2020 International Conference on Intelligent Engineering and Management (ICIEM)

A comprehensive review for test case prioritization in Software Engineering

Ashima
School of Computer Science & Engineering.
Lovely Professional University
Phagwara, 144411 ,Punjab
ashigarg57452@gmail.com

Golmei Shaheamlung
School of Computer Science & Engineering.
Lovely Professional University
Phagwara, 144411 ,Punjab
shaheamlung@gmail.com

Ketusezo Rote
School of Computer Science & Engineering.
Lovely Professional University
Phagwara, 144411 ,Punjab
ketusezo.rote@gmail.com

***Abstract*— In the past years, test case prioritization has been improved in the regression testing by the use of effective test cases. The continuous improvement and attention have increased in terms of prioritization algorithm, coverage criteria, measurement, application scenario and the practice concerned. It has focused mainly on Prioritizing and scheduling the test cases. The main purpose of this paper is to study the different prioritization techniques used by various authors in previous years. Regression testing, a type of testing in which is being used as a tool for checking, testing, and up-gradation of software. The test cases of all scheduling and prioritizing are set in the ordered and proper method and as result, this case shows the detection and a maximum number of faults in the software in which the technical faults are traced and detected as the detected faults. Through the fault detection of the test case, it reduces the test case and minimizes the execution cost. As a result, this method shows the running of the test case at a higher priority in order to reduce and minimize the cost, time and effort of software testing.**

***Keywords*— Regression Testing, Test case prioritization, Prioritization algorithm**

I INTRODUCTION

The usage of Software Testing is basically to reach and find out the errors. It is the most important phase in the SDLC. In this we match the actual outcomes with the expected outcomes to ensure that our software is defect free or not. Any kinds of test suites designed by the software developers are saved by them to be reutilized in future as per the evolution of software. In the software industry, some test suite reutilization is pervasive in the form of regression testing. The cost of the software maintenance is accounted together with other regression testing activities is One-half of its demand. An inordinate amount of time can be consumed by running all test cases in an existing test suite. For attempting to maximize some of the objective functions given, regressions testing for the test case are scheduled by prioritization techniques of test case. For instance, the fastest rate possible is the code coverage could be achieved by the testers that wish to program and arrange the schedule test cases. As per the expected frequency of its use, the features can also be exercised in a manner in which it reflects the propensity of its historical has failed, the subsystems are exercised. When the execution of time requirement for all test cases in a test suite is less the test case then the prioritization is not cost-effective. For scheduling the test cases in any order, this method might be the most expedient. The test case prioritization can be beneficial in time of needs to run and execute of all test cases in which the test suite period is long.

TYPES OF TESTING

White Box Testing and Black Box Testing are the two major terms that are widely used in organization. White box Testing approach handle the structure and function inside the code. White Box testing also known as testing for glass, open box, structure testing or clear-box testing. To carry out the testing, the tester has to experiment and deal with the code and therefore it is required to have the core knowledge of programing, scripting ,coding and logical of internal code work. Through the White box

the tester examine and experiment the program asses which code unit/statement/chunk is defected.

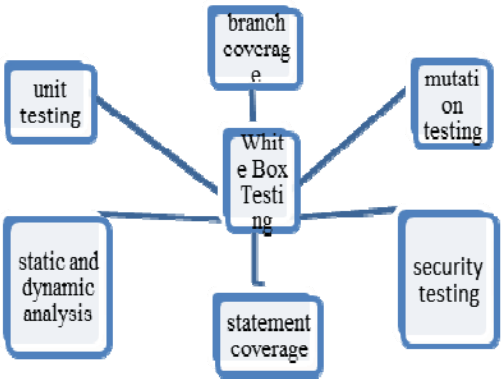

Figure 1.1. White Box Testing

Unit Testing: It is defined as category of testing where a single module of the software is tested. Unit testing is done by developer. During the progress of the software product, the unit testing of the particular product is executed.

The analysis of Static and Dynamic: The static analysis includes all the code that can be feasible and detect the defects in the code. It includes carry out the analyzing and the code of the result.

Statement coverage: Execution of the testing code, in which this type of testing in the way that each application statement in the code is executed just once. It also assures that all the executable statements are executed unaccompanied by any issue.

