupted or halted early at an arbitrary point, the best ordering, the one that finds more faults, is achieved. TCP was introduced by Wong *et al.* (1999) to arrange best possible sequence of test cases.

General TCP criteria and techniques have been described by Sampath and Sprenkle (2016) in literature. Structural coverage is the most commonly used metric for prioritization (Elbaum *et al.*, 2002; Malishevsky *et al.*, 2002; Rothermel *et al.*, 2001). The logic of this criteria is that faster structural coverage of the whole software code leads to maximum detection of faults in a limited time. Therefore, the aim of this approach to achieve higher fault detection rates, via maximizing structural coverage. Although the most common prioritization technique is about structural coverage in different forms, some prioritization techniques have been presented consisting different criteria (Leon and Podgurski, 2003; Srikanth *et al.*, 2005; Tonella *et al.*, 2006; Yoo *et al.*, 2009).

While several approaches exist for model-based web application test case generation (Andrews *et al.*, 2005; Di Lucca *et al.*, 2002; Kung *et al.*, 2000; Ricca and Tonella, 2001). Modeling web applications is difficult due to their dynamic nature, since model-based techniques are difficult to implement for web application testing.

An approach is presented for TCP to test web applications with different user sessions of previous software versions that have been recorded. Criteria that could best be used to prioritize test cases for a web application is session-based due to the reflection of real user patterns, making the testing process quite realistic (Sampath *et al.*, 2008). The comparison of different criteria is made to prioritize test cases. For instance, for a test case, the total HTTP requests are presented numerically, the values of parameters are covered, and the number of page visits is recorded, as well as, the number of values of parameters. It is shown that while the performance of prioritized test cases is better than those that are not prioritized or ordered properly, no single criterion is ideal (Qu *et al.*, 2007).

User-Session based techniques are new lightweight mechanisms of testing that are applied only for web applications (Hao *et al.*, 2014). In user-session approaches, user interactions with the server are collected and the test cases are extracted using a suitable strategy. Clients' requests are data transported as URLs composing of page addresses while name value pairs are the captured data. These data can be found in log files stored in web servers or cookies left in clients' machines. Captured data about user sessions can be used to generate a set of HTTP requests that can be turned into a real test case. The approach is advantageous as it generates test cases without requiring any awareness of web application's internal structures. Test cases generated by user sessions are not dependent on different technologies that are required for web-based applications (Schwartz and Do, 2016).

1.3 Problem Statement

Different research has been done to improve the performance and the effectiveness of user-session approaches for prioritizing test cases in web applications. Researchers have also tried to reduce the size of generated test suites by utilizing user-session based approaches (Jones and Harrold, 2003). However, limited efforts have been done to prioritize test cases generated by user-session techniques (Leon and Podgurski, 2003; Dawson and Kerr, 2005; Elbaum *et al.*, 2005; Miao *et al.*, 2008).

Existing approaches in prioritization and reduction of test cases still struggle to obtain satisfactory fault detection rate and execution time. In Sampath *et al.* (2008), concept analysis was used such as full and spare lattice, in order to increase the effectiveness of finding faults. Reduction of test cases in web application testing poses a risk to testers in discovering faults as fault revealing test cases might be omitted from the final ordering of test cases.

Since the importance of effectiveness of web application regression testing on increase fault detection rate, prior studies have tried to define appropriate test case prioritization techniques so that the most fault detection rates are achieved. Although the proposed techniques have improved the effectiveness of web application regression testing, they are still confronted by the problem of prioritization and execution test cases. Indeed, effectiveness of web application regression testing is affected by prioritization techniques do not follow the dynamic and hybrid criteria of prioritization test cases.

1.4 Research Questions

This research aims to solve the above mentioned problem by considering the dynamic test case prioritization and effectiveness of web application regression testing. The following main research question is proposed:

How to increase the effectiveness of current web application regression testing technique by dynamic prioritization of test cases?"

In order to answer the main question, the following questions need to be answered:

Question 1: How can the new hybrid criteria and clustering model help to improve average percentage of fault detection rate?

Question 2: How effective is the new model utilizing dynamic weighting of static criteria in improving average percentage of fault detection rate?

Question 3: How to evaluate and analyze the performance of the proposed model?

