Journal of Theoretical and Applied Information Technology
20th October 2013. Vol. 56 No.2

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

WEIGHTED METHOD FOR COVERAGE BASED TEST CASE PRIORITIZATION

[1]PRAKASH N, [2]RANGASWAMY T.R

[1]Assistant Professor (sel. Gr.), Department Of Information Technology,
B.S.A Crescent Engineering College, Chennai, India
[2]Dean Academic Affairs, B.S.A Crescent Engineering College, Chennai, India
Email: [1]prakash@bsauniv.ac.in, [2]deanacademic@bsauniv.ac.in

ABSTRACT

Software will be modified either to fix errors or to add features. The modified software should be tested with the intent of the modified parts of the software without affecting the other parts of the software. Regression testing is one of the testing techniques to ensure that the modified piece of code without affecting other parts of code. Test case prioritization is a widely used technique to conduct regression testing in effective manner. All existing prioritization techniques aims to prioritize the test cases based on certain coverage criteria. These prioritization techniques take into account any one coverage criteria (single performance goal) such as statement coverage, function coverage, path coverage, branch coverage and fault coverage. There are certain situation single performance goal will not find faults earlier using test case prioritization. So prioritization of test cases, which cover more than one criterion is an open problem. To alleviate this problem, a new weighted method is proposed to prioritize the test cases. In this method the weight is calculated based on the coverage criteria for each test case. The proposed method was empirically studied in three standard applications and the results show that the proposed method is more effective than the existing method.

Keywords: *Weighted method,. Test Case, regression test, Prioritization, coverage criteria*

1. INTRODUCTION

As quality point of view, software testing is an important activity. Testing may consume around 50 percent of the total software development cost [1-2]. Software industries are often faced with lack of time and resources, which limits their ability to effectively complete testing process [3-4]. Regression testing will be conducted in a time critical situation. To conduct the regression test more effectively different techniques are proposed in the literature [5-7. These are random testing, test case selection, test case reduction, and test case prioritization. Among these techniques test case prioritization is widely used in the software industry. Most of the existing test case prioritization techniques are based on certain coverage criteria. That is the test cases are prioritized based on certain performance goal such as fault coverage, statement coverage, path coverage, branch coverage, and function coverage. Most of the existing research work on test case prioritization is based on single performance goal. Consider the following piece of code [8], which is used to compute the absolute value.

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

To test this code a test case A=0 is enough for statement coverage. The output of the given test case is 0. It shows that the code is working correctly and covers all statements. But, this test case is not enough; it does not tell the absolute value is not being computed by this code. 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 given 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 adequate to execute both branches of the decision. For A=0, we get abs=0 and for A=-2 we get abs=-2, which indicates the existence of fault.

The above discussions shows that the statement coverage testing is alone not enough, we need to apply other testing techniques also. To address the above scenario two possible solutions are available for regression testing. First one is to apply more

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

than one prioritized set of test cases based on different coverage criteria. Second one is single prioritized test cases which are computed with multiple coverage criterions. Regression testing will be conducted in a time critical situation. Therefore the second solution is more appropriate. Most of the research work on test case prioritization is based on single coverage criteria.

In this work a new weighted methods for coverage based test case prioritization is proposed. The main objective of the proposed work is to prioritize the test case which covers more than one coverage criteria for a single set of test cases. To prioritize the test case a weight is calculated based on its coverage criteria for each criteria. In this work statement coverage, function coverage, path coverage, branch coverage, and fault coverage are considered to prioritize the test case.

The rest of this paper is organized as follows: Section 2 of this article presents related works on the test case prioritization. Section 3 presents the proposed weighted method for coverage based test case prioritization. Section 4 presents experimental verification and result analysis. Section 5 concludes the results and discusses further work.

2. RELATED WORKS

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 fault coverage, statement coverage, block coverage, branch coverage etc.

Hema Srikanth et al. proposed PORT approach to prioritize system test cases based on customer priority, implementation complexity, fault proneness, and requirements volatility. The result shows that the PORT method improves the test case prioritization than the random method.

Gregg Rothermel et al. [5] and Elbaum et al. [10-11] conducted an empirical study to prioritize the test cases for various prioritization techniques. The empirical results show that the test case prioritization techniques improve the rate of fault detection.