978-1-7281-4097-1/20/$31.00 ©2020 IEEE

To achieve a specific functionality, we have to branch out the code. All the branches in the code are validated with the help of branch coverage testing and make assured that no branching shows the software abnormal behavior.

Security Testing is used to determine on how good the device can secure itself against unauthorized activity such as, hacking, spoofing, cracking and any other unauthorized access etc. in which, this handles the application code. Sophisticated Testing techniques are required for this security testing.

A kind of testing in which, after fixing a specific bug or defect, the program is checked for the software that was changed. It helps in sorting and finding out which coding technique strategy can help out to successfully improve the software.

Black Box Testing, this is another type of software testing process used to test the application without understanding the Code and program internal structure. The black box's main purpose is to test whether the software works as specified in the specification report and whether or not it meets the user's expectations.

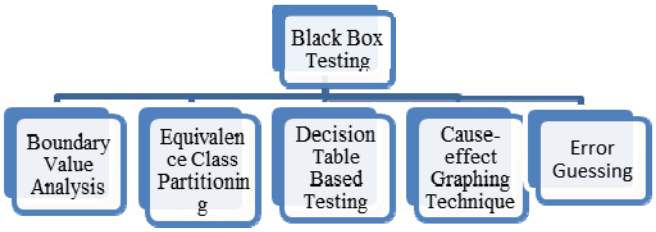

Figure 1.2. Black Box Testing

This type of testing tracks the applications behavior at the boundary level. Boundary value analysis is conducted to determine whether boundary values defects occur. Boundary Value Analysis is used to check a various number set. There is an upper boundary and lower boundary for each set of number and these boundary values are checked.

It is one of type of the Black- Box testing. A group set is selected during the Equivalence Partitioning and a few values or numbers are taken for testing. All the values of that group generate the similar outcomes. The aim of the testing is to remove unnecessary test Cases within the specific group that generates the same output without any defect.

A Decision Table Testing is a perfect way to handle with multiple inputs combination which gives various outcomes. It is known as the Cause-Effect table. It gives a systematic way to stating complex rules of business that are useful for developers for testers.

Cause Effect Graph is a technique of black box testing that graphically explains the relationship between a given result and all the factors influencing the result. It is also called as Ishikawa Diagram and it was invented by kaoruishikawa and because of its appearance it is called fish -bone diagram.

Error Guessing is a technique of software testing that guesses the fault which can persuade under the code. It is an Experience-Based testing method in which test analyst used the experience to learn more about the areas of the problems of the product.

TEST CASE OPTIMIZATION

The process of making the testing process faster while its accuracy is not compromised is called optimization of testing process. The construction will be completed at a reasonable faster pace than the traditional method. The process of test case optimization can be done by doing changes the way the test case are executed that covers the changes in the construction or execute the test in an optimal order. It is understood that testing the product or application and fixing bugs when the software is already on the market is a very expensive process, so to reduce the costs, for increase the productivity of the testing and utilization of hours, techniques for optimization are used.

Test case optimization goals are as follows:

1. Increase the fault detection rate and correction
2. Regression tests only those areas which have been modified
3. Reduce the time required to complete the regression test suites

To achieve the objectives of test case optimization, the proposed system has been designed with two phases of work namely code based Testing (CBT) and model-based testing (MBT). Testing is performed at program code-level is called Code-Based Testing (CBT). Model based testing is a method of software testing in which software execution time behavior is tested against model's predictions. A model is a described the behavior of the system. Behavior can be interpreted with regard to input sequences, behavior , circumstances ,output and information flow from input to output.

Test data is the software program's input. It represents the information affecting or is influenced by the execution of the particular module. Few information can be used for Positive Testing, to check that the expected results produced by the

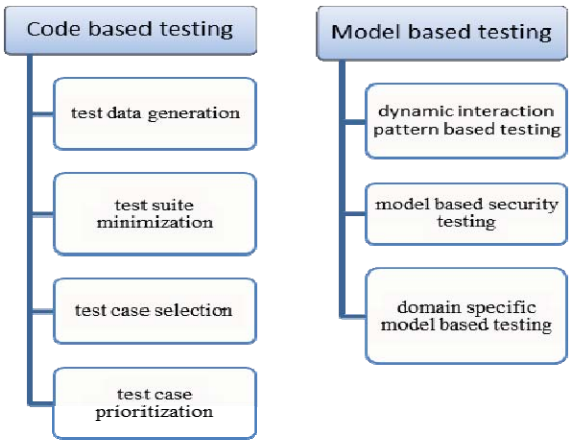

Figure 1.3. Phases of Testing

Test data is the software program's input. It represents the information affecting or is influenced by the execution of the particular module. Few information can be used for Positive Testing, to check that the expected results produced by the given input sets to a given function . for Negative Testing another information may be used to check the ability of the program to handle unusual, extreme, exceptional, or input that is unexpected.

