

Coverage Based Hybrid Approach To Solve Random Selection Problems In Test Case Prioritization

Mohit Arora¹, Puneet², Shivali Chopra³

School of Computer Science and Engineering

Lovely Professional University. Phagwara, Punjab, India

¹mohit.15980@lpu.co.in, ²puneetbhardwaj616@gmail.com, ³shivali.19259@lpu.co.in

Abstract

A large number of prioritization techniques have been developed in software testing. These prioritization techniques are based on code coverage, history based, requirement based, cost effective and many other factors. The existing prioritization techniques provides accurate results to assign the priority to the test cases so that the execution of test cases reduce the cost, time and efforts in testing process of software. In the existing techniques when the test cases having same coverage rate or same fault detection rate, then test cases are selected randomly. Only few researchers work on these random selection problems. Most of the existing research is based on fault-based techniques only. This research works on to resolve these tie cases using coverage-based prioritization technique by calculating the complexity of test cases having same coverage rate, assign priority to test cases according to its complexity and choose the higher priority test cases instead of random selection to reduce the cost of testing process.

Keywords: Test case prioritization, Software Testing, Test case generation, Random Selection problems

1. Introduction

Life cycle of software development consists of main four phases requirements gathering, design of software, implementation, testing and the maintenance phase. Like the name describe in first stage requests from the customers and clients are gathered and analyzed, in the second phase of SDLC software designing is performed by designer, in the next phase of SDLC working is performed and last stage is testing and maintenance of software. The process of testing major incorporates the processing of software to finding the errors from the software before it delivers to the clients and management. Testing is one of the major aspects in the life cycle of software development.

In Software Testing Life Cycle (STLC), regression testing is one of the main costly processes that decide whether any new problems are the result of software variation or change. By make change, with large amount of test cases are tested and then successfully deliver. However, change is applied; testing a code with all test cases is too costly and time overriding process. Thus, either whole software should be tested with a little amount of test cases or in certain area with few test cases to identify the error or new problem due to changes in the existing code.

As the efficiency of TCP approaches is evaluate using Average Percentage of Faults Detected (APFD). This evaluates in how much time duration an unordered test cases find errors that is definitely major concern while prioritizing test cases. Contrary, it does not judge how quick the prioritization method itself is. Inside dissimilarity, the duration vacant for verification implementation cycles decreases, mainly to the industries migrating in the direction of quick discharge or unbroken combination practices [10]. A change in software system that might be environmental change or simply generating new edition, they must be retested to make sure its feature. Regression testing mainly focuses to modify the software as well as perform verification on the improved edition of software system [16]. During the process of software testing, diversity is one helpful assess for the equally choosing and ordering of test cases, as well as many similarity-based methods have been proposed. But, these methods do not extent up to large test suites.

When software is modified then the tester uses these earlier defined test cases to make sure that no new defects are occurred in earlier tested software. But to process all of test cases to verify a little change is no beneficial. This is unsuitable to re-execute all the test case for all programs if once variation occurs. Additionally, this causes very costly technique for re-execution of all test cases for a little modification. Thus, to minimize the cost of regression testing technique and to create it more successful the suggestion of the test cases prioritizations was found by many researchers.

The various techniques including, generation, minimization, selection and prioritization of test suite enclose been introduce towards choose test suite for the regression testing, accordingly the amount of test suite are reduced to decrease the complexity of testing process. These methods are used by the software testers in order to minimize the cost associated with regression testing

without any impact on the software effectiveness. The test suite generation methods create the test suite in favor of the software testing. Initially the tester generates huge amount of test suite. The test suite minimization reduces unnecessary test cases. Test cases selection technique in regression testing is used to choose the largely related test suite that is appropriate as well as pass during the customized parts of the process. Test case prioritization technique in regression testing does not reduce the test cases; it is used to find the ordering of the test cases in which the execution of test cases is done and in return gives more advantages to the software testers. The tester gives the priority to each and every test case that ensures that the test cases having high priority are executed earlier in the process of testing.