Hyunsook Do et al. [9] conducted an empirical study to assess the effects of time constraints on the costs and benefits of prioritization techniques. The empirical results show that the time constraints play a significant role in determining the cost effectiveness of the prioritization techniques.

Zheng Li et al. [12] conducted empirical study and Sihan Li et al. [13] conducted simulation study to prioritize the test cases using greedy algorithm, additional greedy algorithm, 2-optimal greedy algorithm and genetic algorithm. The empirical study shows that the additional greedy algorithm and 2-optimal greedy algorithm performed well than the genetic algorithm and greedy algorithm.

S. Yoo et al. [14] conducted a survey and detailed analysis of trends in regression test case selection, minimization and prioritization. The analysis report suggests that the topic of test case prioritization is of increasing importance.

Krishnamoorthi et al. [15] proposed new method to prioritize test case from software requirement specification to improve user satisfaction based customer priority, changes in requirement, implementation complexity, completeness, traceability and fault impact.

In the literature, several research works have been carried out to prioritize the test cases more effectively based on single coverage criteria or single objective, multiple objectives and cost effective method for white box testing.

At present regression tests are conducted based on single 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 a weighted method for coverage based test case prioritization is proposed.

3. WEIGHTED METHODS FOR COVERAGE BASED TEST CASE PRIORITIZATION

In this section the proposed weighted methods for coverage based test case prioritization technique is discussed. The importance of more than one criterion for test case prioritization is discussed in section I. The five coverage criteria such as (1) statement coverage, (2) function coverage, (3) path coverage, (4) branch coverage and (5) fault coverage are considered to prioritize the test cases. Overview of the proposed weighted method for test case prioritization is show in Fig. 1. The weight is calculated for every test case and for each coverage criteria based on the coverage information. Then average weight is calculated for each test case, and the test case is prioritized based on the weights. The highest weight is considered for high priority and the lowest weight is considered for low priority.

Journal of Theoretical and Applied Information Technology
20th October 2013. Vol. 56 No.2

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

3.1 Steps Involved in the Proposed prioritization method

The proposed weighted method for coverage based test case prioritization is consisting of the following steps.

Step 1: Select the coverage criterion to be considered to prioritize the test case to achieve multiple performance goals.

Step 2: Collect the total count of each criteria which is covered by each test case.

Step 3: Calculate weight for each test case based on its associated coverage criteria.

Step 4: Calculate overall weighting factor for each test case.

Step 5: Sort the test case based on its weight value from maximu to minimum.

3.2 Weight Calculation Process.

Test cases from T1 to T9 and its associated count of statement coverage, function coverage, path coverage, branch coverage and fault coverage are given in Table 1 for a sample application. 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.

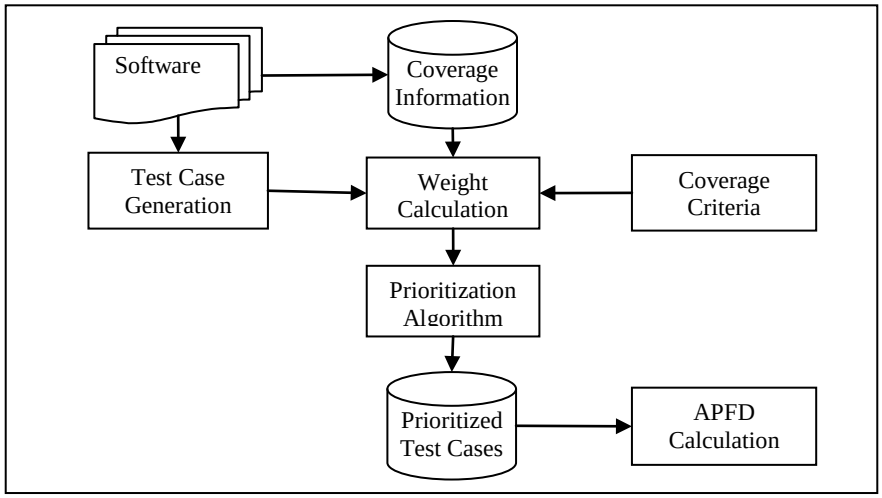


Figure 1. Overview Of Proposed Prioritization Method

Table 1 Test Cases And Its Associated Codes, Functions, Paths, Branches And Faults

