TEST CASE PRIORITIZATION

Praveen Ranjan Srivastava

Computer Science and Information System Group, BITS Pilani, India-333031

E-mail: praveenrsrivastava@gmail.com

ABSTRACT

Test case prioritization techniques involve scheduling over test cases in an order that improves the performance of regression testing. It is inefficient to re execute every test cases for every program function if once change occurs. Test case prioritization techniques organize the test cases in a test suite by ordering such that the most beneficial are executed first thus allowing for an increase in the effectiveness of testing. One of the performance goals i.e. the fault detection rate, is a measure of how quickly faults are detected during the testing process. In this paper I present a new test case prioritization algorithm, which calculates average faults found per minute. I present the results illustrating the effectiveness of algorithm with the help of APFD metric. The main aim of my paper is to determine the effectiveness of prioritized and non-prioritized case with the help of APFD.

Keywords: *Regression Testing, Average Percentage of Faults Detected (APFD), Test Cases.*

1. INTRODUCTION

Regression testing is the re-execution of some subset of test that has already been conducted. In regression testing as integration testing proceeds, number of regression tests increases and it is impractical and inefficient to re execute every test for every program function if once change occurs. It is an expensive testing process used to detect regression faults. Regression test suites are often simply test that software engineers have previously developed, and that have been saved so that they can be used later to perform regression testing [1,2, 3]. So regression testing can be defined as follows: Let P be a program and P' be a modified version of P and T be a test suite developed for P. Regression testing is concerned with validating P'. Regression test selection techniques attempt to reduce the cost of regression testing by selecting and running only a subset of the test cases in an existing test suite

In the previous work an Average Percentage of Faults Detected (APFD) metric [1] was used to determine the effectiveness of the new test case orderings, but it considered faults and test cases cost to be uniform.

2 Problem Statement

Rothermel at el. [2, 7] defines the test case prioritization problem as follows:

Given: T, a test suite; PT, the set of permutations of T; f, a function from PT to the real numbers.

Problem: Find T' belongs to PT such that (for all T'') (T'' belongs to PT) (T'' $\neq$ T') [f (T') $\geq$ f (T'')].

Here, PT represents the set of all possible prioritizations (orderings) of T and f is a function that, applied to any such ordering, yields an award value for that ordering [2,7].

The objective of this research is to develop a test case prioritization technique that prioritizes test cases on the basis of detection of fault rate.

3 METHODOLOGIES

This section provides the methodologies that are related to regression testing. There are four methodologies that are available for regression testing. These methods are [2,5, 8]
2.1 Retest all
2.2 Regression Test Selection
2.3 Test Suite Reduction
2.4 Test Case Prioritization

3.1 Retest –all.

In this technique the test cases that no longer apply to modified version of program are discarded and all the remaining set of test cases are used to test the modified program.

3.2 Regression test selection.
Retest all technique takes time and effort as all test cases are used to test the program again, so may be quite expensive. This technique much better as it uses information about program, modified program, test cases to select subset of test cases for testing.

3.3 Test suite Reduction.
This technique uses information about program and test suite to remove the test cases, which have become redundant with time, as new functionality is added. It is different from Regression test selection as former does not permanently remove test cases but selects those that are required.
Advantage of this technique is that it reduces cost of validating, executing, managing test suites over future releases of software, but the downside of this is that it might reduce the fault detection capability with the reduction of test suite size

3.4 Test Case Prioritization.
In this technique each test cases are assigned a priority. Priority is set according to some criterion and test cases with highest priority are scheduled first For example criterion may be that the test case which has faster code coverage gets the highest priority. Advantage to previous techniques is that it doesn't discard or permanently remove the test cases from test suite. Another criterion may be rate at which fault is detected.

4 DETERMINING TEST SUITE EFFECTIVENESS.

The performance of the prioritization technique used in this paper, it is necessary to assess effectiveness of the ordering of the test suite. Effectiveness will be measured by the rate of faults detected. The following metric is used to calculate the level of effectiveness.

4.1 AVERAGE PERCENTAGE OF FAULTS DETECTED (APFD) METRIC

To quantify the goal of increasing a subset of the test suite's rate of fault detection, i use a metric called APFD developed by Elbaum et al. [1,2,4] that measures the average rate of fault detection per percentage of test suite execution. The APFD is calculated by taking the weighted average of the number of faults detected during the run of the test suite. APFD can be calculated using a notation:
Let T -> The test suite under evaluation

m -> the number of faults contained in the program under test P

n -> The total number of test cases and

TFi -> The position of the first test in T that exposes fault i.

$$APFD = 1 - \frac{TF_1 + TF_2 + \ldots\ldots + TF_m}{nm} + \frac{1}{2n}$$

So as the formula for APFD shows that calculating APFD is only possible when prior knowledge of faults is available. APFD calculations therefore are only used for evaluation.

5. PROPOSED WORK: A NEW PRIORITIZATION TECHNIQUE.

5.1 Introduction

Earlier work [1,2,4] may take long time (may be month or year) depending on the size of the test suite and how long each test case takes be run. However, through the use of an effective prioritization technique, testers can re order the test cases to obtain an increased rate of fault detection.
The technique presented in this paper implemented a new regression test suite prioritization algorithm that prioritizes the test cases with the goal of maximizing the number of faults that are likely to be found during the constrained execution.

