Regression Testing Based on Comparing Fault Detection by multi criteria before prioritization and after prioritization

KanwalpreetKaur[#], Satwinder Singh[*]

#Research Scholar, Dept of Computer Science and Engineering, College, Fatehgarh Sahib.

*Asst Prof, Dept of Computer Science and Engineering, College, Fatehgarh Sahib.

Abstract— **Test case prioritization techniques involve scheduling test cases for regression testing in an order that increases their effectiveness at meeting some performance goal. This is inefficient to re execute all the test cases in regression testing following the software modifications. Using information obtained from previous test case execution, prioritization techniques order the test cases for regression testing so that most beneficial are executed first thus allows an improved effectiveness of testing. This paper presents the new metric for assessing rate of fault detection and an algorithm to prioritize test cases using multi criteria technique. Using the new technique the effectiveness of this prioritization is shown Comparing it with non-prioritized test case. Analysis proves that multi criteria based Prioritized test cases are more effective in detecting more faults.**

Index Terms—**Test Case Prioritization, Regression Testing, Software Testing, Multi criteria for Testing , Fault Detection.**

I. INTRODUCTION

Software Testing means computing the system with purpose of finding errors. It is an application for a concerted action of a system under controlled conditions and evaluating the results. Once system has been developed, it must be tested before it implementation. It is oriented towards Error-detection. Software testing is one element of a broader topic that that is often referred to as verifying and validating that a software application or program. Software testing is useful for finding the defects, fundamental weakness in the application code that must be improved or checked. Software testing has three main purposes: verification, validation, and defect:

• The process of verification confirms that software meets its specifications. It ensures that software correctly implemented for specific function.

• Whereas the process of validation ensures that the software meets the business requirements.

It provides the traceable activities to customers.

• A defect is inconsistency among the expected and actual result. The defect's ultimate source may be traced to a fault introduced in the specification, design, or development phases.

In development of software system, cost of testing a program is associated [1]. Tester has to write test plan and test cases, to set up the proper equipment, systematically execute the test cases, and follow up on problems that are identified also try to remove most of the faults. For faults that are not discovered and removed before the software has been shipped, there are costs. Some of these costs are monetary, and some could be significant in less tangible ways. Customers can lose faith in our business and can get very angry. They can also lose a great deal of money if their system goes down because of our defects. And, software development organizations have to spend a great deal of money to obtain specific information about customer problems and to find and fix the cause of their failures.

To minimize the costs associated with testing and with software failures, a goal of testing must be to uncover as many defects as possible with as little testing as possible. In other words, we want to write test cases that have a high likelihood of uncovering the faults that are the most likely to be observed as a failure in normal use. It is simply impossible to test every possible input-output combination of the system; there are simply too many permutations and combinations. As testers, we need to consider the economics of testing and strive to write test cases that will uncover as many faults in as few test cases .

II. REGRESSION TESTING

Regression Testing : Regression Testing is an important strategy for reducing side effects. We run regression testing every time software experiences a change in form of bug fixes or some additional functionality. It is done to ensure that code had not an adverse effect to the other module or any existing functions and it may not have produced any defect. The regression test suite contains three different classes of test cases:

• A representative sample of tests that will exercise all software functions.

• Additional test that focuses on software function that are likely to be affected by change.

• Test that focus on components that have been changed.

A subset of the regression test cases can be set aside as

Smoke tests. A smoke test is a group of test cases that establish that the system is stable and all major functionality is present and works under "normal" conditions. Smoke tests are often automated, and the selections of the test cases are broad in scope. The smoke tests might be run before deciding to proceed with further testing (why dedicate resources to testing if the system is very unstable). The purpose of smoke tests is to demonstrate stability, not to find bugs with the system.The most crucial phase in the software development life cycle is maintenance phase, in which the development team is supposed to maintain the software which is delivered to the clients by them. Software maintenance results for the reasons like error corrections, enhancement of capabilities, deletion of capabilities and optimization.