Test Case	Statement coverage	Function Coverage	Path Coverage	Branch Coverage	Faults Coverage
T1	25	1	0	2	1
T2	22	1	0	2	2
T3	36	4	4	4	0
T4	18	2	0	2	0
T5	27	3	8	4	1
T6	41	2	0	6	4
T7	34	2	4	4	0
T8	29	2	8	0	1
T9	16	4	2	0	0

a) Test case weight for statement coverage

The main objective of statement coverage testing is that every statement in the code has been executed at least once. Test case weight for the test cases T_i is calculated based on number of codes covered by the test case with respect to the total number of code in the program and the value ranges from a scale of 1 to 10. The weight $W_{cd}T_i$ for test case T_i can be computed by dividing the number of code covered by the test case T_i to the highest number of code covered by any test case T_i. If test

Journal of Theoretical and Applied Information Technology
20th October 2013. Vol. 56 No.2

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

case T_i covers N_{cd} codes and the highest number of codes covered by the test case T_i is M_{cd} then the weight $W_{cd}T_i$ is calculated as,

$$W_{cd}T_i = \frac{N_{cd}}{M_{cd}} * 10 \quad (1)$$

Where: N_{cd} – number of codes covered by the test case T_i

M_{cd} – maximum number of codes covered by any test case T_i

The weights for statement coverage, function coverage, path coverage, branch coverage and fault coverage are presented in Table 2 for test case T1 to T9. The calculated weight for $W_{cd}T_1$ is 6.1 as shown in Table 2. for the sample test case given in Table 1.

b) Test case weigh for function coverage

The main objective of the function coverage testing is that every function in the code has been executed atleast once. Weight for test case T_i for function coverage is calculated based on number of functions covered by the test case with respect to the total number of functions in a program and the value ranges from 1 to 10. The weight $W_{fn}T_i$ for test case T_i can be calculated by dividing the number of functions covered by the test case T_i to the maximum number of functions covered by any test case T_i

$$W_{fn}T_i = \frac{N_{fn}}{M_{fn}} * 10 \quad (2)$$

Where: N_{fn} – number of functions covered by the test case T_i

M_{fn} – maximum number of functions covered by any test case T_i

The calculated weight for $W_{fn}T_1$ is 2.5 as shown in Table 2. for the sample test case which is given in Table 1.

Table 2 Test Cases And Its Weights For Codes, Functions, Paths, Branches And Faults Coverage

Test Case	Weight for Statement coverage	Weight for Function Coverage	Weight for Path Coverage	Weight for Branch Coverage	Weight for Faults Coverage	Test Case Weight
T1	6.1	2.5	0	3.33	2.5	2.89
T2	5.37	2.5	0	3.33	5	3.24
T3	8.78	10	5	6.67	0	6.09
T4	4.39	5	0	3.33	0	2.54
T5	6.59	7.5	10	6.67	2.5	6.65
T6	10	5	0	10	10	7.00
T7	8.29	5	5	6.67	0	4.99
T8	7.07	5	10	0	2.5	4.91
T9	3.9	10	2.5	0	0	3.28
Total	60.49	52.5	32.5	40	22.5	41.60

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
T6	10	5	0	10	10	7.00
T5	6.59	7.5	10	6.67	2.5	6.65
T3	8.78	10	5	6.67	0	6.09
T7	8.29	5	5	6.67	0	4.99
T8	7.07	5	10	0	2.5	4.91
T9	3.9	10	2.5	0	0	3.28
T2	5.37	2.5	0	3.33	5	3.24
T1	6.1	2.5	0	3.33	2.5	2.89
T4	4.39	5	0	3.33	0	2.54
Total	60.49	52.5	32.5	40	22.5	41.60

c) Test case weigh for path coverage

A path is a unique sequence of branches from the functions entry to exit. The main objective of path coverage is that every path in the code has been

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

executed atleast once. The weight for a test case T_i for path coverage is calculated based on number of paths in a program which is covered by the test case T_i. The weight for the path coverage $W_{pt}T_i$ for the test case T_i can be calculated by dividing the number of paths covered by the test case T_i to the maximum number of paths covered by any test case T_i in the test cases and given in equation (3).