5.2 THE ALGORITHM

```
Input: Test suite T, number of faults detected by a test case f, and cost to run each test case $T_{cost.}$
Output: Prioritized Test suite T'.
 1: begin
 2:    set T' empty
 3:    for each test case t ε T do
 4:    calculate average faults found per minute as
        $f/T_{cost}$
 5:    end for
 6:    sort T in descending order based on the on
       the value of each test case
 7:    let T' be T
 8:    end
```

With the assumption that the desired execution time to run the test cases is known in advance, one can trace the number of faults each test case find and in how much time it takes to find the faults. So using this information as input the algorithm prioritizes the test cases of particular test suite. The algorithm calculates the average number of faults found per minute by a test case and using this value sorts the tests cases in decreasing order of test suite.

6. EXPERIMENTATION AND ANALYSIS

Below table shows the number of faults detected by a test case in the test suite and total time taken by each test case.

	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10
F1	*					*				
F2				*			*	*	*	
F3		*			*	*				*
F4							*			
F5		*						*	*	
F6				*						
F7				*	*					
F8		*	*							
F9						*				
F1 0	*									*
No. of fau lt	2	3	1	3	2	3	2	2	2	2
Ti me	5	7	11	4	10	12	6	15	8	9

APFD Result

From proposed algorithm (5.2):

$V_{Ti=}$fault/time(rate of fault detection)
The calculations are:

V_{T1}=2/5=0.4 V_{T2}=3/7=0.42,
V_{T3}=1/11=0.09 V_{T4}=3/4=0.75
V_{T5}=2/10=0.2 V_{T6}=3/12=0.25
V_{T7}=2/6=0.33 V_{T8}=2/15=0.133
V_{T9}=2/8=0.25 V_{T10}=2/9=0.22

Priority set according to decreasing order of value of $V_{Ti,}$ since more the rate of fault detection more will be the priority.
Hence the prioritized order is:
$T_4, T_2, T_1, T_7, T_6, T_9, T_{10}, T_5, T_8, T_3$
In the above table
m=no. of faults = 10
n=no. of test cases = 10
So putting the values of m , n ,TF_i(The position of the first test in T that exposes fault i) in the equation

$$\text{APFD} = 1 - \frac{TF_1 + TF_2 + \dots\dots + TF_m}{nm} + \frac{1}{2n}$$

Putting values:

$$\text{APFD} = 1 - \frac{3+1+2+4+2+1+1+2+5+3}{10*10} + \frac{1}{20}$$

$$= \mathbf{0.81}$$

APFD value for non-prioritized test case:

$$\text{APFD} = 1 - \frac{1+4+2+7+2+4+5+3+6+1}{10*10} + \frac{1}{20}$$

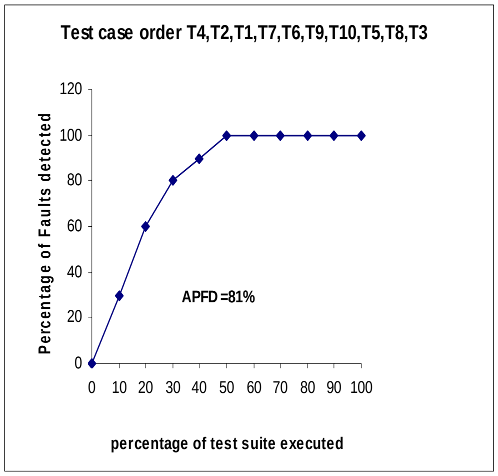
$$= \mathbf{0.70}$$

6.1Analysis of APFD

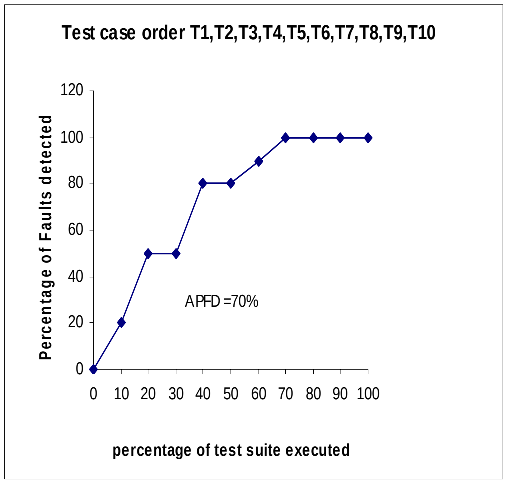
The comparison is drawn between prioritized and non-prioritized case, which shows that value obtained for prioritized case (new approach) is more than previous method, hence more effective of prioritized case

Below two graphs showing the Results for prioritized and non-prioritized case

APFD graph for prioritized test suite



APFD graph for non-prioritized test suite



7. DISCUSSION AND CONCLUSION

This paper proposed an algorithm for test case prioritization in order to improve regression testing. Analysis is done for prioritized and non-prioritized cases with the help of APFD (average percentage fault detection) metric. Graphs prove that prioritized case is more effective. In future I will try on test case prioritization over requirement analysis using APFD and risk metrics.

7. REFRENCES

[1] Alexey G. Malishevsky, Joseph R. Ruthruff, Gregg Rothermel, Sebastian Elbaum, Cost-cognizant Test Case Prioritization, 2006

[2] S. Elbaum, A. Malishevsky, and G.Rothermel Test case prioritization: A family of empirical studies. IEEE Transactions on Software Engineering, February 2002.

[3] Roger S. Pressman, Software engineering a practitioner's approach 6/e, 2005

[4] Sebastian Elbaum, Gregg Rothermel, Satya Kanduri, Alexey G. Malishevsky, Selecting a Cost-Effective Test Case Prioritization Technique, April 20, 2004

[5] Aditya P.Mathur, Foundation of software testing, Pearson Education 1st edition.

[6] Maruan Khoury, Cost-Effective Regression Testing, 2006

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

[8] Maruan Khoury, Cost-Effective Regression Testing, 2006