Test data-generation: the creation of the test case includes many pre steps or configurations of the test environment that are very time consuming in many testing environment.

Test data can be generated:

- By Manual operation
- By making mass copy of production data to environment of the testing
- By making test data replication from existing customer Systems
- By automated producing test data methods.

II LITERATURE REVIEW

- Zainab Sultan et al [1], In this paper the author have analysed that through regression testing the software quality and fault detection can be enhanced. As every technique having its own pros and cons the author states that the tester may use any test case prioritization technique according to their prerequisite.
- Wenhong Liu et al [2], In this paper the author has presented dynamic adjustment approach, and later on the execution order of the following test cases are optimized. This paper suggests that execution of the test case accordingly along with the test case prioritization which it can increase and improve the efficiency. The result of this paper is that it can significantly increase and improve the rate status of finding defects.
- David Paterson et al [3], In this paper presents the results of empirical study on the usage of defect prioritization. The paper shows promising results as by using the defect prediction for TCP the number of required test case to find a fault is less on an average of 9.48% as compared to some existing coverage based technique and 10.4 as compared to history based technique.
- Dan Hao et al [4], in this paper the author presents the aspects that are essential to increase test effectiveness. The main five aspect such as prioritization algorithms, practical concerns involved, measurement, ,coverage criteria and application scenarios. The author has reviewed the achievements regarding test case prioritization of the five aspect cited above and has also given the perspective on the challenges.
- **Chengyu Lu et al [5**], In this paper the author aims to increase the code coverage as fast as possible with the help of ant colony framework. Experimental results considering a real world wide application has proven to outperform the existing approaches with respect to search efficiency and solution quality.
- Riza Dhiman et al [6], In this paper the author represents automated slicing and manual slicing of the detection as many number of detected faults from the newly modified research on the latest version. The results has shown improved FD rate and reduce the execution time due to the test case of automated prioritization.
- Pablo Carballo et al. [8] ,paper is based on the technique of Biased Random- Key Genetic Algorithm(BRKGA).The result shows that BRKGA outperforms two different per-mutational based encoding based GA.
- Maral Azizi et al. [9], based on the technique of the Novel graph-based framework. It improves the effectiveness and efficiency. The performance is good as compare to other technique.
- Dario Di Nucci et al [10], this paper is based on this technique (Hypervolume-based Genetic Algorithm. The results show that HGA is more cost effective. With the help of HGA algorithm effectiveness is improved.
- Masataka Ozawa, et al. [11], based on this technique (Markov reward model). Chebyshev-distance metric give better results based on the reward parameters as compare to the other metrics. TCP method up to 63.08%.
- Hasan Mahmood et al [14], this paper based on this technique (Using practical priority factors we improve prioritization of test case). In Results this paper shows that it safe cost and time of the software.
- GayamRupa Sri, et al, based on this technique (Hierarchical Clustering is used). The result shows that the experiment of grouping can be recognized as viable experiments with high rate review of proportion and the rate of significant is exact rate.

Table 2.1 Comparison of existing test prioritization in software engineering by different authors