Test cases prioritization technique (TCP) is one of the major approaches to list the order of tests execution and may be considered for practice also in regression testing or non-regression testing. The prioritization of test case has been largely studied. This wants to managing the order of execution of the test case towards recovering the testing process beneficial. In recent years, prioritization of abstract test cases has also been broadly considered, mainly during the field of testing in software product line as well as combinatorial testing [21]. The major problem of test case prioritization be to recognize which type of test cases might give necessary in order for stoppage recognition and error localization. As far as this, various test case prioritization methods have been discovered [23]. Earlier up-to 1997, people used to generate the test cases randomly and then choose one of the best test cases on the behalf of on those randomly generated without assessing the quality of test cases. Test suites is composed of large variety of test cases (may be in thousands or more). In fact, looking for identical things is a basic difficulty in mining of data, along with extremely capable algorithms exists to deal with his problem. While, check of a software edition following required changes reveal whether modules perform fit, and prove has been termed as the regression test. This test is considered as search the existence of practical problems and regression tests can also be there executed in an exact duration or designed into all edition of software system [42]. In regression testing, test case prioritization is generally practical. Since modifications are defined into software, prior saved test suite wants to be re-executed in the direction of ensure quality and constancy.

2. Background

Xiaolin Wang et al. [1] offered the test case prioritization called prioritization of requirement for test. Requirements prioritization is defined by the users and developers. This paper not examined the severity of faults; for that reason, it will be a research focus to improve the test effectiveness in the prospect. Hongwei Zeng et al. [2] in their paper present test case prioritization based on history. The requirement priorities of the test sets are used to initialize priorities of the test cases and after that are determined at run time on the basis of the past information in the regression testing. The results prove an enhanced work in favor of this technique through by utilizing the calculations of Average Percentage of Faults Detected and Fault Detection Rate. Wenhao Fu et al. [3] in their paper, they proposed a test case prioritization algorithm which is based on program changes and method invocation relationship. They performed experiments on 11 programs. The results showed that the technique is forever better to, or at slightest as successful as, several of the well-established prioritization approaches in the literature, counting in addition to two greedy algorithms based on method coverage and statement respectively. They will extra find how to improve the performance of regression testing by combining the fault severity, historical information, time etc. in future. Faiza Farooq et al. [4] in their paper suggest variety of white box prioritization approaches. In short, it concludes that the major drawback of this technique is the test case with maximum number of faults was given lower priority in branch coverage. Remo Lachmann et al. [5] in their paper describe a test case prioritization technique without the code access within the testing at system level. In favor of black box testing it uses high stage artefacts which are requirements, test cases. Their method is capable to identify faults previous and, therefore, allow for more improved regression testing. Jiany Zhou et al. [6] in their paper describe the study of comparing the efficiency of various approaches based on static and dynamic coverage, by using 15 Java programs. Erum Ashraf et al. [7] in their paper present a hybrid approach of artificial intelligence with the concept of value based to solve the prioritization problem in regression testing. Fatemeh MosalaNejad et al. [8] in their paper designed four memetic algorithms based on Genetic Algorithm. The improved results derived using local search method that can judge both the best case and worst-case solutions. Yiting Wang et al. [9] in their paper mainly refer to additional statement-based test case prioritization. This technique sorted the prioritization of test cases based on severity of the faults when the code coverage rates of the program entities are equal. Breno Miranda et al. [10] introduced some test

