Improving Test Efficiency through Multiple Criteria Coverage Based Test Case Prioritization

N.Prakash, K.Gomathi

Abstract— Software Testing is the set of activities conducted with the intent of finding bugs in software. It is the most important phase of software development life cycle and consumes significant resources in terms of effort, time and cost. Regression testing is used to reduce the cost of Software testing. It is effective technique to conduct testing with modified piece of code. The purpose of regression testing is to confirm that a recent program or code change has not adversely affected existing features. To reduce the cost and time of the regression testing, Test case prioritization approach is used. Test case prioritization techniques prioritize the test cases in an order that attempts to maximize the objective function such as improving the rate of fault detection. The existing system prioritizes test cases that cover only one coverage criteria. The execution of prioritized test cases for single coverage criteria is not efficient for regression testing. So, testing should be conducted multiple times for each coverage criteria. The proposed system prioritizes test cases which cover more than one criterion such as code coverage, branch coverage, function coverage, path coverage and fault coverage. Thus, the Coverage information is gathered and analyzed both manually and with the help of automation coverage tool. Based on the coverage information, it uses Multiple Criteria Coverage Based Test Case Prioritization method to prioritize the test cases and improves efficiency through execution of single prioritized test cases, for conducting regression testing. The proposed test case prioritization method is empirically studied with three standard applications and it is compared with existing prioritization method. This study shows that the proposed method improves the performance of regression testing.

Index Terms — Coverage Analysis, Coverage Based Testing , Coverage Criteria, Regression Testing, Software Testing, Test Case , Test Case Prioritization.

1 INTRODUCTION

Software Development Life Cycle describes the activities performed at each stage of a software development project. It is also known as software life cycle. Various phases of software development life cycle are planning, requirements gathering and analysis, design, development, integration, testing, training & Implementation. Software Testing is the set of activities conducted with the intent of identifying defects in the software. The main goal of software Testing is as follows: It ensures quality and Customer satisfaction. It is found that one third of the cost can be avoided if better software testing can be performed when building the software product.

This paper aims at improving test efficiency through Multiple criteria coverage based test case prioritization. Test Efficiency is a measure of the percentage of failures detected over the period of test suite execution. Through Multiple Criteria Coverage based Test Case Prioritization system, we aim to increase test efficiency by improving the rate of failure detection. Regression testing is performed when the software or its environment is changed. It is performed to ensure that the defects have not been introduced in unchanged areas of software, as a result of changes made. The widely Known method for Regression Testing is Test Case prioritization. Test Case consists of test input, the entry criteria, the exit criteria, and the expected output. After executing the test cases, the expected result is compared with the actual result. The result indicates whether the software is functioning as desired or not. Test case prioritization approach is one of the approach to reduce time and cost of testing process. The test cases with highest priority can be executed first. The Prioritization of test cases can improve test efficiency. Let us have a glance regarding the need for Test case prioritization approach in Regression Testing. Regression testing phase consumes lot of time and cost to run. There is not enough time or resources to run the entire test suite. There is a need to decide which test cases to be executed first. In this kind of scenario, Test case prioritization is the best option for regression testing. Code coverage provides a measure of how well test suite actually tests the product. Coverage analysis is a way of measuring how much of the code has been exercised during testing. Coverage analysis can be used to determine when sufficient testing has been carried out. An optional aspect is that it is possible to identify redundant test cases that do not increase coverage. Various types of coverages are Statement coverage, Basic block coverage, Branch coverage, Path coverage, and Loop coverage. Let us discuss about that the importance of multiple coverage critetia. Consider the following code which is used to compute absolute value:

```
if( A>=0)
  A=0-A;
  abs=A;
```

when testing this code with test case A=0, it is sufficient for statement coverage. The output is 0. It indicates that the code is working exactly and covers all statements. But, this test

- *N.Prakash is currently working as Assistant Professor (Selection Grade) in Department of Information Technology, B.S.Abdur Rahman University, Chennai. E-mail: prakash@bsauniv.ac.in*
- *K.Gomathi is currently pursuing master's degree program in Department of Information Technology, B.S.Abdur Rahman University, Chennai. E-mail: itathi12@gmail.com*

case is not sufficient; it does not reveal the fact that the absolute value is not being computed by this code for all inputs. Because, for positive value the output will be a negative value and for negative value the output will also be a negative value. Therefore, the statement coverage testing alone is not enough to test the above code. Then another testing technique must be applied to check the code. Branch coverage is an appropriate testing technique. The purpose of the branch coverage is that each decision is evaluated to true and false at least once. Two test cases A=-2 and A=0 are required to execute both branches of the decision. By giving test input A=0, output is abs=0 and for input A=-2, output is abs=-2, which indicates the existence of fault. The above discussion indicates that the statement coverage testing is alone not enough, we need to apply other testing techniques also. Thus it is obvious that the multiple criteria coverage based test case prioritization can reveal more faults.