1.5 Research Goal and Objectives

Suitable prioritizing criteria needs to be determined to arrange test cases in a way that leads to a faster detection of maximum available faults in a regressed version of web application. This research proposes a new approach that merges two approaches that prioritize and cluster test suites, utilizing a dynamic weighting criteria to arrange test cases in web application regression testing. The research aims to improve the effectiveness of web application testing in terms of rate of fault detection and prioritization execution time.

In order to achieve aforementioned goal, the following objectives are constructed:

1. To develop a Hybrid Static Criteria Model (HSCM) using static criteria and clustering techniques.

2. To enhance the effectiveness of test case prioritization through dynamically assigning weights to a static criteria model.

3. To evaluate and analyze the performance of the proposed models and to provide a framework to facilitate the use of the models.

1.6 Research Justification

In this section, the research objectives on modern web application testing have been considered. Changing web application to dynamic type necessitates a speedier testing and a greater effectiveness in revealing faults, due to continuous modifications. Prioritization as one of the recent approaches for regression testing has been used for testing web applications, which aims to arrange test cases in a way that reveal faults in a lesser time. In this research, the authors propose test case clustering, followed by test case prioritization based on three criteria to increase the effectiveness of fault detection rate and optimize testing time. User session has been used for web application testing as this approach is independent of implementing web application and adequate for dynamic web application domain. Therefore, the authors conduct the research utilizing logged user session data in prioritizing test cases for web application testing. The lines of code of large web applications could reach up to millions, thus, object interactions would inevitably involve significant volumes of user interactions. Attributed to this, automated testing becomes an arduous and complicated task, especially, due to a continuous maintenance process and due to a change in user profiles (Kirda *et al.*, 2001).

Many studies in the literature have highlighted two approaches for regression testing such as reduction and prioritization. Reduction of test suite size concerns with producing a smaller number of test cases, which may result in failure to detect faults, due to omission of test cases. Therefore, clustering is proposed instead of reduction as it ensures all test cases are considered, by organizing test cases according to clusters. Furthermore, clustered test cases may be prioritized based on dynamic weighting.

In Appendix A systematic mapping for TCP, only some of the papers related to test case prioritization addressed web applications (Catal and Mishra, 2012). Session-based technique is one of the useful approaches offered for web application testing that still has opportunities for improvement. Unlike other applications, web applications supply usage logs to testers. Test cases drawn from user data are called "user-session based testing" (Elbaum *et al.*, 2005; Sampath *et al.*, 2007). The contents of the test cases contain a series of base requests and pairs of parameter values. Logged user sessions provide knowledge about web interaction of the users. Commonly, cookies information and web server logs are used to generate user-session based test cases.

1.7 Scope

In this thesis, the subject of web application regression testing has been reviewed. This study tries to increase the effectiveness of regression testing with prioritized test cases that have been produced and collected in server side application. Ordering test cases based on new criteria may offer an improved average of fault detection rate.

I. Web application regression testing:

This research estimates average percentage of fault detection rate, which is one of the most important web application testing metrics. This metric informs testers of the capability of an ordered sequence of test cases in revealing faults, often quantified into a single percentage value. Specifically, this metric is known as Average Percentage of Faults Detected (APFD). Further, the research also concerns with prioritization execution time.

II. Test case prioritization:

The proposed test case prioritization technique orders the execution of test cases according to some criterion to satisfy a performance metric considered. The development of the technique includes analysis, design, and implementation.

III. Soft computing techniques:

In order to increase the effectiveness of APFD rate, it is worthwhile to analyze and explore TCP techniques utilizing soft computing techniques. Evolutionary computing, clustering, imperialist competitive algorithm, optimization algorithm, and clustering are among soft computing techniques employed in this research.

1.8 Thesis Outline

This thesis comprises of seven chapters as follows:

Chapter 1 gives an overview on the structure and the nature of the research. The background of the study is elaborated and the research problem is discussed and explained. Moreover, objectives and the main research questions are presented. Finally, the importance of research is justified in this chapter.

Chapter 2 investigates past studies that are related with the current research. Related studies are classified, explained, and analyzed to synthesize gaps and problems.

Chapter 3 describes proposed clustering and TCP techniques comprising static and dynamic techniques utilized in the current research. The structure and the flow of research steps are outlined in this chapter.