$$W_{pt}T_i = \frac{N_{pt}}{M_{pt}} * 10 \quad (3)$$

Where : N_{pt} – number of paths covered by the test case T_i

M_{pt} – maximum number of paths covered by any test case T_i

Weight for $W_{pt}T_1$ is 0 for the sample test case given in Table 1. Because the test case T_1 did not have its associated paths.

d) Test case weigh for branch coverage

The main objective of the branch coverage is that each branch in the code is executed at least once. This requires that each decision is evaluated to true and false at least once. With the total of N_{br} branches covered by the test case Ti and M_{br} branch is the maximum number of branches covered by the test case Ti then the weight for the test case Ti is calculated as follows.

$$W_{br}T_i = \frac{N_{br}}{M_{br}} * 10 \quad (4)$$

Where: N_{br} – number of branches covered by the test case T_i

M_{br} – maximum number of branches covered by any test case T_i

The weights for branch coverage for the sample test cases of Table 1 are depicted in Table 2.

e) Test case weigh for fault coverage

A fault is a result of incorrect functionality of the program. Fault coverage is referred as the ratio of detected faults to the total faults. Main objective of the fault coverage is that each fault in the code is covered at least once. The weight for the fault coverage $W_{fl}T_i$ for the test case T_i is calculated by dividing the number of faults covered by the test case T_i to the maximum number of faults covered by the test case T_i in the test case list. The fault coverage weight $W_{fl}T_i$ is calculated as follows

$$W_{fl}T_i = \frac{N_{fl}}{M_{fl}} * 10 \quad (5)$$

Where: N_{fl} – number of faults covered by the test case T_i

M_{fl} – maximum number of faults covered by any test case T_i

The calculated weight for the test case T_1 for its associated fault coverage is 2.5 given in Table 2.

For n test cases and m coverage criterion, the overall weight for each test case *TCW* is as follows,

$$TCW_i = \frac{\sum_{j=1}^{m} Wx_j T_i}{m} \quad (6)$$

where

x_j – test case criteria

m – total number of criterion

The calculated weight value for test case T1 to T9 of the sample application based on its associated coverage criteria and overall weight for each test case is given in the Table 2. The prioritized test case order is T6, T5, T3, T7, T8, T9, T2, T1, and T4. Prioritized Test Cases and its weights for codes, functions, paths, branches and faults coverage are given in Table 3.

4. EXPERIMENTAL VERIFICATION AND RESULT ANALYSIS

This section describes the experimental verification of the proposed weighted method for coverage based test case prioritization. Three softwares namely University Student Monitoring System (USMS) which is developed in C++, Hospital Management System (HMS) which is developed in Visual Basic, and Industrial Process Operation System (IPOS) which is developed in Java are considered to verify the practicality of the proposed method. The details of empirical studies conducted for three experiments are given in the Table 4.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Table. 4 Subject Programs And Its Characteristics.

Program	Language	Lines Of Code	No. of Test Cases	No. of Modules	No. of Faults
USMS	VB	525	60	5	33
HMS	C++	900	122	6	39
IPOS	Java	1750	161	9	75

To measure the effectiveness of the proposed method, a modified Average Percentage of Fault Detection (APFD) metrics is used. Let *T* be a test suite containing *n* test cases and let *cd* be a set of *m* codes revealed by *T*. Let Tcd_i be the first test case in the reordered test suite *T'* of *T* that reveals fault *i*. The average percentage of statement coverage ($AP_{cd}C$) is calculated as,

$$AP_{cd}C = 1 - \frac{\sum_{i=1}^{m} Tcd_i}{nm} + \frac{1}{n} \quad (7)$$

Similarly Average Percentage of function Coverage ($AP_{fn}C$), Average Percentage of path Coverage ($AP_{pt}C$), Average Percentage of branch Coverage ($AP_{br}C$), and Average Percentage of fault Coverage ($AP_{fl}C$) is calculated using the formula 8, 9, 10, and 11 respectively.