Authors	Year	Techniques	ML used?	Result	Future Scope
Mahdi Noorian, et al. [23]	2010	Classification Framework.	Yes	To investigate and extract the important information from the previous research works in ML and ST	In future research is reviewing and classifying all current work in the area of ML and ST. Classification framework is strong to catch the various feature of work in ML and ST.
Farn Wang, et al. [21]	2011	Using Neural Networks and Program Slicing Techniques	Yes	The experiments show that our approach perform better on test case prioritization and with the help of this approach we can save the execution time.	In future with the help of previous SUT we can explore it more.
Benjamin Busjaeger et al. [22]	2016	Novel approach for multiple previous techniques	Yes	The results show that this approach perform better as compare to the previous techniques on a large dataset. Our approach is planned to be well-suited for the industrial settings.	In future we evaluate new features, such as superior support for the external artefacts.
Ramzi A. Haraty, et al. [24]	2016	Code and Clustering Change and usage of Relevance.	Yes	Results shows that it execute the appropriate test cases. The prioritization of the test cases are in a way the highly prioritized, the test cases are prior and executed than after another test cases.	In future to expand the results validation on large systems. It also shows the explore of the performance using its same approach.
HasanMahmood et al. [14]	2017	Using practical priority factors we improve prioritization of test case.	No	In Results this paper shows that it safe cost and time of the software.	In future we can use more practical factors that will help to improve the method and we find optimal solution.
Sujata, et al. [16]	2017	Classification of TCP techniques	No	Best approach for classification is fuzzy logic which provide the accuracy up to 93.333%.	The proposed work is the addition of previous work that is already proposed selection schema, more work could be done by getting complex data set.
Jian Ding, et al. [17]	2017	Comparison of ART and DRT	No	Based on the various environments, It supports the selection of the testing approaches and its mechanism of generating test cases.	For more in future we can study in detail, like different types of Patterns Failure.
Rubing Huang`, et al. [12]	2017	Empirically examines possible ATCP techniques	No	1.ICBP give more effectiveness as compare to other techniques. 2.Using ICBP techniques we find the higher strength of FICBP techniques. 3.Using IMBP techniques we find the higher TIMBP and AIMBP techniques.	In future do large scales empirical study using all type of experimental data and metrics to analyse testing effectiveness and efficiency.
HelgeSpieker, et al. [18]	2017	Reinforcement Learning for Automatic TCP	Yes	RETECS algorithm is used to extract the data from the industrial studies, we apply this first time reinforcement learning which give them better results in CI. The results show fast learning of Retecs in industrial cases.	In future we can add the large networks and also use the deep learning techniques.
Shuai Wang1,et al. [13]	2018	Novel rule mining and multi-objective search	No	The results shows that the REMAP perform better as compare to other approaches up to 96% of the studies It achieve 18% higher APFD.	In future we use REMAP more test case TCP(e.g., linear programming based)
Sebastian Ulewicz [28]	2018	Automated production system	No	Show the support during the process of regression testing of using APs and industrial relevance.	The runtime overhead could be decreased even further. Combination of the other tracing techniques increases the applicability of APSs.

Rubing Huang, et al,	2018	Fixed strength interaction coverage based prioritization	No	It achieve the higher Rate of Fault Detection	In future we can find out the relationship between the generation and prioritization strength using FICBP.
Masataka Ozawa, et al,	2018	Markov reward model	No	Chebyshev-distance metric give better results based on the reward parameters as compare to the other metrics. TCP method up to 63.08%	In future we can use this algorithm with the large size of programs.
RayapureddyKalyani, et al	2018	Using Requirements Clustering	Yes	Using requirements information the productivity has been increased by 80% in prioritizing the test cases. In the testing phase it helps to find the faults and errors easily.	
Junjie Chen, et al	2018	Distribution Analysis	Yes	The results show that Predictive Test Prioritization give the optimal solution for 46 out of 50 projects. It achieve 43.16% to 94.92% improvement.	In future we can extend Predictive Test Prioritization using various measurements to evaluate their effectiveness.
Remo Lachmann	2018	Using ML Driven Test Case Prioritization Approaches	Yes	Best performance is achieved when using logistic regression, which is used for the first time in this paper to solve the test case prioritization problem based on natural language artefacts	Investigating techniques, which are able to improve traceability in a semi-automatic fashion. To investigate the potential of our historical ensemble approach to prioritize test cases.
MaralAzizi et al.	2018	Novel graph-based framework	No	It improves the effectiveness and efficiency. The performance is good as compare to previous techniques.	In future with the same research work we take more data and different metrics In regression testing how we can apply this research in different areas.
Dario Di Nucci et al.	2018	Hypervolume-based Genetic Algorithm	No	The results show that HGA is more cost effective. With the help of HGA algorithm effectiveness is improved.	In future study on the multi-objective methodologies for test case selection and minimization to improve performance of HGA
Pablo Carballo et al. [8]	2018	Biased Random-Key Genetic Algorithm [BRKGA]	No	The result shows that BRKGA outperforms two different permutational based encoding based GA	In future we can use BRKGA algorithm to evaluate large dataset.
Fadel Toure, et al. [26]	2018	Prioritizing Unit Testing Effort Using Software Metrics and Machine Learning Classifiers	Yes	The new classifiers correctly suggest classes to be tested. tested classes are particularly well predicted in the case of large-size systems.	To analyse and compare the actual performance on covering faulty classes.
Garima	2018	Combination of both the techniques Genetic Algorithm and repair reschedule adaptive method	Yes	It maximize the coverage and fault detection cost	In future it is also possible to combine genetic and fix reschedule algorithm
Durelli, et al. [25]	2019	The Systematic Mapping Study using Machine Learning	Yes	For the test-case generation and evaluation machine learning algorithms are used. This paper result is ML algorithms are used for automation of software-testing. To understand the present state of the ML research which is applied for testing the software.	Highlighting the ML is mostly used and widely showed the algorithms and identifying of several avenues for the future research.