Chapter 4 explains the development of technique for prioritizing test cases by hybrid criteria and clustering based technique.

Chapter 5 describes design details, implementation, and experimental results related to the second proposed technique.

Chapter 6 evaluates the performance of the proposed techniques subjected under similar conditions. The chapter includes discussions on the strengths and weaknesses of each technique.

Chapter 7 includes recommendations to be considered prior to utilizing the proposed technique. Furthermore, internal and external threats to research validity are explained. Finally, the contributions of research and future work are elaborated.

REFERENCES

Andrews, A. A., Offutt, J. and Alexander, R. T. (2005). Testing web applications by modeling with FSMs. *Software & Systems Modeling*. 4(3), 326-345.

Arafeen, M. J., and Do, H. (2013). Test case prioritization using requirements-based clustering. *Proceeding of the Sixth International Conference on Software Testing, Verification and Validation (ICST)* (pp. 312-321).

Ardakani, M. M. and Morovvati, M. (2007). Multi-agent system approach for user-session-based testing of web applications. *Lecture Notes in Signal Science, Internet and Education (Ssip'07/Miv'07/Diweb'07)*. 326-331.

Atashpaz-Gargari, E. and Lucas, C. (2007). Imperialist competitive algorithm: an algorithm for optimization inspired by imperialistic competition. *In Evolutionary computation, 2007. CEC 2007. IEEE Congress on* (pp. 4661-4667).

Benedikt, M., Freire, J. and Godefroid, P. (2002). VeriWeb: Automatically testing dynamic web sites. *Proceedings of 11th International World Wide Web Conference.*

Bezdek, J. C. 1981. Pattern recognition with fuzzy objective function algorithms. *Plenum Press, New York, NY.*

Bryce, R. C., Sampath, S. and Memon, A. M. (2011). Developing a single model and test prioritization strategies for event-driven software. *Software Engineering, IEEE Transactions on*. 37(1), 48-64.

Budayan, C., Dikmen, I. and Birgonul, M. T. (2009). Comparing the performance of traditional cluster analysis, self-organizing maps and fuzzy C-means method for strategic grouping. *Expert Systems with Applications*. 36(9), 11772-11781.

Budnik, C. J., Hasling, W. M., McKenna, M. and Subramanyan, R. (2010). Supporting concern-based regression testing and prioritization in a modeldriven environment. *Proceedings of the 2010 Computer Software and Applications Conference Workshops (COMPSACW),* IEEE, 323-328.

Catal, C. and Mishra, D. (2012). Test case prioritization: a systematic mapping study. *Software Quality Journal*. 1-34.

Celebi, M. E. (2015). Partitional clustering algorithms. *Springer.*

Chen, G. Y. and Rogers, J. (2010). Arranging software test cases through an optimization method. *Proceedings of the 2010 Technology Management for Global Economic Growth (PICMET),* IEEE, 1-5.

Chen, S. B., Miao, H. K. and Song, B. (2012). AGT4W: Automatic generating tests for web applications. *Frontiers in Computer Education*. 133, 885-892.

Chen, X., Gu, Q., Wang, X., Li, A. and Chen, D. (2009). A hybrid approach to build prioritized pairwise interaction test suites. *Proceedings of the 2009 Computational Intelligence and Software Engineering, 2009. CiSE 2009. International Conference on*: IEEE, 1-4.

Dawson, S. and Kerr, K. (2005). Testing web applications. *Dr Dobbs Journal*. 30(9), 60-63.

de Jesus, F. R., de Vasconcelos, L. G. and Baldochi, L. A. (2015). Leveraging task-based data to support functional testing of web applications. *Proceeding of 2015 Symposium on Applied Computing, Vols. I-Ii.* 10.1145/2695664.2695917, 783-790.

Di Lucca, G., Fasolino, A. R. and Tramontana, P. (2006). A technique for reducing user session data sets in web application testing. *Proceedings of the 2006 Web Site Evolution, 2006. WSE'06. Eighth IEEE International Symposium on*: IEEE, 7-13.

Di Lucca, G. A., Fasolino, A. R., Faralli, F. and De Carlini, U. (2002). Testing web applications. *Proceedings of the 2002 Software Maintenance, 2002. Proceedings. International Conference on*: IEEE, 310-319.

