TEST CASE PRIORITIZATION USING SWARM INTELLIGENCE ALGORITHM TO IMPROVE FAULT COVERAGE AND TIME FOR WEB APPLICATIONS

TABLE OF CONTENTS

TITLE **i**

TABLE OF CONTENTS **ii**

CHAPTER 1 INTRODUCTION **1**

1.1 Research Background 1

1.2 Problem Statement 3

1.3 Objectives and Aims 4

1.4 Scope of Research 4

1.5 Research Significance 5

REFERENCES **6**

CHAPTER 1

INTRODUCTION

1.1 Research Background

Software testing is one of the important and primary parts of software quality assurance (SQA). Testing a product or application is the main stage of Software Development Life Cycle (SDLC). It assumes an extremely fundamental part in choosing the quality and usefulness of the product and guarantees that the finished result is according to the customers' interest. The Testing of the product is done to identify flaws and mistakes in its functionality, based on which the performance and features of the product is enhanced further. The least known aspect of the software development process is software testing. Software testing is performed repeatedly, and it is often done in a rush due to time constraints and limited resources (Khatibsyarbini, Isa, Jawawi, Hamed, & Mohamed Suffian, 2019).

Test Case Prioritization (TCP) application appears to enhance the test viability in software testing activity (Khatibsyarbini et al., 2019). Test Case Prioritization (TCP) is a technique whereby each test cases are assigned a priority. TCP orders the test cases on some properties so that highly ranked test cases execute at a higher priority (Bajaj & Sangwan, 2019). It enables a way to execute highly significant test cases initially and produce the ideal result, such as identifying flaws prior and giving feedback to the testers. It likewise assists with tracking down the ideal permutation of a progression of test cases and could be executed in a reliable manner (Khatibsyarbini, Isa, Jawawi, & Tumeng, 2018). TCP techniques are very popular in the industry and are the focus of the study (Prado Lima & Vergilio, 2020). The number of publications related to the test case prioritization is still growing in recent years which includes the interest in the application of artificial intelligence, machine learning in TCP (Mece, Binjaku, & Paci, 2020). Most researches conduct the study by implementing metaheuristic algorithm to detect faults and time execution performance (Padmnav, Pahwa, Singh, & Bansal, 2019).

Metaheuristic is a significant level structure to create heuristic optimization algorithms that may give an adequate proper solution for an optimization problem particularly with inadequate data (Padmnav et al., 2019). When compared to optimization algorithms, iterative approaches, or simple heuristics, metaheuristics enables to find good solutions with less computational effort. Evolutionary computation, genetic algorithms and particle swarm optimization are population based metaheuristic (Soghrati & Moeini, 2020).

Swarm Intelligence (SI) is a new research area inspired by collective intelligence of social insects or animals in groups, such as ant colonies, bee colonies, bird flocking, fish schooling and animal herding (Chakraborty & Kar, 2017). The term collective intelligence was first used to describe how insects and animals self-organize and behave intelligently (Kumar, Kumar, & Jarial, 2017). Recent studies in TCP problem implemented swarm intelligence algorithms like Firefly Algorithm (FA) (Khatibsyarbini et al., 2019), Ant Colony Optimization (ACO) (Panwar, Tomar, Harsh, & Siddique, 2018) and Artificial Bee Colony (ABC) (Padmnav et al., 2019) and etc. From these studies, the results shown to be that the algorithms implemented was a success for fault coverage. Hence, this research is going to focus on two swarm intelligence algorithms which are ABC and ACO to compare which is a better algorithm in terms of fault coverage and time execution.

ABC is a metaheuristic and swarm intelligence algorithm proposed by Karaboga in 2015 (Huo & Liu, 2018). ABC is a bee-based optimization strategy that takes into account how groups of bees behave in the real world. In the actual Bee colony, there are mainly three kinds of bees which are Scout, Worker and Onlooker Bees (Kabra & Birajdar, 2016). A possible arrangement of the optimization problem is addressed by the placement of a food source, and the consistency of the solution is referred to as fitness (Xue, Jiang, Zhao, & Ma, 2018). The ABC algorithm is an evolutionary optimization method that is simple, efficient, reliable, and robust. ABC has emerged as a possible appliance for resolving both local and global optimization issues which can also be implemented in TCP (Wen, Shao, Fang, & Xue, 2015).