III CONCLUSION

Test case optimization is used to minimize the effort required to test the changes by recognizing only the appropriate test cases necessary for the test cycles. The methods of the subject of the machine learning are broadly for the usage of the test case optimization and show for the better outcomes. Various optimization techniques are used for solving the problems of test case which have provided the various results and accuracy. In this paper, various techniques of test case prioritization are reviewed and analyzed in terms of certain parameters.

With the help of the given usage techniques such as Genetic Algorithms, to bring and generate a single objective formulation from a multiple approaches,

Greedy algorithm, an algorithm to boost and speed up the performance better than single objective strategies, including BAT, PSO algorithm .This paper shows an optimization techniques review in which all software testing domain are in used. The main motive of this paper is to identify criteria adequacy and optimization methods which has a lot of future scope to present and show the benefits of algorithm on quality purpose with the use of hybrid technique.

Some of the future scope and works area are as follow

1. Identification of the behavioral adequacy with the help of optimization techniques
2. Comparison and difference of the mutation adequacy and operational coverage.
3. Bringing up on hybrid technique with multi objective algorithm.

REFERENCES

[1] Z. Sultan, R. Abbas, S. Nazir, and S. Asim, "Analytical Review on Test Cases Prioritization Techniques: An Empirical Study," Int. J. Adv. Comput. Sci. Appl., vol. 8, no. 2, pp. 293–302, 2017.

[2] W. Liu, X. Wu, W. Zhang, and Y. Xu, "The research of the test case prioritization algorithm for black box testing," Proc. IEEE Int. Conf. Softw. Eng. Serv. Sci. ICSESS, pp. 37–40, 2014.

[3] D. Paterson, J. Campos, R. Abreu, G. M. Kapfhammer, G. Fraser, and P. McMinn, "An empirical study on the use of defect prediction for test case prioritization," Proc. - 2019 IEEE 12th Int. Conf. Softw. Testing, Verif. Validation, ICST 2019, no. 1, pp. 346–357, 2019.

[4] D. Hao, L. Zhang, and H. Mei, "Test-case prioritization: achievements and challenges," Front. Comput. Sci., vol. 10, no. 5, pp. 769–777, 2016.

[5] C. Lu, J. Zhong, Y. Xue, L. Feng, and J. Zhang, "Ant Colony System With Sorting-Based Local Search for Coverage-Based Test Case Prioritization," IEEE Trans. Reliab., no. 1, pp. 1–15, 2019.

[6] R. Dhiman and V. Chopra, "Novel Approach for Test Case Prioritization Using ACO Algorithm," 2019 IEEE 2nd Int. Conf. Inf. Comput. Technol. ICICT 2019, pp. 292–295, 2019.

[7] M. Computing, "A New Technique for Test Case," vol. 2, no. September, pp. 107–109, 2013.

[8] P. Carballo, P. Perera, S. Rama, and M. Pedemonte, "A Biased Random-Key Genetic Algorithm for Regression Test Case Prioritization," 2018 IEEE Lat. Am. Conf. Comput. Intell. LA-CCI 2018, 2019.

[9] M. Azizi and H. Do, "Graphite: A Greedy Graph-Based Technique for Regression Test Case Prioritization," Proc. - 29th IEEE Int. Symp. Softw. Reliab. Eng. Work.s ISSREW 2018, pp. 245–251, 2018.