case prioritization techniques under FAST family that commonly used in the field of big data to find the homogeneous items. FAST approach has the capability to prioritize large number of test cases in fewer than 20 minutes. Sonia Sharma et al. [11] present a review on test case prioritization using Ant Colony Optimization. ACO is an effective and optimistic technique to solve the selection and prioritization problems of the test cases. Dima Suleiman et al. [12] present a survey on prioritization of test cases in regression testing. From the literature they classified prioritization techniques into three main classes: requirement, coverage and model based. Dongdong Gao et al. [13] prioritizes test cases in regression testing based on ACO (Ant Colony Optimization). Three factors to be considered for prioritizations are fault severity, fault detection and execution time. Kamma Solanki et al. [14] present a technique which is based on modified Ant Colony Optimization (m-ACO). The proposed approach increases the range of faults enclosed. Andreea Vescan et al. [15] provide formal approach based on requirement dependencies for prioritization of test case in regression testing. The kind of method used is system level which considers functional requirements and dependencies between the requirements. Xiao-Yi Zhang et al. [17] describe random and coverage-based technique for prioritization of fault localization. Within future effort, to deal with FLP problems extra adaptive error localization methods have been presented, emphasizing the put in the ground association among the prioritization of test sets as well as the error localization methods. Mohammad Rava et al. [18] review the test case prioritization techniques. They show the research development of test case prioritization process with time structure. Satendra Kumar et al. [19] describe prioritization of test cases for software product line. Software product line testing is tougher and too costly than single system testing. Andrey Morozov et al. [20] present test suit prioritization of model based automated software. The technique depends on two principles: (i) Triggering of bug by the test case in the updated block and (ii) this triggered bug must shift to the place where it is more likely to be found. Rubing Huang et al. [21] in their paper present an assessment on abstract test case prioritization. This paper analysed promising ATP techniques, according to the four categories: non-information-guided prioritization (NIGP); input-model mutation-based prioritization (IMBP); interaction coverage-based prioritization (ICBP) and similarity-based prioritization (SBP). Zainab Sultan et al. [22] describe an analytical review on test case prioritizations which compare the prioritization techniques with random methods. Jian Ding et al.

[23] in their paper discover the relationship between DRT (Dynamic Random Testing) and ART (Adaptive random Testing). Besides, they will also build utilization of its individual compensation to suggest original approaches of adaptive testing. Sulaxna Solanki et al. [24] suggested a new hybrid technique for regression test case prioritization to enhance the fault detection rate. This algorithm uses statement, critical and branch coverage criteria for prioritizing the test case. Johanna Ahmed et al. [25] reviewed various test case prioritization techniques for sequence of events. The review details on different TCP used to show the search for coverage criteria, size and type of the test cases, techniques in assessment and validity threats of earlier mechanism. David Paterson et al. [26] in their paper present coverage-based test case prioritization. In his work, the author had discussed the results of experiments using four test prioritization techniques. Omdev Dahiya et al. [27] reviewed various studies related to the regression test case prioritization approaches. This research wants to find the existed research of various researchers in just beginning a variety of prioritization of test cases techniques utilized into the regression testing area. Qi Luo et al. [28] describe a major contrast of static and dynamic techniques of prioritization of test cases. In this effort, they make the first wide study designed at empirically evaluating four static test case prioritization approaches matching them within condition of study. Kevin Moran et al. [29] in their paper present test case prioritization on real faults and mutants. In regression testing, the prioritization of test cases is considered as a key component, allowing for prior fault detection or help to decrease testing cost and time. Abid Robeala et al. [30] suggest a new some criterion on the basis of prioritization of test sets methods which takes two criterions in favor of order test sets by utilizing extra scheme. Maral Azizi et al. [31] present a structure of graph based with the aim of plotting the order assignment towards an algorithm of graph traversal. Jianlei Chi et al. [32] describe a method call sequences on the basis of order of test set measure. In favor of stability the prioritization efficiency and effectiveness as established exposure basis order of test sets approach frequently face the difficulty. Jeff Offutt et al. [33] describe the model of mutant stubbornness, which is the origin for their study approach. The results indicate that mutant inflexibility method construct test cases which be equally smallest within amount as well as ordered well, while fine or else enhanced as compared to earlier approaches. Dipesh Pradhan et al. [34] present a method in favor of black box dynamic ordering of test sets with multi objective search and rule mining. Paruchuri Ramya et al. [35]

establish that during the stage of verification incorporating needs has useful many in favor of determining the bugs as well as errors basically. Ovidiu Baniyas et al. [36] they learn the weakness of the methods in prioritization of test sets on the basis of instruction exposure as well as and helpful in favor of field test sets. Manoj Kumar Sahu et al. [37] proposed an approach using business criticality value in regression testing to create prioritized test case suite. Hasan Mahmood et al. [38] will expand the earlier work to contain actual existence realistic feature in favor of map test sets. Based on the useful aspects that will envelop up on the whole of the production functionally. Bo Jiang et al. [39] describe a part orient randomized algorithm (PORA) for the ordering of test cases. Through optimize space among the ordering of a large number of test cases PORA guides test case prioritization as well as chain of command of allotment of test contribution information. Surendra Mahajan et al. [40] describe the ordering of test sets on the basis of element arrangement that process in favor of find out extra strong bugs on an early on phase and in the direction of get enhanced system invention deliverable performance utilizing java decoding method using greedy algorithm. Muhammed Maruf et. al. [42] in their paper describes an algorithm named as bat inspired for prioritization of the test cases with the help of limited and overall search attributes of a bat algorithm.