$$AP_{fn}C = 1 - \frac{\sum_{i=1}^{m} Tfn_i}{nm} + \frac{1}{n} \quad (8)$$

$$AP_{pt}C = 1 - \frac{\sum_{i=1}^{m} Tpt_i}{nm} + \frac{1}{n} \quad (9)$$

$$AP_{br}C = 1 - \frac{\sum_{i=1}^{m} Tbr_i}{nm} + \frac{1}{n} \quad (10)$$

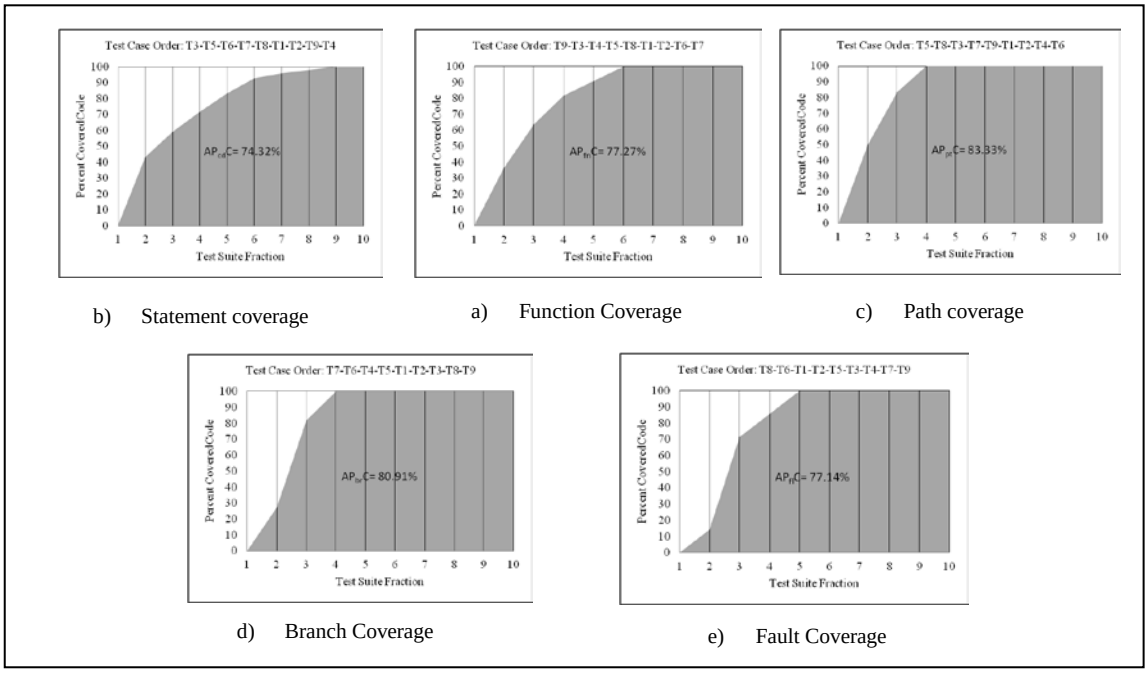
$$AP_{fl}C = 1 - \frac{\sum_{i=1}^{m} Tfl_i}{nm} + \frac{1}{n} \quad (11)$$

After detailed simulation study by Sihan Li et al. (2010) the additional greedy algorithm and 2-optimal greedy algorithm performed well than greedy algorithm and genetic algorithm. Therefore 2-optimal greedy algorithm has been proven as suiTable algorithm to compare with the proposed test case prioritization method. Here after it is called as normal test case prioritization method. The normal test case prioritization method is said to prioritize the test case based on single coverage criteria. The average percent covered by each criterion using normal test case prioritization is given in Fig. 2 and where, the test case order vary for each coverage criteria. Fig. 3 shows the average percent covered by each criterion using proposed test case prioritization and the test case order is same for all coverage criteria. Average percentage of statement coverage is 74.32% in normal test case prioritization as shown in Fig 2.a and 74.63% in the proposed test case prioritization method as shown in Fig 3.a. The statement coverage for the proposed method is 0.31% greater than the normal test case prioritization which is a negligible difference.