Regression testing is defined as "The process of retesting the modified parts of the software and ensuring that no new errors have been introduced into previously tested code".The various types of techniques for regression testing are:

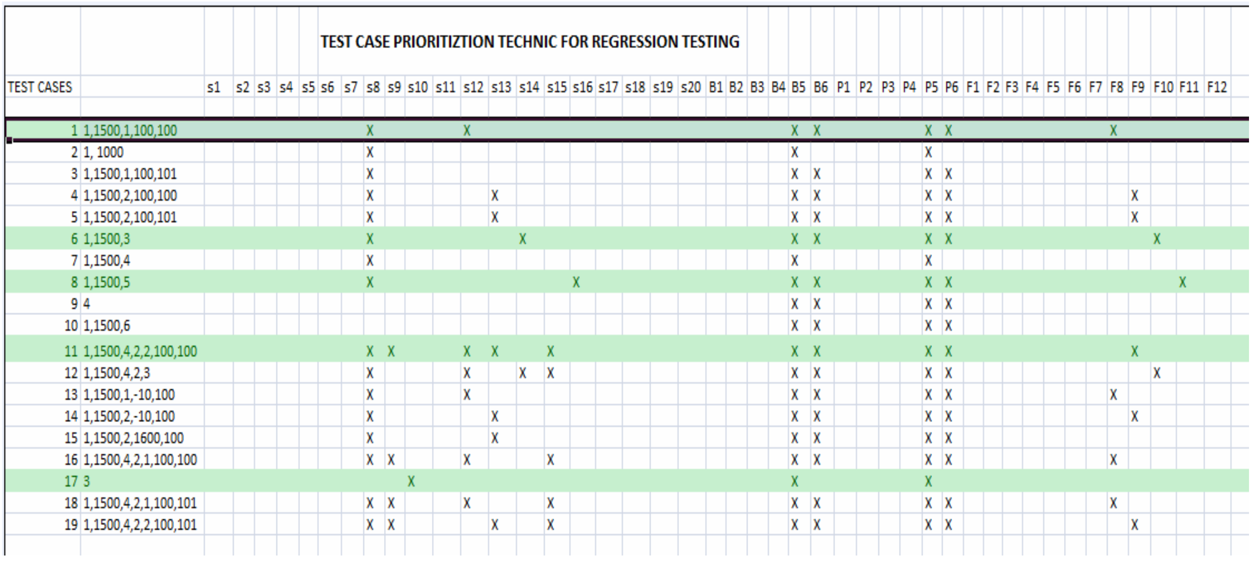
• Retest all: method is one of the conventional methods for regression testing in which all the tests in the existing test suite are re-runned. So the retest all technique is very expensive as compared to techniques which will be discussed further as regression test suites are costly to execute in full as it require more time and budget.

• Regression test selection: approaches attempt to reduce the cost of regression testing by selecting some appropriate subset of the existing test suite .Test selection techniques normally use the source code of a program to determine which tests should be executed during the regression testing stage .

• Hybrid approach: also known as regression test distribution is another alternative that can make regression testing more practical by more fully utilizing the computing resources that are normally available to a testing team.

III. Proposed Structural Work

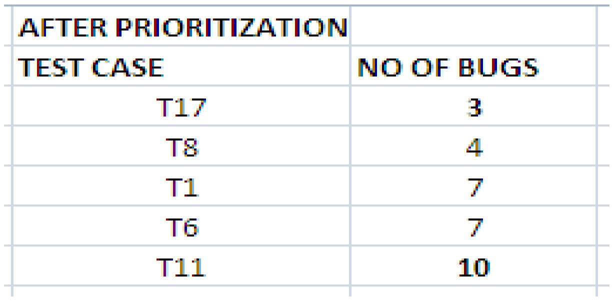
In our Research we focused on improving this model description by adding more criteria for selection of test cases for testing of the software in regression testing. Various testing have been fetched on industrial experience. Application based on JAVA language and C++ has been implemented according to the proposed optimized proposed work for software testing. We have tried to introduce more testing dependencies. In regression testing, we have added code coverage, branch coverage, reusability coverage, path coverage and fault coverage dependencies test which is helpful in finding issues in overall testing phase. Software testing has been done by various tools and we have linked to proposed theory. Test model selection and test volume evaluation method has been applied to the software testing work of Industrial applications and has been compared with traditional method.