Di Nardo, D., Alshahwan, N., Briand, L., and Labiche, Y. (2015). Coverage ‐ based regression test case selection, minimization and prioritization: A case study on an industrial system. *Software Testing, Verification and Reliability*, 25(4), 371-396.

Do, H., Mirarab, S., Tahvildari, L. and Rothermel, G. (2010). The effects of time constraints on test case prioritization: A series of controlled experiments. *IEEE Transactions on Software Engineering*. 36(5), 593-617.

Dong, L. L., Li, H. and Zhang, S. M. (2013). Workflow-based modeling of web application and automatically generating test sequences. *Proceeding of the 2013 13th International Conference on Quality Software (Qsic)*. 10.1109/Qsic.2013.37, 378-381.

Elbaum, S., Gable, D. and Rothermel, G. (2001). Understanding and measuring the sources of variation in the prioritization of regression test suites. *Proceedings of the 2001 Software Metrics Symposium, 2001. METRICS 2001. Proceedings. Seventh International*: IEEE, 169-179.

Elbaum, S., Karre, S. and Rothermel, G. (2003). Improving web application testing with user session data. *Proceedings of the 2003 Proceedings of the 25th International Conference on Software Engineering*: *IEEE Computer Society*, 49-59.

Elbaum, S., Malishevsky, A. G., and Rothermel, G. (2007). Prioritizing test cases for regression testing. (Vol. 25, No. 5, pp. 102-112). ACM.

Elbaum, S., Malishevsky, A. G. and Rothermel, G. (2002). Test case prioritization: A family of empirical studies. *IEEE Transactions on Software Engineering*. 28(2), 159-182.

Elbaum, S., Rothermel, G., Karre, S. and Fisher, I. (2005). Leveraging user-session data to support web application testing. *IEEE Transactions on Software Engineering*. 31(3), 187-202.

Fang, C., Chen, Z., Wu, K., and Zhao, Z. (2014). Similarity-based test case prioritization using ordered sequences of program entities. Software Quality Journal, 22(2), 335-361.

Fazlalizadeh, Y., Khalilian, A., Azgomi, M. A. and Parsa, S. (2009). Incorporating historical test case performance data and resource constraints into test case prioritization tests and proofs (pp. 43-57) Springer.

Ferrucci, F., Gravino, C., Oliveto, R., and Sarro, F. (2010). Genetic programming for effort estimation: an analysis of the impact of different fitness functions. *Proceeding of the Search Based Software Engineering (SSBSE), 2010 Second International Symposium on (pp. 89-98)*. IEEE.

Gan, G., Ma, C. and Wu, J. (2007). Data clustering: theory, algorithms, and applications. (Vol. 20) *Society for Industrial and Applied Mathematics*.

Garg, D. and Datta, A. (2012). Test case prioritization due to database changes in web applications. *Proceedings of the 2012 Software Testing, Verification and Validation (ICST)*. IEEE, 726-730.

Garima, S. and Manju, K. (2016). Dual Security Testing Model for Web Applications. *International Journal of Advanced Computer Science and Applications*. 7(2), 185-191.

Garousi, V., Mesbah, A., Betin-Can, A. and Mirshokraie, S. (2013). A Systematic Mapping Study of Web Application Testing. *Information and Software Technology*.

Goren, H. G., Tunali, S., and Jans, R. (2010). A review of applications of genetic algorithms in lot sizing. *Journal of Intelligent Manufacturing,* 21(4), 575-590.

Graham, D., Van Veenendaal, E., and Evans, I. (2008). Foundations of software testing: ISTQB certification. *Cengage Learning EMEA*.

Hao, D., Zhang, L. M., Zhang, L., Rothermel, G. and Mei, H. (2014). A Unified Test Case Prioritization Approach. *ACM Transactions on Software Engineering and Methodology*. 24(2).

Harman, M., and Alshahwan, N. (2008). Automated session data repair for web application regression testing. *Proceeding of the Software Testing, Verification, and Validation, 2008 1st International Conference on (pp. 298-307).*

Hossain, M., Do, H., & Eda, R. (2014, March). Regression testing for web applications using reusable constraint values. *Proceeding of the Software Testing, Verification and Validation Workshops (ICSTW), 2014 IEEE Seventh International Conference on (pp. 312-321)*. IEEE.