Average percentage of function coverage is 77.27% in normal test case prioritization method as shown in Fig 2.b and 77.27% in the proposed test case prioritization method as shown in Fig 3.b. The function coverage in proposed method is equal to the normal test case prioritization. Average percentage of path coverage is 80.91% in normal test case prioritization shown in Fig 2.c and 81.82% for the proposed test case prioritization method as shown in Fig 3.c. Where is the proposed method is 11.66% lesser than the normal test case prioritization. Average percentage of branch coverage is 80.91% in normal test case prioritization as shown in Fig 2.d and 81.82% in the proposed test case prioritization method shown in Fig 3.d and where the proposed method yields 0.91% improved performance than the normal test case prioritization. Average percentage of fault coverage is 77.14% in normal test case prioritization as shown in Fig 2.e and 78.57 in the proposed test case prioritization method shown in Fig 3.e. In this case both the proposed method 1.43% greater than the normal test case prioritization. On the outset of this comparison, the statement coverage, branch coverage and fault coverage yields improved performance and then function coverage yields same performance. Therefore, a single test case order is covered more than one criterion. It proves that the proposed method shows an improved performance than the normal test case prioritization.

Journal of Theoretical and Applied Information Technology
20th October 2013. Vol. 56 No.2

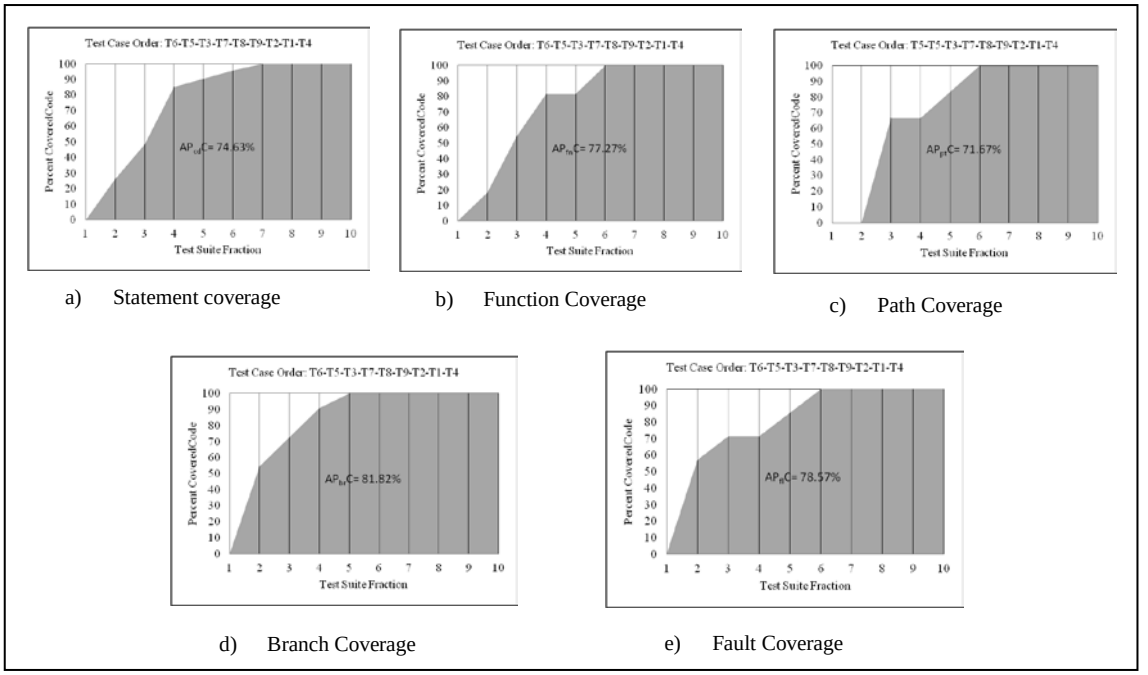
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195



b) Statement coverage a) Function Coverage c) Path coverage

d) Branch Coverage e) Fault Coverage

Figure 2. Average Percent Covered By Each Criterion Using Normal Test Case Prioritization Method



a) Statement coverage b) Function Coverage c) Path Coverage

d) Branch Coverage e) Fault Coverage

Figure 3. Average Percent Covered By Each Criterion Using Proposed Weighted Test Case Prioritization Method