								TEST CASE PRIORITIZTION TECHNIC FOR REGRESSION TESTING																																					
TEST CASES		s1	s2	s3	s4	s5	s6	s7	s8	s9	s10	s11	s12	s13	s14	s15	s16	s17	s18	s19	s20	B1	B2	B3	B4	B5	B6	P1	P2	P3	P4	P5	P6	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12
1	1,1500,1,100,100								X				X													X	X					X	X								X				
2	1, 1000								X																	X						X													
3	1,1500,1,100,101								X																	X	X					X	X												
4	1,1500,2,100,100								X					X												X	X					X	X									X			
5	1,1500,2,100,101								X					X												X	X					X	X									X			
6	1,1500,3								X						X											X	X					X	X										X		
7	1,1500,4								X																	X						X													
8	1,1500,5								X								X									X	X					X	X											X	
9	4																									X	X					X	X												
10	1,1500,6																									X	X					X	X												
11	1,1500,4,2,2,100,100								X	X			X	X		X										X	X					X	X									X			
12	1,1500,4,2,3								X				X		X	X										X	X					X	X										X		
13	1,1500,1,-10,100								X				X													X	X					X	X								X				
14	1,1500,2,-10,100								X					X												X	X					X	X									X			
15	1,1500,2,1600,100								X					X												X	X					X	X												
16	1,1500,4,2,1,100,100								X	X			X			X										X	X					X	X								X				
17	3										X															X						X													
18	1,1500,4,2,1,100,101								X	X			X			X										X	X					X	X								X				
19	1,1500,4,2,2,100,101								X	X				X		X										X	X					X	X									X			

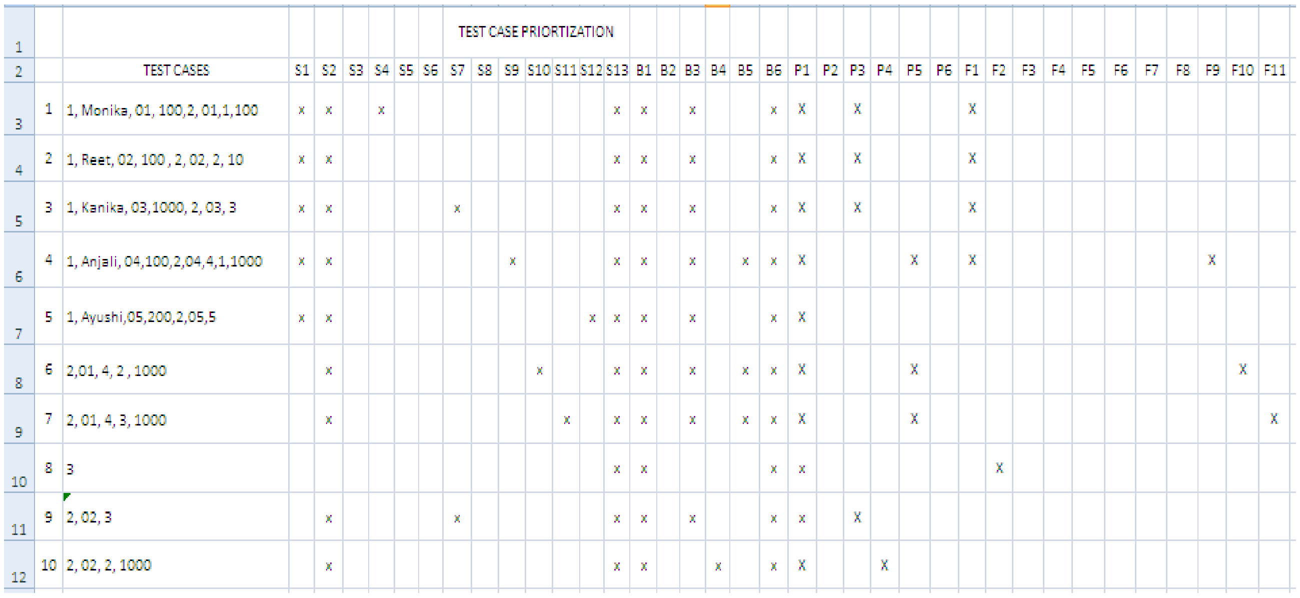
Fig 3.1Test Cases with its associated faults, statements, branches and paths coverage IN JAVA

Fig 3.1 shows nineteen test cases (T1to T19) and their associated fault coverage, branch coverage , statement coverage and path coverage have been merged into one table.On this mutilpe test case prioritization technique will be applied to get optimized test cases which will have maximum of fault detection.



AFTER PRIORITIZATION	
TEST CASE	NO OF BUGS
T17	3
T8	4
T1	7
T6	7
T11	10

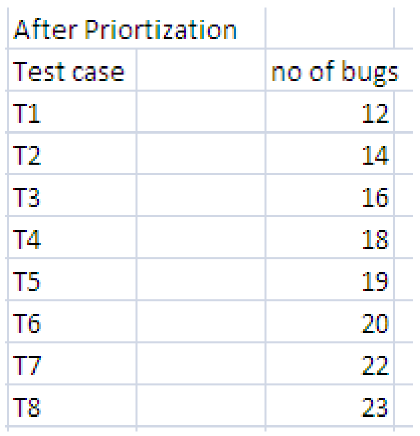
Fig 3.2 Optimized test cases after implementation of Multiple Criterion Based Prioritization Technique.