Huang, C.Y., Chang, J.-R. and Chang, Y.H. (2010a). Design and analysis of GUI test-case prioritization using weight-based methods. *Journal of Systems and Software*. 83(4), 646-659.

Huang, Y.C., Huang, C. Y., Chang, J.R. and Chen, T.Y. (2010b). Design and analysis of cost-cognizant test case prioritization using genetic algorithm with test history. *Proceedings of the 2010b Computer Software and Applications Conference (COMPSAC). IEEE*, 413-418.

Huidrom, N., and Bagoria, N. (2013). Clustering techniques for the identification of web user session. *International Journal of Scientific and Research Publications*, 3(1), 1-4.

Ibrahim, S., Idris, N. B., Munro, M., and Deraman, A. (2005). A requirements traceability to support change impact analysis. *Asian Journal of Information Tech*, 4(4), 345-355.

Jain, A. K., Murty, M. N. and Flynn, P. J. (1999). Data clustering: a review. *ACM Computing Surveys (CSUR)*. 31(3), 264-323.

Jain, A. K., and Dubes, R. C. (1988). Algorithms for clustering data.

Jeffrey, D. and Gupta, N. (2006). Test case prioritization using relevant slices. *Proceedings of the 2006 Computer Software and Applications Conference.* IEEE, 411-420.

Jiang, B., Zhang, Z., Chan, W. and Tse, T. (2009a). Adaptive random test case prioritization. *Proceedings of the 2009a Automated Software Engineering, ASE'09.* IEEE, 233-244.

Jiang, B., Zhang, Z., Tse, T. and Chen, T. Y. (2009b). How well do test case prioritization techniques support statistical fault localization. *Proceedings of the 2009b Computer Software and Applications Conference, COMPSAC'09.* IEEE, 99-106.

Jiao, J. A. (2009). applying user session based technique in web applications testing. *Proceeding of the 10th International Conference on Intelligent Technologies*. 36-38.

Jones, J. A. and Harrold, M. J. (2003). Test-suite reduction and prioritization for modified condition/decision coverage. *IEEE Transactions on Software Engineering*. 29(3), 195-209.

Kavitha, R., Kavitha, V. and Kumar, N. S. (2010). Requirement based test case prioritization. *Proceedings of the 2010 Communication Control and Computing Technologies (ICCCCT).* IEEE, 826-829.

Khatibsyarbini, M., Isa, M. A., Jawawi, D. N., and Tumeng, R. (2017). Test case prioritization approaches in regression testing: A systematic literature review. *Information and Software Technology.*

Kim, J.-M. and Porter, A. (2002). A history-based test prioritization technique for regression testing in resource constrained environments. *Proceedings of the 2002 Software Engineering, ICSE 2002.* IEEE, 119-129.

Kim, S. and Baik, J. (2010). An effective fault aware test case prioritization by incorporating a fault localization technique. *Proceedings of the 2010 Proceedings of the 2010 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement*: ACM, 5.

Kirda, E., Jazayeri, M., Kerer, C. and Schranz, M. (2001). Experiences in engineering flexible web services. *Multimedia, IEEE*. 8(1), 58-65.

Kohonen, T. (1984). Self-Organization and Associative Maps.

Konsaard, P. and Ramingwong, L. (2015). Total coverage based regression test case prioritization using genetic algorithm. *Proceeding of the 2015 12th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (Ecti-Con).*

Korel, B., Koutsogiannakis, G. and Tahat, L. H. (2008). Application of system models in regression test suite prioritization. *Proceedings of the 2008 Software Maintenance, 2008. ICSM 2008*. IEEE, 247-256.

Krishnamoorthi, R. and Sahaaya Arul Mary, S. (2009). Factor oriented requirement coverage based system test case prioritization of new and regression test cases. *Information and Software Technology*. 51(4), 799-808.

Kumar, K. S. V., Lallu, A. and Mathew, S. (2014). An efficient approach for distributed regression testing of object oriented programs. 10.1145/2660859.2660944, 1-7.

Kung, D. C., Liu, C.-H. and Hsia, P. (2000). An object-oriented web test model for testing web applications. *Proceedings of the 2000 Quality Software, 2000. Proceedings. First Asia-Pacific Conference on*: IEEE, 111-120.