ACO algorithm is also a metaheuristic and swarm intelligence algorithm inspired by a real-life ant monitoring behaviour. This algorithm is an example of how computer scientists are stimulating real ants in artificial ants (Agrawal & Kaur, 2018).

While each ant is visually impaired, fragile, and insignificant on its own, the province of ants demonstrates complex activities by allowing them to assist one another. One of them is the ability to lay down special chemicals called pheromones to find the quickest path to a food source or other interesting landmark (Srivastava et al., 2016). More pheromone deposits on a path increase the likelihood that the path will be followed based on the maximum pheromone deposit from the start node, and the path will be evaluated for optimality (S & S S, 2020). The ACO algorithm is suitable to identify faults early as it has a special technique to find the quickest path.

Thus, in this research, we proposed a study on TCP using ABC and ACO algorithms to improve fault coverage and time execution for web applications. The results will then be compared to determine the effectiveness of both ABC and ACO algorithms.

1.2 Problem Statement

Test Case Prioritization (TCP) is a technique that assigns a priority to each test case. Priority is assigned based on a set of criteria, and the highest-priority test cases are scheduled first. The criterion might be that the test case with the fastest fault coverage receives the highest priority. This method has the advantage of neither discarding or permanently removing test cases from the test suite (Raamesh, 2016).

Recently, several researches performed a study on test case prioritization by implementing Genetic Algorithm (GA), Firefly Algorithm (FA), Bat Algorithm (BA), etc. The study outcome showed various results based on the domains used as well as the algorithms. The Firefly Algorithm showed a good response however, it slightly outperformed in terms of execution time (Khatibsyarbini et al., 2019). Besides that, the research that was carried out to test the effectiveness between GA and ACO showed a positive outcome that ACO was a better algorithm when implemented in time-constrained environment (Panwar et al., 2018). Hence, this research will be carried out in order to test and compare the effectiveness of two different algorithms under Swarm Intelligence, where ABC and ACO algorithms will be implemented in the study. The test will be executed to find the fault coverage and time execution when both algorithms are implemented as these are the curial aspects in software testing to ensure good quality products are produced within the timeline.

Besides that, this research also will be carried out on web application test cases. As web applications or dynamic websites are commonly used by the broad population around the world for information, e-commerce, and e-learning, this is to test the effectiveness of both algorithms on the test cases (Pradesh & Pradesh, 2017). This would help to improve the fault coverage and time execution as these are the most crucial aspects in software testing to maintain good quality assurance within a proposed period of time.

1.3 Objectives and Aims

The aim of the research is to implement swarm intelligence algorithm for test case prioritization to improve fault coverage and time execution for web applications. To achieve the aim, following are the objectives of this research:

i To implement Artificial Bee Colony (ABC) and Ant Colony Optimization (ACO) in test case prioritization for web applications.

ii To compare the effectiveness of ABC and ACO algorithm to improve the fault coverage and time execution for web application.

1.4 Scope of Research

The scopes of the study are:

i Conduct the study by using web application test cases datasets.

ii The algorithms that will be used are artificial bee colony and ant colony optimization.

iii To improve fault coverage and the time of execution.

1.5 Research Significance

This study conducts a comparative analysis between two TCP algorithms under swarm intelligence, ABC and ACO algorithms. The significant of the proposed algorithm is to evaluate the effectiveness of the algorithms implemented in improving fault coverage and time execution. The result was then compared to identify the better algorithm. To the best of our knowledge, there is no attempts made by researchers previously by comparing two algorithms under swarm intelligence for web applications. Hence, research gives a new contribution to TCP.

REFERENCES

Agrawal, A. P., & Kaur, A. (2018). A comprehensive comparison of ant colony and hybrid particle swarm optimization algorithms through test case selection. *Advances in Intelligent Systems and Computing*, *542*, 397–405. https://doi.org/10.1007/978-981-10-3223-3_38

Bajaj, A., & Sangwan, O. P. (2019). A Systematic Literature Review of Test Case Prioritization Using Genetic Algorithms. *IEEE Access*, *7*, 126355–126375. https://doi.org/10.1109/ACCESS.2019.2938260

Chakraborty, A., & Kar, A. K. (2017). Swarm intelligence: A review of algorithms. *Modeling and Optimization in Science and Technologies*, *10*, 475–494. https://doi.org/10.1007/978-3-319-50920-4_19

Huo, J., & Liu, L. (2018). An optimization framework of multiobjective artificial bee colony algorithm based on the MOEA framework. *Computational Intelligence and Neuroscience*, *2018*. https://doi.org/10.1155/2018/5865168

Kabra, M. P., & Birajdar, M. J. (2016). A SYSTEMATIC REVIEW ON ARTIFICIAL BEE COLONY OPTIMIZATION TECHNIQUE, *1*(June), 77–88.