2 RELATED WORK

Test case prioritization techniques aim to increase the efficiency of regression testing by reordering the test cases to increase the likelihood of fault detection ability. Many research works have been carried out, but major focus is on single coverage criterion such as statement coverage, branch coverage, block coverage, fault coverage etc.

Hema Srikanth et al. [2] proposed test case prioritization approach called PORT and the goal of their research is to develop and validate a system level test case prioritization scheme for identifying the more severe failures earlier in system test. PORT prioritizes system-level black box tests by considering four prioritization factors: Customer Priority, Implementation Complexity, Fault Proneness, and Requirements Volatility. PORT is designed to be an easy–to-use scheme where in factor values can be collected by the engineering team with minimal effort. The open source support Requirements based testing (ReBaTe) Tool automates the ordering of system test cases according to PORT scheme. The result shows that the PORT method improves the test case prioritization than the random method. It concludes customer priority is the most significant contributor.

Krishnamoorthi et al. [13] proposed new method to prioritize test case prioritization method. This is a model for system level Test case prioritization from SRS, which aims at 1)Improving user satisfaction 2)Achieving cost effectiveness 3)Improving the rate of severe fault detection. The Prioritization is done based on 6 factors such as Customer Priority, Changes in Requirement, Implementation Complexity, Completeness, Traceability, and Fault Impact. The factors customer-assigned priority, implementation complexity, requirement changes are the new test case factors. Fault impact of requirements, completeness and traceability are the regression test case factors. They conducted feasibility study in 3 phases to measure the effectiveness of the proposed prioritization technique. The Results proved that there is improved rate of detection of faults than random ordering of test cases. Also it is tested experimentally that the number of test cases executed to find the injected fault is less in case of proposed prioritized execution of test cases.

Siripong et al. [14] conducted survey and listed survey result of existing test case prioritization techniques. They proposed new test case prioritization method along with practical weight factors. Also, this paper discusses about research problems.1) Ignoring practical weight factors, 2) Inefficient test case prioritization methods, 3) Ignoring the size of test cases. This paper proposed TCP process called 2R-2S-3R, introduces new practical set of weight factors used in TCP process. The new process contains two processes named 2R -Requisite and Reordering. The first process consists of two sub-processes, called 2s which are select test case prioritization technique and specify coverage or factors. The second process is composed of three sub-processes called 3R, includes re-assign weight value, re-calculate priority value and re-order test cases. This method reserves the large number of high priority test cases than random method employed in Hema's technique and Alexey's work.

Gregg Rothermel et al. [8] conducted an empirical study to prioritize the test cases for various prioritization techniques. This paper described several techniques for test case prioritization such as no prioritization, Random prioritization, Optimal prioritization, Total branch coverage prioritization, Additional branch coverage prioritization, Total statement coverage prioritization, Additional statement coverage prioritization, Total fault-exposing potential prioritization. And thus, this paper gives overview of various techniques for prioritizing test cases. Thus, it projects on performing studies utilizing programs and types of test suites. Also, it gives results measuring the effectiveness of these techniques for improving rate of fault detection of test suites.

Hyunsook Do et al. [5] conducted an empirical study to assess the effects of time constraints on the cost and time benefits of prioritization techniques. It is based on the results of series of experiments to access the effects of time constraints on the cost and benefits of prioritization techniques. The first experiment manipulates time constraint levels and shows that time constraints play significant role in determining both the cost-effectiveness of prioritization and the relative cost-benefit trade-offs among techniques. The second experiment replicates the first experiment controlling for several threats to validity including number of faults present, and shows that the results generalize to this wider context. Third experiment manipulates the number of faults present in the programs to examine the effects of faultiness levels on prioritization and shows the faultiness level affects the relative cost-effectiveness of prioritization techniques. The results have several implications for test engineers wishing cost-effectively regression test their software systems. It includes suggestions about when and when not to prioritize, techniques to employ and differences in testing process may relate to prioritization cost-effectiveness.

Sihan li [11] conducted simulation study and this paper is based on the study conducted to perform a simulation experiment to study five search algorithms for test case prioritization namely Greedy algorithm, Additional Greedy Algorithm, 2-optimal Greedy algorithm, Hill climbing algorithm and genetic algorithm. Also it compares the performance of the above said algorithms. This simulation study provides two useful guidelines: two search algorithms namely Additional

greedy algorithm and 2-optimal greedy algorithm, outperform the other three search algorithms in most cases. The performance of five search algorithm will be affected by the overlap of test cases with regard to test requirements. The results indicate that the 2 search algorithms, additional greedy algorithm and 2-optimal greedy algorithm performs better.