3. Proposed Approach

Inside regression testing of software, test cases prioritization is the most important technique to prioritize test cases to improve the test efficiency. When test cases have equal code coverage area, then the randomly selection of test cases will influence the effect of sorting. There are very few techniques founded by the researchers to handle tie cases. In some cases, like for small size programs, using more time and cost these approaches works well, but for large programs and for short time duration these random cases are not handled by the existing approaches. And also, there are very few techniques which handle those tie cases. Many existing techniques don't handle tie cases. Due to small research founded in handling tie cases, a new problem is taken into account and solves random selection problems in code coverage using hybrid algorithm of test case prioritization.

From the above viewed descriptions, the research objectives of work are:

- a. Implementation of Hybrid coverage-based approach for solving random selection problems during test cases prioritization.

- b. Detection of more faults in small duration for large number of test cases.
- c. Prioritize test cases using BAT inspired algorithm.

4. Research Methodology

4.1 Test case Generation

Test cases can be generated by taking apache server open source code from SIR repository and then using plug in Eclemma, statement coverage of each test case can be analyzed and denoted as “G” in the dataset and not covered statement is denoted as “R”. Total of 100 test cases with their coverage can be taken as a dataset.

4.2 Calculate the statement coverage priority of test cases

For this Adaptive approach is used. In this, until our statement not covers, we sort the test case if test cases missing or we can state breakdown test cases those are not capable to cover up any statement it way the statement coverage is not complete entirely. We acquire that missing test cases behind applying forceful technique and achieve BAT algorithm on these test case. In BAT algorithm we relate three main approaches to instruct the test case similar to this our APSC enhanced as compare to Adaptive technique.

Adaptive Algorithm Process: The adaptive technique calculates the early fault detection ability (denoted as priority (t)) of every test set t on the basis of statement coverage of the test sets lying on the earlier program and select some test case ‘ts’ among the major priority. After that implementation and test case prioritization occurs concurrently. Here particular, programmers implement the preferred test case ‘ts’, taking either its productivity is accepted otherwise rejected. After that base on its productivity, the adaptive technique alters the priority of unclosed test sets as well as chooses the test sets by the main customized priority. The adaptive technique repeats the earlier procedure until every test case inside T are chosen and prioritized.

Prioritization approach of adaptive test case has two uniqueness. Initial, its ordering of test sets improvement as well as implementation procedure perform at the same time, diverse from the offered prioritization of test cases techniques. Next, the adaptive method is computing the error finding ability of test sets as rapidly as programmers close executing a test case.

4.3 Calculate the priority of test cases having same coverage

The test sets having the similar statement coverage rate, we find the priority of each test set to differentiate them and according to precedence of test cases we assign priority to every test case and based upon the test cases priority we choose the test cases as compared to random selection. The test cases having high priority complexity, acquires higher priority. The precedence of test sets is analyzed and is dependent on the sequence of new test cases; the scheduling of test can be done if in case priority is also same.

4.4 Initialize the test cases

After finding the priority of each test case, initialize the test set for BAT algorithm with this prioritize test case population. Instead of assigning the test cases set as an initial population randomly, adaptive prioritize population of test cases is given as an initial population.

4.5 Scheduling of test cases using BAT algorithm

- Beside to address the random test set issue, an improvised approach has been proposed.
- This approach attempts to overcome these limitations by combining BAT-inspired algorithm and adaptive approach.
- To begin with, the approach employs adaptive technique for prioritizing individual's test cases that include attained statement coverage and faults detection on the earlier program.
- These prioritize test cases are fed as an initial population of BAT inspired algorithm.
- Inside this method, the hybrid method succeeds within achieving nearly perfect statement coverage and fault detection in minimum time.