Khatibsyarbini, M., Isa, M. A., Jawawi, D. N. A., Hamed, H. N. A., & Mohamed Suffian, M. D. (2019). Test Case Prioritization Using Firefly Algorithm for Software Testing. *IEEE Access*, *7*, 132360–132373. https://doi.org/10.1109/ACCESS.2019.2940620

Khatibsyarbini, M., Isa, M. A., Jawawi, D. N. A., & Tumeng, R. (2018). Test case prioritization approaches in regression testing: A systematic literature review. *Information and Software Technology*, *93*, 74–93. https://doi.org/10.1016/j.infsof.2017.08.014

Kumar, A., Kumar, D., & Jarial, S. K. (2017). A review on artificial bee colony algorithms and their applications to data clustering. *Cybernetics and Information Technologies*, *17*(3), 3–28. https://doi.org/10.1515/cait-2017-0027

Mece, E. K., Binjaku, K., & Paci, H. (2020). The Application Of Machine Learning In Test Case Prioritization - A Review. *European Journal of Electrical Engineering and Computer Science*, *4*(1), 0–9. https://doi.org/10.24018/ejece.2020.4.1.128

Padmnav, P., Pahwa, G., Singh, D., & Bansal, S. (2019). Test case prioritization based on historical failure patterns using ABC and GA. *Proceedings of the 9th*

International Conference On Cloud Computing, Data Science and Engineering, Confluence 2019, 293–298. https://doi.org/10.1109/CONFLUENCE.2019.8776936

Panwar, D., Tomar, P., Harsh, H., & Siddique, M. H. (2018). Improved Meta-Heuristic Technique for Test Case Prioritization. *Advances in Intelligent Systems and Computing*, *583*, 647–664. https://doi.org/10.1007/978-981-10-5687-1_58

Pradesh, U., & Pradesh, U. (2017). Test case prioritisation during web application testing Munish Khanna * and Naresh Chauhan Dilip Kumar Sharma Abhishek Toofani, *56*(3), 230–243.

Prado Lima, J. A., & Vergilio, S. R. (2020). Test Case Prioritization in Continuous Integration environments: A systematic mapping study. *Information and Software Technology*, *121*(July 2019), 106268. https://doi.org/10.1016/j.infsof.2020.106268

Raamesh, L. (2016). Test case prioritization. *International Journal of Applied Engineering Research*, *10*(13), 32917–32922.

S, S. S., & S S, V. C. (2020). *An Ant Colony Optimization Algorithm Based Automated Generation of Software Test Cases*. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* (Vol. 12145 LNCS). Springer International Publishing. https://doi.org/10.1007/978-3-030-53956-6_21

Soghrati, F., & Moeini, R. (2020). Deriving optimal operation of reservoir proposing improved artificial bee colony algorithm: Standard and constrained versions. *Journal of Hydroinformatics*, *22*(2), 263–280. https://doi.org/10.2166/hydro.2019.125

Srivastava, P. R., Ramachandran, V., Kumar, M., Talukder, G., Tiwari, V., & Sharma, P. (2016). Generation of test data using meta heuristic approach. *IEEE Region 10 Annual International Conference, Proceedings/TENCON*, 1–6. https://doi.org/10.1109/TENCON.2008.4766707

Wen, X., Shao, L., Fang, W., & Xue, Y. (2015). Efficient feature selection and classification for vehicle detection. *IEEE Transactions on Circuits and Systems for Video Technology*, *25*(3), 508–517. https://doi.org/10.1109/TCSVT.2014.2358031

Xue, Y., Jiang, J., Zhao, B., & Ma, T. (2018). A self-adaptive artificial bee colony algorithm based on global best for global optimization. *Soft Computing*, *22*(9), 2935–2952. https://doi.org/10.1007/s00500-017-2547-1