Zheng Li et al. [10] conducted empirical study and Sihan li Conducted simulation study to prioritize the test cases using greedy algorithm, additional greedy algorithm, 2-optimal greedy algorithm and genetic algorithm. This paper presented the results of an empirical study that investigated their relative effectiveness. Data and analysis indicates that the greedy algorithm performs much worse than Additional Greedy, 2-optimal and genetic algorithms overall. Also it shows that 2-optimal algorithm overcomes weakness of other algorithms. The experiments however indicate that, in terms of effectiveness, there is no significant difference between the performances of 2-optimal and additional greedy algorithms. This suggests that, where applicable the cheaper to implement and execute Additional Greedy algorithm should be used. The criteria studied here is based on code coverage, criteria based on fault detection can be future work.

3 Concept and Problem Definintion

Regression testing will be conducted in time critical situations. To conduct regression testing more effectively, different techniques are proposed. These are random testing, test case selection, test case reduction, and test case prioritization. Among these techniques, Test case prioritization is most effective technique.

In Existing Systems, Test case prioritization approaches uses Single Coverage Criteria. Following are the issues in Single Coverage Criteria 1) Execution of prioritized test cases for single coverage criteria are not efficient for regression testing. 2) Testing should be conducted multiple times for each coverage criteria. To ensure quality, more than one prioritization techniques are required to be executed during regression testing. But it is time consuming and more expensive process. To overcome these problems, Multiple Criteria Coverage Based Test Case Prioritization method is proposed.

4 System Architecture

The system architecture is shown in Figure 1, and it clearly outlines every module. The main objective of this system is to prioritize the test cases based on its coverage ability of the test cases. So, regression testing can be conducted efficiently in single attempt. Thus it is possible to achieve more than one performance goals such as statement coverage, path coverage, branch coverage, function coverage and fault coverage. The software with coverage analysis Information and concerned Test cases in random order are considered. The coverage criteria such as statement coverage, branch coverage, path coverage, fault coverage and function coverage are considered to prioritize the test cases. While analyzing the coverage information, any of the automation tool can be selected for Justification process. The automation tool is used to collect coverage history of coverage criteria for every test case. The weight of coverage criteria is calculated based on coverage information of each coverage criteria for every test case. Then average weight is calculated for each test case, and the test case is prioritized based on its weight. The highest weight is considered for high priority and lowest weight is considered for low priority. By reordering the test cases based on its weight value, Test cases can be prioritized.

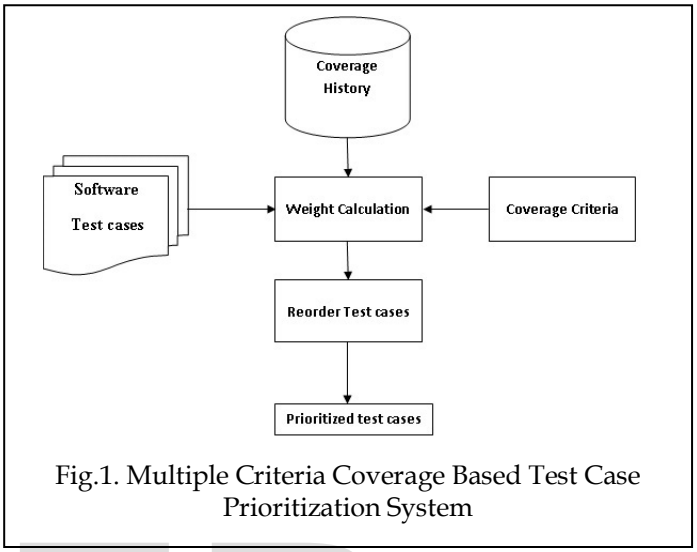

Fig.1. Multiple Criteria Coverage Based Test Case Prioritization System

Thus, the proposed work consists of three main modules such as 1.Analysis of coverage information, 2.Weight Calculation and 3.Reorder Test Cases.

4.1 Analysis of coverage information

This module mainly involves data collection and analysis. In this module, detailed study is carried out regarding collection of various coverage criteria available. Then the coverage criteria to be considered to prioritize the test cases such as code coverage, branch coverage, path coverage, function coverage and fault coverage are selected. Then the application developed in any language which can support coverage based tool for justification is selected with suitable test cases. In this case, Java projects and coverage automation tool named EclEmma is selected. Then for each test case, associated coverage criteria are identified. Finally the total count of each criterion which is covered by each test case is calculated.