Laranjeiro, N., Vieira, M. and Madeira, H. (2012). A robustness testing approach for SOAP Web services. *Journal of Internet Services and Applications*. 3(2), 215-232.

Leon, D. and Podgurski, A. (2003). A comparison of coverage-based and distribution-based techniques for filtering and prioritizing test cases. *Proceedings of the 2003 Software Reliability Engineering, 2003. ISSRE 2003*. IEEE, 442-453.

Li, J. h. and Xing, D. d. (2011). user session data based web applications test with cluster analysis. *Advanced Research on Computer Science and Information Engineering* (pp. 415-421) Springer.

Li, S., Bian, N., Chen, Z., You, D. and He, Y. (2010). A simulation study on some search algorithms for regression test case prioritization. *Proceedings of the 2010 Quality Software (QSIC)*. IEEE, 72-81.

Liu, C. H. (2006). Data flow analysis and testing of JSP-based Web applications. *Information and Software Technology*. 48(12), 1137-1147.

Liu, Y., Wang, K., Wei, W., Zhang, B. and Zhong, H. (2011). User-session-based test cases optimization method based on agglutinate hierarchy clustering. *Proceedings of the 2011 Internet of Things (iThings/CPSCom)*. IEEE, 413-418.

Luo, X., Ping, F. and Chen, M. H. (2009). Clustering and tailoring user session data for testing web applications. *Proceedings of the 2009 Software Testing Verification and Validation, 2009. ICST'09*. IEEE, 336-345.

Malishevsky, A. G., Rothermel, G. and Elbaum, S. (2002). Modeling the cost-benefits tradeoffs for regression testing techniques. *Proceedings of the 2002 Software Maintenance, 2002. Proceedings*. IEEE, 204-213.

Mansour, N. and El-Fakih, K. (1999). Simulated annealing and genetic algorithms for optimal regression testing. *Journal of Software Maintenance*. 11(1), 19-34.

Mariani, L., Papagiannakis, S. and Pezze, M. (2007). Compatibility and regression testing of COTS-component-based software. *Proceedings of the 2007 Software Engineering, 2007. ICSE 2007*. IEEE, 85-95.

Miao, H. K., Qian, Z. S. and Song, B. (2008). Towards automatically generating test paths for web application testing. *Proceedings of the TASE 2008: Second IFIP/IEEE International Symposium on Theoretical Aspects of Software Engineering*. 10.1109/Tase.2008.26, 211-218.

Mirarab, S. and Tahvildari, L. (2007). A prioritization approach for software test cases based on Bayesian networks. *Fundamental Approaches to Software Engineering* (pp. 276-290) Springer.

Onoma, A. K., Tsai, W.T., Poonawala, M. and Suganuma, H. (1998). Regression testing in an industrial environment. *Communications of the ACM*. 41(5), 81-86.

Park, H., Ryu, H. and Baik, J. (2008). Historical value-based approach for cost-cognizant test case prioritization to improve the effectiveness of regression testing. *Proceedings of the 2008 Secure System Integration and Reliability Improvement, 2008. SSIRI'08.* IEEE, 39-46.

Peng, X. and Lu, L. (2011). A new approach for session-based test case generation by GA. *Proceedings of the 2011 Communication Software and Networks (ICCSN).* IEEE, 91-96.

Petersen, K., Feldt, R., Mujtaba, S. and Mattsson, M. (2008). Systematic mapping studies in software engineering. *Proceedings of the 2008 12th International Conference on Evaluation and Assessment in Software Engineering*, 1.

Purohit, G. N. (2015, February). A Schema Support for Selection of Test Case Prioritization Techniques. *Proceeding of the Advanced Computing and Communication Technologies (ACCT), 2015 Fifth International Conference on (pp. 547-551).* IEEE.

Qu, X. (2009). Configuration aware prioritization techniques in regression testing. *Proceedings of the 2009 Software Engineering-Companion Volume, 2009. ICSE-Companion 2009.* IEEE, 375-378.

Ricca, F. and Tonella, P. (2001). Analysis and testing of web applications. *Proceedings of the 2001 Proceedings of the 23rd international conference on Software engineering*: IEEE Computer Society, 25-34.

