www.ijecs.in
International Journal Of Engineering And Computer Science ISSN:2319-7242
Volume 3 Issue 4 April, 2014 Page No. 5304-5306

Clustering Approach To Test Case Prioritization Using Code Coverage Metric

Medhun Hashini D.R[1]
[1]Assistant Professor
Department of Computer Applications (UG)
VLB Janakiammal College of Arts and Science
Coimbatore, India.
medhun.hashini@gmail.com

Abstract

Software testing is important phase of software development life cycle which ensures the developer that the developed software works according to specifications or not. Prioritization techniques that incorporate a clustering approach and utilize code coverage, code complexity and history faults as well to increase the effectiveness of the prioritization. To make testing efficient and effective a techniques of test case prioritization are used. An efficient Test case prioritization technique reduces the cost of testing and fault detection capabilities of testing. Results show that test case prioritization that utilizes a clustering approach can improve the effectiveness of test case prioritization techniques.

Keyword: Clustering Approach, Regression Testing, Test Case Prioritization

1. Introduction:

Software testing is a process done with an intention to find out the defect in existing software. It is also the process of evaluation a software item to detect differences between given input and expected output also to assess the feature of a software item. Testing assesses the quality of the product. Software testing is a process that should be done during the development process. In other words software testing is verification and validation process of computer program or application/product to meet the requirements that guided its design and development, works as expected, can be implemented with the same characteristics, and satisfies the needs of stakeholders.

2. Test Case Prioritization:

Test case prioritization is a mechanism needed for arranging a test case in appropriate order to increase their effectiveness at meeting some performance goal and rate of fault detection. Test case prioritization is a method to prioritize and schedule test cases in appropriate order. To run test cases of higher priority before than the lower priority test case in order to minimize time, cost and effort during software testing phase. various performance goals are like rate of fault detection which a is a measure of how quickly the fault is detected so that during testing faster feedback can provide about system under testing and allow the software tester to correct the software at earlier phase as possible. Regression testing is process of retesting the modified software and ensures that new error does not introduce into the previously tested source code due to these modifications. regression testing is very expensive testing process .in order to decrease the cost of regression testing the software tester may prioritize the test case so that the test case which are more important are run earlier during regression testing process.

3. Proposed Work

New Approach for Prioritization Using Clustering

3.1 Introduction

Earlier test cases are not prioritized and executed in any order and no merging of test cases is done. Test case clustering is done to club the test cases. The closest two test cases in terms of code coverage similarity are combined into a cluster first. Then the table of pair-wise similarities is recomputed with the new cluster being considered as a single document. Having created clusters, prioritization techniques to them. First, we reorder test cases within each cluster using specific software metrics. The first test case will be done according to the priority and rest will be ordered by using software metrics method. We considered four different test case prioritization techniques that usually use the three types of information written as below:

- Code Coverage
- Complexity Metric
- Fault History Information

3.2 Factors of Clustering (Prioritization)

The purpose is to define factors for testing based on which clustering can be done.

3.2.1 Code Coverage

This variable is also a numeral as it includes the calls to a routine, procedure, method, function or subprogram. It is sum of both built in functions as well as user defined functions.

No.of Function calls (Fc) =
Local functions + nested functions +
private functions + overloaded methods

After the test suite is executed first time for the software then it yields a result that when it is non – prioritized. Now as we will be able to keep an account for the re-test that is regression test that what is the statement coverage for each test case and how many function calls are occurring in the particular statement coverage for the particular test case. As described above we will be calculating the product of the statement coverage and the no. of function calls. This becomes the software metric or the basis of the ordered set of test cases. As each test case has a value now so they will be ordered as the one with greatest value as the highest prioritized test case and followed in descending order. For this an algorithm is proposed with the assumption that the statement coverage and no. of function calls are known, such that this is given as the input to the algorithm and the output is the prioritized (in some order (either ascending or descending)) test suite.

3.3. The Algorithm

Now an algorithm for test case prioritization which is framed in a set pattern but with few changes in metric is done. The algorithm calculates the Product P metric for every test case given that the statement coverage St and Number of function calls Fc is known for every test case in advance. Sorting algorithms like quick sort merge sort or heap sort can be used to sort the values in descending order.

Definition 1: The Test Case Prioritization Problem with clustering

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

Problem: Find T′ ∈ PT such that (∀T″) (T″ ∈ PT) (T″ != T′) [f(T′) ≥ f(T″)]

Prioritized test suite T’

```
begin
set T’ empty
for each test case t € T do
Clusteri =i=1k Ti
calculate Product P as St * Fc
end for
sort T in descending order based on the value of P for each test case
let T’ be T
end
```

Now the test cases are arranged in descending order according to the value of P. But there can be a possibility that the product for the two or more test cases comes out to be same.

4. Analysis

Here’s the example that make more clear that, how clustering is helpful for testing.

Test Cases	Values(Stmt Covered)				
T1	45				
T2	56				
T3	21				
T4	78				
T5	34				
T6	17				
T7	22				
T8	72				
T9	2				
T10	10				
T11	3				
T12	24				
T13	45				
T14	77				
T15	31				
Clusters	**C1**	**C2**	**C3**	**C4**	**C5**
Test Cases	T1 T2 T3	T4 T5	T6 T7 T8 T9	T10 T11 T12 T13	T14 T15

Code coverage is calculated for each test case that is executed in a test suite. This usually based on just the count of the number of statements traversed by the particular cluster. Mathematically, it is calculated as the number of statements covered upon the total number of clusters.