Table 1: Test cases and its associated Statements, Functions, Paths, Branches and Faults

Test Case	Statement Coverage	Function Coverage	Path Coverage	Branch Coverage	Faults Coverage
T1	36	4	4	4	0
T2	27	3	8	4	1
T3	41	2	0	6	4
T4	29	2	8	0	1
T5	34	2	4	4	0

The above Table shows Test cases and its associated Statements, functions, paths, branches and faults.

The Figure 2 shows the code coverage session view of Auto-

mation Tool.

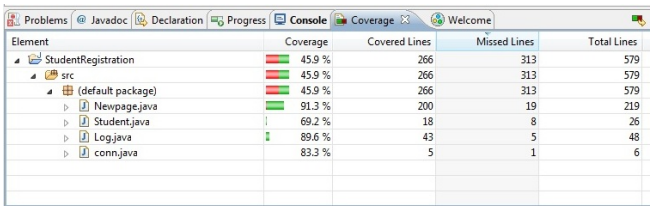

Fig.2: Coverage session view

4.2 Weight Calculation

Test cases with its associated count of statement coverage, function coverage, path coverage, branch coverage and fault coverage for selected application should be listed for calculation. The weight assigned for each test case is 1 to 10 for uniformity. Weight value 10 will be considered as high value and 1 will be considered as low value, where 0 indicates the associated test case does not cover any criteria. Thus in this module, first step is the calculation of each coverage criteria for each test case. Then the overall weighting factor for each test case is calculated.

The following algorithm shows step by step procedure of the proposed Multiple Criteria Coverage Based Test Case Prioritization System.

Input: Test suite T.
Output: Prioritized Test suite T'.
Begin
for each test case t ϵ T **do**
collect coverage information for statement, branch, method and path coverage criteria
end for
for each coverage criteria(cc) **do**
for each test case t ϵ T **do**
calculate test case weight as $W_{cc}\ T_i = \frac{N_{CI}}{M_{CI}} * 10$
end for
end for
for each test case t ϵ T **do**
calculate overall weight as $OWTC_i = \frac{\sum_{j=1}^{n} W x_j T_i}{n}$
end for
sort T_i in descending order based on OWTC value of each test case
let T' be T

end
where,
$W_{cc}\ T_i$ = weight for each coverage criteria for test case T_i
N_{CI} = Coverage information for T_i
M_{CI} = Maximum coverage information for CC
x_j = Test case criteria
n = Total number of criteria

The Below table shows Test Cases and its weights for codes, functions, paths, branches and faults coverage.

Table 2 Test Cases and its weights for codes, functions, paths, branches and faults coverage

Test Case	Weight for Code Coverage	Weight for Function Coverage	Weight for Path Coverage	Weight for Branch Coverage	Weight for Faults Coverage	Test Case Weight
T1	8.78	10	5	6.66	0	6.08
T2	6.58	7.5	10	6.66	2.5	6.64
T3	10	5	0	10	10	7
T4	7.07	5	10	0	2.5	4.91
T5	8.29	5	5	6.66	0	4.99

4.3 Reorder Test Cases

In this module, based on the weight value from maximum to minimum, the test cases are sorted form maximum to minimum. Thus, the prioritized Test cases are obtained as the output. The below table shows the Prioritized Test Cases and its weights for codes, functions, paths, branches and faults coverage.

Table 3 Prioritized Test Cases and its weights for codes, functions, paths, branches and faults coverage

Test Case	Weight for Code Coverage	Weight for Function Coverage	Weight for Path Coverage	Weight for Branch Coverage	Weight for Faults Coverage	Test Case Weight
T3	10	5	0	10	10	7
T2	6.58	7.5	10	6.66	2.5	6.64
T1	8.78	10	5	6.66	0	6.08
T5	8.29	5	5	6.66	0	4.99
T5	8.29	5	5	6.66	0	4.99

5 Experimental Verification and Result Analysis

This section describes the experimental verification of the proposed multiple criteria coverage based test case prioritization method. Three applications which are developed in Java are considered to analyze the coverage information both manually and with automation coverage tool. The following table shows Subject programs and its characteristics.

Table 4: Subject programs and its characteristics

Application Software	LOC (Lines of Code)	No of Modules	No of Test Cases
Hospital Management System	1722	6	73
Library Management System	3594	9	91
Student Registration System	579	3	45

For the above programs, given the input and outputs are verified. Manual computation is done to calculate coverage information. The coverage criteria with its coverage information are verified with the coverage tool "cobertura" and "EclEmma". After collecting coverage History, Test case weight is calculated as per algorithm of Multiple Criteria Coverage Based Test Case Prioritization and test cases are prioritized based on its weight.