5. Pseudo Code

An adaptive approach presents an effective method for obtaining solutions to prioritization problems and is thus, gaining popularity lately. Unlike prevailing test set methods techniques that carry out precedence of test sets prior to their execution, an adaptive approach performs these two processes side by side. Initially, it calculates an early fault-finding ability for every test

case t , based on the test case's previously achieved statement coverage. Then it picks out the test set having high priority, which is indicated by the equation given below:

$$Priority(t) = \sum_{s \text{ is executed by } t} Potential(s) \dots \dots \dots (1)$$

where $Potential(s)$ refers to the likelihood of a statement s containing faults that are missing undetected with an active test suite. For each statement, it must lie in interval $[0, 1]$. In the beginning, every statement is assigning a $Potential$ 1. Further, the adaptive approach executes the test set having the high precedence as well as and modifies the $Potential$ of every statement s in agreement with its output using equation 1. The following statement is applied for this purpose:

$$Potential(s) = \begin{cases} Potential'(s), & \text{if } s \text{ is executed by } t' \\ Potential'(s) * q, & \text{if } s \text{ is executed by } t' \wedge t' \text{ is passed} \\ Potential'(s) * p, & \text{if } s \text{ is executed by } t' \wedge t' \text{ is failed} \end{cases}$$

Here, $Potential'(s)$ represents the probability of some statement contain original faults before executing any test case t' . Let, p and q are two non-negative constants, for which values lie somewhere between 0 and 1. This procedure continues till every of the test cases contain prioritized and executed.

Input: Test suite T

Output: Prioritized test suite T'

Declaration: t_s represents the newest particular test case

Begin

For every test case t in T .

Compute first $priority(t)$.

end for.

Choose the test case (t_s) by the main priority in T .

Add t_s to T'

$T \leftarrow T - \{t_s\}$.

Run t_s .

As T is not empty **do.**

For every test case t in T .

Alter $\text{priority}(t)$ based on the output of t_s .

End for

Choose the test case(t_s) with main priority which cover statement.

Add t_s to T' .

$T \leftarrow T - \{t_s\}$.

Run t_s .

end as

return T'

Initial population of BAT algorithm is generated using Adaptive approach then BAT algorithm prioritizes the test suite.

Establish paths of bats as implementation duration of test set $t_1, \dots, t_n$

f_i the frequency of associated t_i s in inverse relation $f_1, \dots, f_n$

Initialize bat population with x_i, v_i

Assign r_i and A_i by looking the frequencies and faultiness (rA)

$\text{minMax} = f\text{MinMax}(f)$

$rA_{\text{New}} = x\text{Star}(rA);$

ast<number of iteration **do**

for $j=0$ to $\text{length}(f_n)$ **do**

$f_frequency[k] = f_frequency[\text{minMax}[0].I_tem1] + (f_frequency[\text{minMax}[0].I_tem2] - f$