Statement coverage (St) = (No. of statements covered / Total no. of statements) * 100

<u>Non-prioritized</u>

C1=T1+T2+T3	(45+56+21)/3	=40.67
C2=T4+T5	(78+34)/2	=56
C3=T6+T7+T8+T9	(17+22+72+2)/4	=28.25
C4=T10+T11+T12+T13	(10+3+24+45)/4	=20.5
C5=T14+T15	(77+31)/2	=54

(C1+C2+C3+C4+C5)/5
(40.67+56+28.25+20.5+54)/5 =39.88
APFD= 39.88/100 = 0.3988
Formula Σ =1-0.3988 =**0.60**

<u>Prioritized</u>

C1=T1+T2+T3	(2+3+10)/3	=5
C2=T4+T5	(17+21)/2	=19
C3=T6+T7+T8+T9	(22+244+31+34)/4=27.25	

C4=T10+T11+T12+T13 (45+45+56+72)/4=54.5
C5=T14+T15 (77+78)/2 =77.5
(C1+C2+C3+C4+C5)/5
(5+19+27.25+54.5+77.5)/5 =36.75
APFD= 36.75/100 = 0.3675
Formula Σ =1-0.3675 =**0.63**

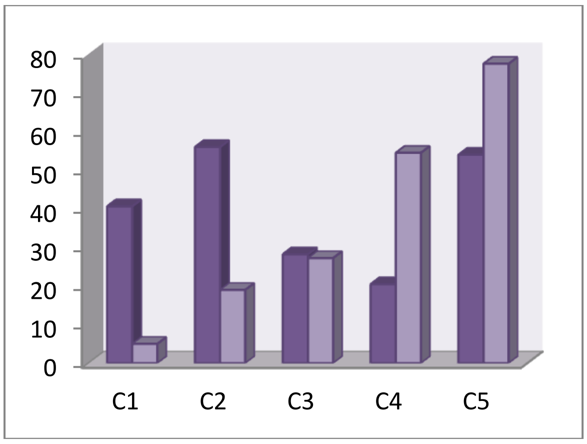

Fig1: Test Cases under Non Prioritized Cluster

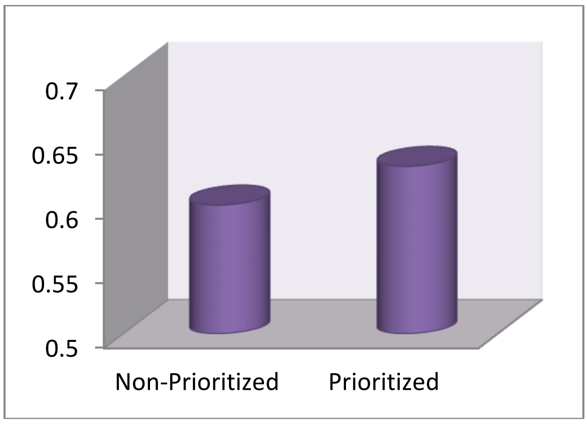

Fig2: Code Coverage under Prioritized and Non Prioritized Test Cluster

Here comparison among the results of prioritized and non-prioritized suite is done based on the results of the APFD metric. This is average percentage of faults detected. APFD is a standardized metric that is used to find the degree of faults detected.

Thus the prioritized test cases yield better fault detection than the non – prioritized test cases. The above figure shows the difference between non-prioritized and prioritized. This will clearly examine the difference in code coverage as well as in functionality.

5. Conclusion

This paper proposed such approach for test case prioritization using clustering in order to improve or to increase the efficiency in code coverage. This will make the task easy. Analysis is done in clustering prioritized and non-prioritized cases with the help of APFD (average percentage fault detection). Prioritized cases always gives the best result rather than non-prioritized. In future test case prioritization with clustering should be done in such way that will increase more efficiency in time as well.

References

1. Gregory M. Kapfhammer Software Testing
2. R. Pressman (2009), Software Engineering: A Practitioner's Approach. Boston: *McGraw Hill*.
3. Alexey G. Malishevsky, Joseph R. Ruthruff, Gregg Rothermel, Sebastian Elbaum, Costcognizant (2006) Test Case Prioritization
4. G. Rothermel, R. Untch, C. Chu, and M. J. Harrold(2001), Prioritizing test cases for regression testing, *IEEE Transactions on Software Engineering*, vol. 27, no. 10, pp. 929–948.
5. W. E. Wong, J. R. Horgan, S. London, and H. Agrawal(1997), A study of effective regression testing in practice, in *Proceedings of the International Symposium on software Reliability Engineering*.pp. 230-238
6. S. Yoo, M. Harman, P. Tonella, and A. Susi (2010), Clustering test cases to achieve effective and scalable prioritisation incorporating expert knowledge, in Proceedings of the *International Conference on Software Testing and Analysis*.
7. P. Tan, M. Steinbach, and V. Kumar(2006), Introduction to Data Mining. *Addison-Wesley*.
8. S. Elbaum, A. G. Malishevsky, and G. Rothermel(2002), Test case prioritization: A family of empirical studies, *IEEE Transactions on Software Engineering*, vol. 28, no. 2, pp. 159–182
9. S. Elbaum, A. G. Malishevsky, and G. Rothermel(2002), Test case prioritization: A family of empirical studies, *IEEE Transactions on Software Engineering*, vol. 28, no. 2, pp. 159–182
10. Srivastava and J. Thiagarajan(2002), Effectively prioritizing tests in development environment, in *Proceedings of the International Symposium on Software Testing and Analysis.*
11. Malishevsky, G. Rothermel, and S. Elbaum(2002), Modelig the costbenefits tradeoffs for regression testing techniques, in *Conf. Softw. Maint* pp. 204-213.