6 CONCLUSION

In this paper entirely new approach for Test Case Prioritization is introduced called Multiple Criteria Coverage Based Test Case Prioritization, which prioritizes test cases efficiently for regression testing. Three applications namely Hospital Management system, Library Management system, Student Registration system is considered to verify the efficiency of proposed method. The coverage criteria for the above application are collected. More than one coverage criteria such as statement coverage, path coverage, branch coverage, function coverage and fault coverage are taken in to consideration for the above said applications. Manual computation is done to check the statement coverage, path coverage, branch coverage, function coverage and fault coverage for the Application. Also, it is justified with the coverage tool named "Cobertura" and "EclEmma". Weight Calculation is done for coverage criteria and for test cases. Then the test cases are reordered based on its weightage for respective Test cases. It is obvious from this study that Multiple Criteria Coverage Based Test Case Prioritization improves rate of fault detection. For certain Test Cases, we obtained similar Test Case Weight. In future, it will be studied and improved in future work.

REFERENCES

[1] Harrold M, Testing: A Roadmap, 2000, *International Conference on Software Engineering*, Limerick,Ireland, pp. 61-72.

[2] Hema Srikanth and Sean Banerjee, 2012, Improving Test Efficiency Through System Test Prioritization, *The Journal of Systems and Software, Elsevier*, Vol., No., pp. 1176-1187.

[3] James F. Peters and Witold Pedrycs, 2000, Software Engineering-An Engineering Approach, John Wiley & Sons Inc. pp. -461.

[4] Hema Srikanth, Laurie Williams, and Jason Osborne, 2005, System Test Case Prioritization of New and Regression Test Cases, *IEEE*.

[5] Hyunsook, Siavash Mirarab, 2010, The Effects of time constraints on Test Case Prioritization : A Series of controlled experiments., *IEEE Transaction on Software Engineering*, Oct 2001.

[6] Rothermel. G, R. Untch, C. Chu, and M.J. Harrold, 1999, Test Case Prioritization: An Empirical Study, Proc. *Int'l Conf. Software Maintenance*, pp. 179-188, Sept.

[7] Rothermel. G, R. Untch, C. Chu, and M.J. Harrold, 2001, Prioritizing Test Cases for Regression Testing, *IEEE Trans. Software Eng.*, vol. 27, no. 10, pp. 929-948, Oct.

[8] Elbaum S., Alexey G. Malishevsky, and Gregg Rothermel, 2002, Test Case Prioritization: A Family of Empirical Studies, *IEEE Transaction on Software Engineering*, Vol 28, No. 2, February, pp 159-182.

[9] Elbaum S, Gregg Rothermel, Satya Kanduri, and Alexey G. Malishevsky, 2004, Selecting a Cost-Effective Test Case Prioritization Technique, *Journal Software Quality Control*, Vol. 12 No. 3,Sep.

[10] Zheng Li, Mark Harman, and Robert M. Hierons, 2007, Search Algorithms for Regression Test Case Prioritization, *IEEE Transactions on Software Engineering*, Vol. 33, No. 4. pp. 225-237.

[11] Sihan Li, Naiwen Bian, Zhenyu Chen, Dongjiang You, Yuchen He, 2010, A Simulation Study on Some Search Algorithms for Regression Test Case Prioritization, *IEEE*, pp. 72-81.

[12] Yoo S, Harman M, 2010, Regression testing minimization, selection and prioritization: a survey, *Wiley Online Library*.

[13] Krishnamoorthi R and Sahaaya Arul Mary S. A, 2009, Factor Oriented Requirement Coverage Based System Test Case Prioritization of New and Regression Test Cases, *Internatonal Journal of Information and Software Technology, Elsvevier*, No. 53, pp. 799-808.

[14] Siripong Roongruangsuwan and Jirapun Daengdej, 2010, A Test Case Prioritization with Practical Weight Factors, *Journal of Software Engineering, Academic Journals Inc.*, Vol 4, No. 3, pp. 193-214.

[15] Salehie M, Sen Li, Tahvildari L, Dara R, Shimin Li, and Moore M, 2011, Prioritizing Requirements-Based Regression Test Cases: A Goal-Driven Practice, i, Mar. 2011, pp. 329-332.

[16] Yu-Chi Huang, Chin-Yu Huang, Jun-Ru Chang and Tsan-Yuan Chen, 2010, Design and Analysis of Cost-Cognizant Test Case Prioritization Using Genetic Algorithm with Test History, *IEEE*, pp.413-418.