$frequency[\text{MinMax}[0].I_tem1]) *$

$generateBeta()$

$v_i = v_i + (pop[j].I_tem1 - rA_{\text{New}}[0].I_tem1)$

$R = \text{rand}()$

if $R \geq r_j$ **then**

$rA_{\text{New}} = x\text{Star}(rA_{\text{New}});$

else

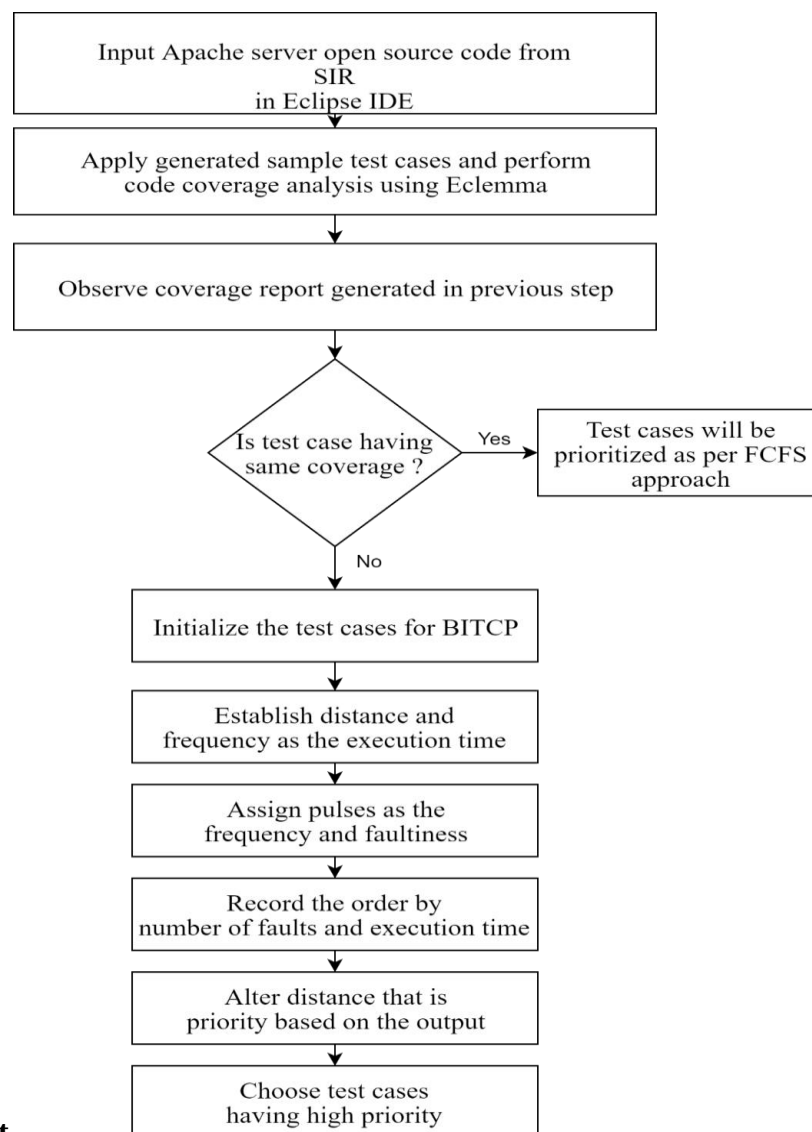
increase r_j reduce a_j

end if

end for

Return the record with bats ranked by the volume of errors and test set implementation duration

end as



6. Flowchart

Fig 1: Flow chart for Research Methodology

7. Conclusion and Future Scope

It can be deduced from the proposed methodology, APFD can be significantly improved and this taking research into advance frontier area. A hybrid rank based prioritization approach has been proposed.

We have also implemented the proposed work and will be published in the sequels. As a future scope, one may extend the work using optimization technique like Satin Bower Bird, Sail fish optimizer etc.

REFERENCES

- [1] T. Ma, H. Zeng, and X. Wang, "Test Case Prioritization based on Requirement Correlations," IEEE, no. 61170044, pp. 0–5, 2016.
- [2] X. Wang and H. Zeng, "History-Based Dynamic Test Case Prioritization for Requirement Properties in Regression Testing," 2016 Int. Work. Contin. Softw. Evol. Deliv. Hist., pp. 41–47, 2016.
- [3] W. Fu, H. Yu, G. Fan, X. Ji, and X. Pei, "A Regression Test Case Prioritization Algorithm Based on Program Changes and Method Invocation Relationship," Asia-Pacific Softw. Eng. Conf., pp. 169–178, 2017.
- [4] F. Farooq and A. Nadeem, "A Fault Based Approach to Test Case Prioritization," 2017 Int. Conf. Front. Inf. Technol., pp. 52–57, 2017.
- [5] R. Lachmann, M. Nieke, C. Seidl, I. Schaefer, and S. Schulze, "System-Level Test Case Prioritization Using Machine Learning," IEEE Int. Conf. Mach. Learn. Appl., 2016.
- [6] J. Zhou and D. Hao, "Impact of Static and Dynamic Coverage on Test-Case Prioritization: An Empirical Study," 10th IEEE Int. Conf. Softw. Testing, Verif. Valid. Work. Poster, 2017.