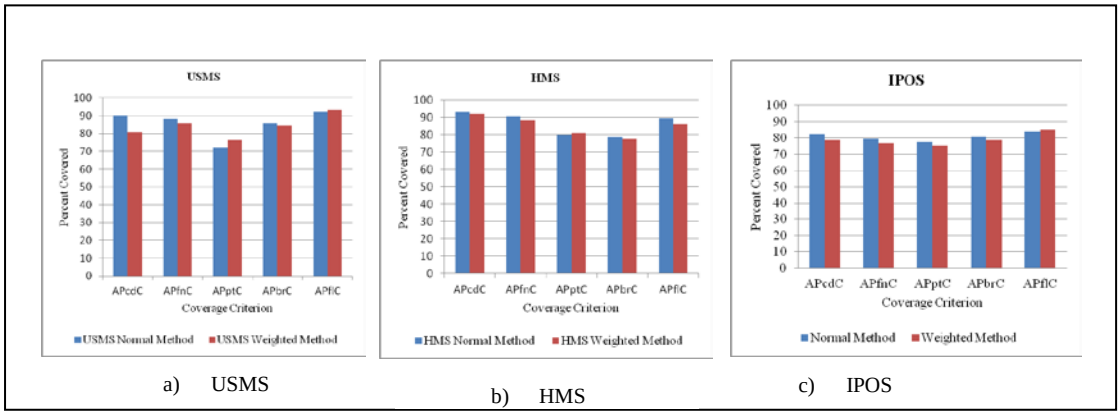
The proposed test case prioritization method is applied for the three applications namely USMS, HMS, and IPOS. The summary of average percentage of different coverage criteria for USMS, HMS, and IPOS are shown in Table 5. $AP_{cd}C$ for USMS is 90.21% in normal test case

Journal of Theoretical and Applied Information Technology
20th October 2013. Vol. 56 No.2

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Table. 5 Summary Of Average Percentage Of Different Coverage Criterion

Program	Normal Test Case Prioritization (%)					Proposed Test Case Prioritization (%)				
	APC_{cd}	APC_{fn}	APC_{pt}	APC_{br}	APC_{fl}	APC_{cd}	APC_{fn}	APC_{pt}	APC_{br}	APC_{fl}
USMS	90.21	88.23	72.25	85.76	92.08	80.78	85.75	76.12	84.36	93.27
HMS	93.45	90.45	79.78	78.81	89.65	92.15	88.56	80.99	77.45	86.00
IPOS	82.11	79.45	77.25	80.56	83.89	78.45	76.78	75.23	78.56	85.22



a) USMS b) HMS c) IPOS

Figure 4 Overall Comparative Analysis.

prioritization method and 80.78% for the proposed method. The proposed method performs 9.43% lesser than the normal test case prioritization method. APC_{fn} for USMS is 88.23% in normal test case prioritization method and 85.75% for the proposed method. The proposed method performs 2.48% lesser than the normal test case prioritization method. APC_{pt} for USMS is 72.25% in normal test case prioritization method and 76.12% for the proposed method. The proposed method yields 3.87% improved performs than the normal test case prioritization method. APC_{br} for USMS is 85.76% in normal test case prioritization method and 84.36% for the proposed method. The proposed method performs 1.4% lesser than the normal test case prioritization method. APC_{fl} for USMS is 92.08% in normal test case prioritization method and 93.27% for the proposed method. The proposed method yields 1.19% improved performance. For path coverage and fault coverage the proposed method gives improved performance. Similarly the empirical study for HMS and IPOS is given in Table 5. The graphical representation of the comparative analysis is given in Fig. 4.

At the outset of this study, the proposed weighted method based test case prioritization covers all the coverage criteria in single test case order. Therefore the testers, not necessary to execute the test cases more than one time for different coverage criteria during regression testing.

5. CONCLUSION

In this paper a weighted method for test case prioritization which covers more than one coverage criteria is proposed. The main target of this method is to increase coverage criteria by executing single prioritized test cases. To attain this goal the test cases are prioritized based on calculated weight for statement coverage, function coverage, path coverage, branch coverage and fault coverage. The effectiveness of the proposed method is highlighted by comparing the 2-optimal algorithm with the proposed new weighted method for test case prioritization. This shows that the weighted method for test case prioritization is more effective than the 2-optimal test case prioritization algorithm. The limitation of the proposed test case prioritization method that is yields better performance for some coverage criteria. Future work is carried out to yield improved performance for all the coverage criteria by considering technological factors like developer-perceived code complexity and fault impact.