[10] D. Di Nucci, A. Panichella, A. Zaidman, and A. De Lucia, "A Test Case Prioritization Genetic Algorithm guided by the Hypervolume Indicator," IEEE Trans. Softw. Eng., vol. PP, no. c, p. 1, 2018.

[11] M. Ozawa, T. Dohi, and H. Okamura, "How Do Software Metrics Affect Test Case Prioritization?," Proc. - Int. Comput. Softw. Appl. Conf., vol. 1, pp. 245–250, 2018.

[12] R. Huang et al., "On the Selection of Strength for Fixed-Strength Interaction Coverage Based Prioritization," Proc. - Int. Comput. Softw. Appl. Conf., vol. 1, pp. 310–315, 2018.

[13] D. Pradhan, S. Wang, S. Ali, T. Yue, and M. Liaaen, "REMAP: Using Rule Mining and Multi-objective Search for Dynamic Test Case Prioritization," Proc. - 2018 IEEE 11th Int. Conf. Softw. Testing, Verif. Validation, ICST 2018, pp. 46–57, 2018.

[14] M. H. Mahmood and M. S. Hosain, "Improving test case prioritization based on practical priority factors," Proc. IEEE Int. Conf. Softw. Eng. Serv. Sci. ICSESS, vol. 2017-November, pp. 899–902, 2018.

[15] R. Huang, W. Zong, D. Towey, Y. Zhou, and J. Chen, "An empirical examination of abstract test case prioritization techniques," Proc. - 2017 IEEE/ACM 39th Int. Conf. Softw. Eng. Companion, ICSE-C 2017, pp. 141–143, 2017.

[16] Sujata and G. N. Purohit, "Classification model for test case prioritization techniques," Proceeding - IEEE Int. Conf. Comput. Commun. Autom. ICCCA 2017, vol. 2017-January, pp. 919–924, 2017.

[17] J. Ding and X. Y. Zhang, "Comparison analysis of two test case prioritization approaches with the core idea of adaptive," Proc. 29th Chinese Control Decis. Conf. CCDC 2017, pp. 1723–1730, 2017.

[18] H. Spieker, A. Gotlieb, D. Marijan, and M. Mossige, "Reinforcement learning for automatic test case prioritization and selection in continuous integration," ISSTA 2017 - Proc. 26th ACM SIGSOFT Int. Symp. Softw. Test. Anal., pp. 12–22, 2017.

[19] J. Chen et al., "Optimizing test prioritization via test distribution analysis," ESEC/FSE 2018 - Proc. 2018 26th ACM Jt. Meet. Eur. Softw. Eng. Conf. Symp. Found. Softw. Eng., pp. 656–667, 2018.[20] R. Lachmann, "Machine Learning-Driven Test Case Prioritization Approaches for Black-Box Software Testing," Eur. Test Telem. Conf., pp. 3s00–309, 2018.

[21] F. Wang, S. C. Yang, and Y. L. Yang, "Regression testing based on neural networks and program slicing techniques," Adv. Intell. Soft Comput., vol. 124, pp. 409–418, 2011.

[22] B. Busjaeger and T. Xie, "Learning for test prioritization: An industrial case study," Proc. ACM SIGSOFT Symp. Found. Softw. Eng., vol. 13-18-November-2016, pp. 975–980, 2016.

[23] M. Noorian, E. Bagheri, and W. Du, "Machine learning-based software testing: Towards a classification framework," SEKE 2011 - Proc. 23rd Int. Conf. Softw. Eng. Knowl. Eng., pp. 225–229, 2011.

[24] R. A. Haraty, N. Mansour, L. Moukahal, and I. Khalil, "Regression Test Cases Prioritization Using Clustering and Code Change Relevance," Int. J. Softw. Eng. Knowl. Eng., vol. 26, no. 5, pp. 733–768, 2016.

[25] V. H. S. Durelli et al., "Machine Learning Applied to Software Testing: A Systematic Mapping Study," IEEE Trans. Reliab., vol. 68, no. 3, pp. 1189–1212, 2019.

[26] F. Toure and M. Badri, "Prioritizing unit testing effort using software metrics and machine learning classifiers," Proc. Int. Conf. Softw. Eng. Knowl. Eng. SEKE, vol. 2018-July, pp. 653–658, 2018.