- [7] E. Ashraf, S. Ahmed, T. ahmed Khan, and K. Mahmood, "Value based PSO Test Case Prioritization Algorithm," *Int. J. Adv. Comput. Sci. Appl.*, vol. 8, no. 1, pp. 389–394, 2017.
- [8] F. M. Nejad, R. Akbari, and M. mehdidejamShahabi, "Using Memetic Algorithms for Test Case Prioritization in Model Based Software Testing," *1st Conf. Swarm Intell. Evol. Comput. (CSIEC2016)*, High. Educ. Complex Bam, Iran, 2016, pp. 142–147, 2016.
- [9] Y. Wang, X. Zhao, and X. Ding, "An Effective Test Case Prioritization Method Based on Fault Severity," *IEEE*, 2015.
- [10] B. Miranda, E. Cruciani, R. Verdecchia, and A. Bertolino, "FAST Approaches to Scalable Similarity-based Test Case Prioritization," *ICSE*, 2018.
- [11] S. Sharma, P. Vrqlkdulox, P. L. Jpdlo, F. R. P. Pdlo, D. Vdlql, and J. Frp, "Model-Based Test Case Prioritization Using ACO : A review," *Fourth Int. Conf. Parallel, Distrib. Grid Comput.*, 2016.
- [12] D. Suleiman, M. Alian, and A. Hudaib, "A Survey on Prioritization Regression Testing Test," *8th Int. Conf. Inf. Technol.*, pp. 854–862, 2017.
- [13] D. Gao, X. Guo, and L. Zhao, "Test Case Prioritization for Regression Testing Based on Ant Colony Optimization," *IEEE*, no. 91118007, 2015.
- [14] K. Solanki, Y. Singh, and S. Dalal, "Test Case Prioritization : An Approach Based on Modified Ant Colony Optimization (m-ACO)," *IEEE Int. Conf. Comput. Commun. Control*, 2015.
- [15] A. Vescan and L. Dios, "Requirement Dependencies based Formal Approach for Test Case Prioritization in Regression Testing," *IEEE*, pp. 181–188, 2017.
- [16] N. kumarSaklani and P. , Singh, "Review of Prioritization Techniques in Regression Testing," *Int. J. Comput. Sci. Mob. Comput.*, vol. 6, no. 4, pp. 216–221, 2017.
- [17] X. Zhang, D. Towey, T. Y. Chen, Z. Zheng, and K. Cai, "A Random and Coverage-based Approach for Fault Localization Prioritization," *IEEE*, pp. 3354–3361, 2016.

- [18] M. Rava and W. M. N. Wan-kadir, "A Review on Prioritization Techniques in Regression Testing," *Int. J. Softw. Eng. Is Appl.*, vol. 10, no. 1, pp. 221–232, 2016.
- [19] S. Kumar and Rajkumar, "Test case prioritization techniques for software product line: A survey," 2016 *Int. Conf. Comput. Commun. Autom.*, pp. 884–889, 2016.
- [20] A. Morozov, K. Ding, T. Chen, and K. Janschek, "Test Suite Prioritization for Efficient Regression Testing of Model-based Automotive Software," *Int. Conf. Softw. Anal. Test. Evol.*, no. 1, 2017.
- [21] R. Huang, W. Zong, D. Towey, Y. Zhou, and J. Chen, "An Empirical Examination of Abstract Test Case Prioritization Techniques," *IEEE/ACM 39th IEEE Int. Conf. Softw. Eng. Companion*, 2017.
- [22] Z. Sultan, S. nazir Bhatti, R. Abbas, and S. asifali Shah, "Analytical Review on Test Cases Prioritization Techniques : An Empirical Study," *Int. J. Adv. Comput. Sci. Appl.*, vol. 8, no. 2, pp. 293–302, 2017.
- [23] J. Ding and X. Zhang, "Comparison analysis of two test case prioritization approaches with the core idea of adaptive," *IEEE*, pp. 1723–1729, 2017.
- [24] S. Solanki and A. singh Yadav, "A New Approach For Regression Test Case Prioritization Using Branch Coverage , Decision Coverage And Criticality Coverage Techniques," *Int. J. Recent Innov. Trends Comput. Commun.*, vol. 5, no. April, pp. 109–114, 2017.
- [25] J. Ahmad and S. Baharom, "A systematic literature review of the test case prioritization technique for sequence of events," *Int. J. Appl. Eng. Res.*, vol. 12, no. 7, pp. 1389–1395, 2017.
- [26] D. Paterson, G. M. Kapfhammer, G. Fraser, and P. McMinn, "Using Controlled Numbers of Real Faults and Mutants to Empirically Evaluate Coverage-Based Test Case Prioritization," 2018 *IEEE/ACM 13th Int. Work. Autom. Softw. Test*, pp. 57–63, 2018.
- [27] O. Dahiya and K. Solanki, "A systematic literature study of regression test case prioritization approaches," *Int. J. Eng. Technol.*, no. September, 2018.