Rothermel, G., Untch, R. H., Chu, C. and Harrold, M. J. (2001). Prioritizing test cases for regression testing. *IEEE Transactions on Software Engineering.* 27(10), 929-948.

Ruspini, E. H. (1969). A new approach to clustering. *Information and control*, 15(1), 22-32.

Sabharwal, S., Sibal, R. and Sharma, C. (2010). Prioritization of test case scenarios derived from activity diagram using genetic algorithm. *Proceedings of the 2010 Computer and Communication Technology (ICCCT),* IEEE, 481-485.

Sahu, M. and Mohapatra, D. P. (2014). Data flow testing of CGI based web applications. *Proceeding of the 2014 International Conference on Information Technology (ICIT).*, 106-111.

Sampath, S. and Bryce, R. C. (2012). Improving the effectiveness of test suite reduction for user-session-based testing of web applications. *Information and Software Technology.* 54(7), 724-738.

Sampath, S., Bryce, R. C., Viswanath, G., Kandimalla, V. and Koru, A. G. (2008). Prioritizing user-session-based test cases for web applications testing.

Proceedings of the 2008 Software Testing, Verification, and Validation. IEEE, 141-150.

Sampath, S. and Sprenkle, S. (2016). Advances in web application testing, 2010–2014. 101, 155-191.

Sampath, S., Sprenkle, S., Gibson, E., Pollock, L. and Greenwald, A. S. (2007). Applying concept analysis to user-session-based testing of web applications. *IEEE Transactions on Software Engineering*. 33(10), 643-658.

Sapna, P. and Mohanty, H. (2009a). Prioritization of scenarios based on UML activity diagrams. *Proceedings of the 2009a Computational Intelligence, Communication Systems and Networks, 2009. CICSYN'09*. IEEE, 271-276.

Sapna, P. and Mohanty, H. (2009b). Prioritizing Use Cases to aid ordering of Scenarios. *Proceedings of the 2009b Computer Modeling and Simulation, 2009*. IEEE, 136-141.

Schwartz, A. and Do, H. (2016). Cost-effective regression testing through adaptive test prioritization strategies. *Journal of Systems and Software*. 115, 61-81.

Simons, C. and Paraiso, E. C. (2010). Regression test cases prioritization using Failure Pursuit Sampling. *Proceedings of the 2010 Intelligent Systems Design and Applications (ISDA)*. IEEE, 923-928.

Sprenkle, S., Gibson, E., Sampath, S. and Pollock, L. (2005). Automated replay and failure detection for web applications. *Proceedings of the 2005 Proceedings of the 20th IEEE/ACM international Conference on Automated software engineering*: ACM, 253-262.

Srikanth, H., Cohen, M. B. and Qu, X. (2009). Reducing field failures in system configurable software: cost-based prioritization. *Proceedings of the 2009 Software Reliability Engineering, 2009. ISSRE'09*. IEEE, 61-70.

Srikanth, H., Williams, L. and Osborne, J. (2005). System test case prioritization of new and regression test cases. *Proceedings of the 2005 Empirical Software Engineering, 2005*. IEEE, 10 pp.

Srivastava, A. and Thiagarajan, J. (2002). Effectively prioritizing tests in development environment. *Proceedings of the 2002 ACM SIGSOFT Software Engineering Notes*: ACM, 97-106.

Srivastava, P. R. (2008). Model for optimizing software testing period using non homogenous poisson process based on cumulative test case prioritization.

Proceedings of the 2008 TENCON 2008-2008 IEEE Region 10 Conference: IEEE, 1-6.

Srivastava, P. R., Ray, M., Dermoudy, J., Kang, B. H. and Kim, T. H. (2009). Test case minimization and prioritization using CMIMX technique. *Advances in Software Engineering*. 59, 25-33.

Tarhini, A., Ismail, Z. and Mansour, N. (2008). Regression testing web applications. *2008 International Conference on Advanced Computer Theory and Engineering*. 10.1109/ICACTE.2008.82, 902-906.

Tonella, P., Avesani, P. and Susi, A. (2006). Using the case-based ranking methodology for test case prioritization. *Proceedings of the 2006 Software Maintenance, 2006. ICSM'06*. IEEE, 123-133.

Wang, X. L. and Zeng, H. W. (2014). Dynamic test case prioritization based on multi-objective. *2014 15th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing*. 441-446.