REFERENCES

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

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[2] Srinivasan Desikan,Gopalaswamy Ramesh, "Software Testing Principles and Practices", Pearson Education, 1st Edition, 2006, pp. 43-52.

[3] Hema Srikanth and Sean Banerjee, "Improving Test Efficiency Through System Test Prioritization", *The Journal of Systems and Software*, Elsevier, Vol. 85, No. 5, 2012, pp. 1176-1187

[4] Hema Srikanth, Laurie Williams, and Jason Osborne, "System Test Case Prioritization of New and Regression Test Cases", *IEEE International Symposium on Empirical Software*, 2005, pp. 64-73.

[5] G. Rothermel, R. Untch, C. Chu, and M.J. Harrold, "Test Case Prioritization: An Empirical Study" *In proceedings of International Conference on Software Maintenance*. 1999, pp. 179-188.

[6] G. Rothermel, R. Untch, C. Chu, and M.J. Harrold, "Prioritizing Test Cases for Regression Testing", *IEEE Transaction on Software Engineering*, Vol. 27, No. 10, 2001, pp. 929-948.

[7] Fevzi Belli and Christof J. Budnik, "Towards Minimization of Test Sets for Coverage Testing of Interactive Systems", *In Proceedings of the 18th international conference on Innovations in Applied Artificial Intelligence*, 2005, pp. 300-309.

[8] James F. Peters and Witold Pedrycs, "Software Engineering – An Engineering Approach", *John Wiley & Sons Inc.* 2000, pp. 461-462.

[9] Hyunsook Do, Siavash Mirarab, Ladan Tahvildari, and Gregg Rothermel, "The Effects of Time Constraints on Test Case Prioritization: A Series of Controlled Experiments", *IEEE Trans. on Software Engineering*, Vol. 36, No. 54, 2010, pp. 593-617

[10] S. Elbaum , Gregg Rothermel, Satya Kanduri, and Alexey G. Malishevsky, "Selecting a Cost-Effective Test Case Prioritization Technique", *Software Quality Journal*, Vol. 12, No. 3, 2004, pp. 185-210.

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

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

[13] Sihan Li, Naiwen Bian, Zhenyu Chen, Dongjiang You, Yuchen He, "A Simulation Study on Some Search Algorithms for Regression Test Case Prioritization" *In proceedings of IEEE International Conference on Software Quality*, 2010, pp. 72-81.

[14] S. Yoo, Harman M, "Regression testing minimization, selection and prioritization: a survey", *Journal of Software: Testing, Verification and Reliability*, 2012, pp. 67-120.

[15] Bo Jiang, Zhenyu Zhang, W. K. Chan, T. H. Tse, "Adaptive Random Test Case Prioritization", *In Proceedings of IEEE/ ACM International Conference on Automated Software Engineering*, 2009, pp. 233 – 244

[16] Krishnamoorthi R and Sahaaya Arul Mary S.A, "Factor Oriented Requirement Coverage Based System Test Case Prioritization of New and Regression Test Cases", *International Journal of Information and Software Technology*, Vol. 51, No. 4, 2009, pp. 799-808.

[17] Bogdan Korel, George Koutsogiannakis, Luay H. Tahat, "Application of System Models in Regression Test Suite Prioritization", *In Proceedings of IEEE International Conference on Software Maintenance*, 2008, pp. 247-256.

[18] M. Salehie, Sen Li, Tahvildari L, Dara R, Shimin Li, and Moore M, "Prioritizing Requirements-Based Regression Test Cases: A Goal-Driven Practice", *In proceedings of IEEE European Conference on Software Maintenance and Reengineering*, 2011, pp. 329 – 332

[19] Yu-Chi Huang, Chin-Yu Huang, Jun-Ru Chang and Tsan-Yuan Chen, "Design and Analysis of Cost-Cognizant Test Case Prioritization Using Genetic Algorithm with Test History", *In proceedings of IEEE 34th Annual Conference on Computer Software and Applications*, 2010, pp. 413- 418

[20] Fevzi Belli and Christof J. Budnik, "Towards Minimization of Test Sets for Coverage Testing of Interactive Systems", *In Proceedings of the 18th international conference on Innovations in Applied Artificial Intelligence*, 2005, pp. 300-309.