- [28] Q. Luo, K. Moran, and D. Poshyvanyk, "A Large-Scale Empirical Comparison of Static and Dynamic Test Case Prioritization Techniques," 2018.
- [29] Q. Luo, K. Moran, D. Poshyvanyk, and M. Di Penta, "Assessing Test Case Prioritization on Real Faults and Mutants," 2018.
- [30] R. Abid and A. Nadeem, "A Novel Approach to Multiple Criteria Based Test," 2017.
- [31] M. Azizi and H. Do, "Graphite : A Greedy Graph-Based Technique for Regression Test Case Prioritization," 2018 IEEE Int. Symp. Softw. Reliab. Eng. Work., pp. 245–251, 2018.
- [32] J. Chi, Y. Qu, Q. Zheng, W. Jin, D. Cui, and T. Liu, "Test Case Prioritization Based on Method Call Sequences," 42nd IEEE Int. Conf. Comput. Softw. Appl., 2018.
- [33] L. Gonzalez-hernandez, B. Lindstrom, J. Offutt, P. Potena, and M. Bohlin, "Using Mutant Stubbornness to Create Minimal and Prioritized Test Sets," 2018 IEEE Int. Conf. Softw. Qual. Reliab. Secur., pp. 446–457, 2018.
- [34] D. Pradhan, S. Wang, S. Ali, T. Yue, and M. Liaaen, "REMAP : Using Rule Mining and Multi-Objective Search for Dynamic Test Case Prioritization," IEEE 11th Int. Conf. Softw. Testing, Verif. Valid., pp. 46–57, 2018.
- [35] P. Ramya, V. Sindhura, and P. V. Sagar, "Clustering based prioritization technique," 2018 Second Int. Conf. Inven. Commun. Comput. Technol., no. Iccict, pp. 1181–1185, 2018.
- [36] O. Baniyas, "The drawbacks of statement code coverage test case prioritization related to domain testing," 11th IEEE Int. Symp. Appl. Comput. Intell. Informatics, pp. 221–224, 2016.
- [37] M. K. Sahu and A. Natha, "Test case prioritization technique for regression testing," Int. J. Adv. Res. Comput. Eng. Technol., vol. 3, no. 8, pp. 2730–2740, 2014.
- [38] H. Mahmood and S. Hosain, "Improving Test Case Prioritization Based on Practical Priority Factors," IEEE, pp. 7–10, 2017.
- [39] B. Jiang, W. K. Chan, and T. H. Tse, "PORA : Proportion-Oriented Randomized Algorithm for Test Case Prioritization," 2015 IEEE Int. Conf. Softw. Qual. Reliab. Secur., no. 61202077, 2015.

- [40] S. Mahajan, S. D. Joshi, and V. Khanaa, "Component-Based Software System Test Case Prioritization with Genetic Algorithm Decoding Technique Using Java Platform," 2015 Int. Conf. Comput. Commun. Control Autom., 2015.
- [41] E. Khandelwal and M. Bhadauria, "Various Techniques Used For Prioritization of Test Cases," Int. J. Sci. Res. Publ., vol. 3, no. 6, pp. 3–6, 2013.
- [42] M. M. Öztürk, "A bat-inspired algorithm for prioritizing test cases," Vietnam J. Comput. Sci., vol. 5, no. 1, pp. 45–57, 2018.