Wang, X. L. and Zeng, H. W. (2016). History-based dynamic test case prioritization for requirement properties in regression testing. *Proceeding of the International Workshop on Continuous Software Evolution and Delivery, CSED 2016*. 10.1145/2896941.2896949, 41-47.

Wieringa, R., Maiden, N., Mead, N. and Rolland, C. (2006). Requirements engineering paper classification and evaluation criteria: A proposal and a discussion. *Requirements Engineering*. 11(1), 102-107.

Wong, W. E., Horgan, J. R., London, S. and Agrawal, H. (1997). A study of effective regression testing in practice. *Proceedings of the 1997 Eighth International Symposium On Software Reliability Engineering*. IEEE, 264-274.

Wong, W. E., Horgan, J. R., Mathur, A. P. and Pasquini, A. (1999). Test set size minimization and fault detection effectiveness: A case study in a space application. *Journal of Systems and Software*. 48(2), 79-89.

Xu, D. and Ding, J. (2010). Prioritizing state-based aspect tests. *Proceedings of the 2010 Software Testing, Verification and Validation (ICST)*. IEEE, 265-274.

Xu, L., Xu, B., He, Y., Chen, H., and Zhu, Q. (2005, November). Research on the analysis and measurement for testing results of web applications. *Proceeding of the Cyber Worlds, 2005*. (pp. 7-pp). IEEE.

Xu, R. and Wunsch, D. (2005). Survey of clustering algorithms. *IEEE Transactions on Neural Networks*. 16(3), 645-678.

Xue, Y. F. and Mao, Y. G. (2015). Test case prioritization based on clustering analysis. *Proceedings of the 4th International Conference on Mechatronics, Materials, Chemistry and Computer Engineering 2015 (ICMMCCE 2015).* 39, 1449-1453.

Yang, J.-T., Huang, J.-L., Wang, F.-J. and Chu, W. C. (2002). Constructing and object-oriented architecture for web application testing. *Journal of Information Science and Engineering*. 18(1), 59-84.

Yoo, S. and Harman, M. (2012). Regression testing minimization, selection and prioritization: A survey. *Software Testing, Verification and Reliability*. 22(2), 67-120.

Yoo, S., Harman, M., Tonella, P. and Susi, A. (2009). Clustering test cases to achieve effective and scalable prioritisation incorporating expert knowledge. *Proceedings of the 2009 Proceedings of the eighteenth international symposium on Software testing and analysis*: ACM, 201-212.

Yu, Y. T., and Lau, M. F. (2012). Fault-based test suite prioritization for specification-based testing. *Information and Software Technology*, 54(2), 179-202.

Yuan, J. and Jiang, Z. (2010). Constructing prioritized interaction test suite with interaction relationship. *Proceedings of the 2010 Education Technology and Computer Science (ETCS).* IEEE, 181-184.

Zahn, C. T. (1971). Graph-theoretical methods for detecting and describing gestalt clusters. *IEEE Transactions on computers*, 68-86.

Zamli, K. Z., Hassin, M. H. M., and Al-Kazemi, B. (2014). tReductSA–Test redundancy reduction strategy based on simulated annealing. *Proceeding of the International Conference on Intelligent Software Methodologies, Tools, and Techniques* (pp. 223-236). Springer.

Zhang, C., Groce, A. and Alipour, M. A. (2014). Using test case reduction and prioritization to improve symbolic execution. *Proceedings of the 2014 Proceedings of the 2014 International Symposium on Software Testing and Analysis*: ACM, 160-170.

Zhang, Z., Huang, J., Zhang, B., Lin, J. and Chen, X. (2010). regression test generation approach based on tree-structured analysis. *Proceedings of the 2010 Computational Science and Its Applications (ICCSA).* IEEE, 244-249.

APPENDIX A

1. Systematic Mapping

Our study was conducted according to the guidelines provided by (Petersen *et al.*, 2008). Their five-step process, included here as Figure 1, can be described as follows:

2. Research Question

Research questions define the scope and focus of a systematic mapping. This study was motivated by three research questions:

I. What are the states of art and future works in testing web application?
II. What type of papers published on web application testing?
III. What techniques are used for web application testing?

This question was included specifically to find all the testing techniques that are used for web application testing and estimate which one is more effective based on popularity.