1									TEST CASE PRIORTIZATION																													
2		TEST CASES	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13	B1	B2	B3	B4	B5	B6	P1	P2	P3	P4	P5	P6	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11
3	1	1, Monika, 01, 100,2, 01,1,100	x	x		x									x	x		x			x	X		X				X										
4	2	1, Reet, 02, 100 , 2, 02, 2, 10	x	x											x	x		x			x	X		X				X										
5	3	1, Kanika, 03,1000, 2, 03, 3	x	x					x						x	x		x			x	X		X				X										
6	4	1, Anjali, 04,100,2,04,4,1,1000	x	x							x				x	x		x		x	x	X				X		X								X		
7	5	1, Ayushi,05,200,2,05,5	x	x										x	x	x		x			x	X																
8	6	2,01, 4, 2 , 1000		x								x			x	x		x		x	x	X				X											X	
9	7	2, 01, 4, 3, 1000		x									x		x	x		x		x	x	X				X												X
10	8	3													x	x					x	x							X									
11	9	2, 02, 3		x					x						x	x		x			x	x		X														
12	10	2, 02, 2, 1000		x											x	x			x		x	X			X													

Fig 3.3Test Cases with its associated Faults, Statements, Branches and Paths Coverage IN C++

Fig 3.3 shows nineteen test cases (T1to T12) and their associated fault coverage, branch coverage , statement coverage and path coverage have been merged into one table.On this mutilpe test case prioritization technique will be applied to get optimized test cases which will have maximum of fault detection.



After Priortization		
Test case		no of bugs
T1		12
T2		14
T3		16
T4		18
T5		19
T6		20
T7		22
T8		23

Fig 3.4 Optimized test cases obtained after Multiple Criterion Based Prioritization Technique in C++

IV. EXPERIMENTATION AND RESULTS

The primary concern of the software testing process is to save testing resources and to find maximum output in form of bugs finding from limited resources. For providing optimized solution for the same we have done changing in regression testing process by introduction of multiple criteria's for testing. Basically we did here in our research ad hoc regression testing, testing in which test cases are made only if any bug found in application. Testing is done in two programming languages for checking compatibility and test case prioritization is consider in both of the languages for same application.

The figure shown below shows the comparison between percentages of fault detection found before prioritization and after prioritization

PERCENTAGE OF FAULT DETECTED IN JAVA		
NO. OF TEST CASES	*NO OF FAULTS*	*PERCENTAGE OF FAULT DETECTION*
19	*120*	*6.32%*
5	*120*	*24%*

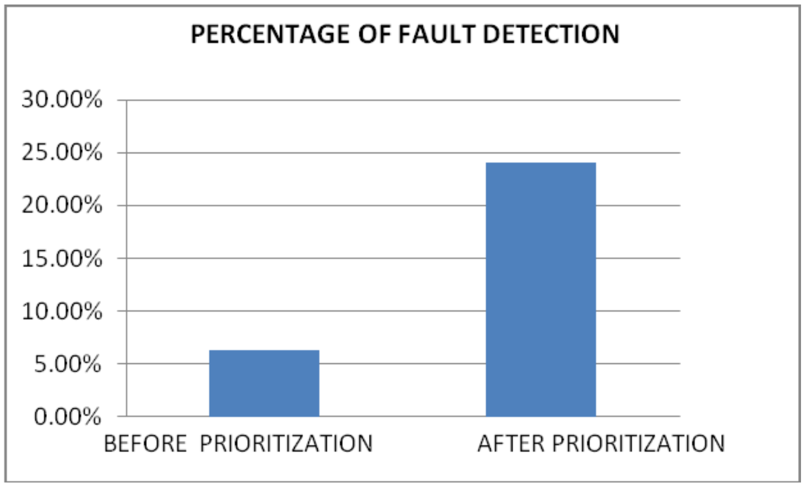


Fig 4.1 fault detection found before prioritization and after prioritization

The figure shown below shows the comparison between percentages of fault detection found before prioritization and after prioritization in C++

PERCENTAGE OF FAULT DETECTED IN C++		
NO. OF TEST CASES	*NO OF FAULTS*	*PERCENTAGE OF FAULT DETECTION*
10	*89*	*8.90%*
8	*89*	*11%*

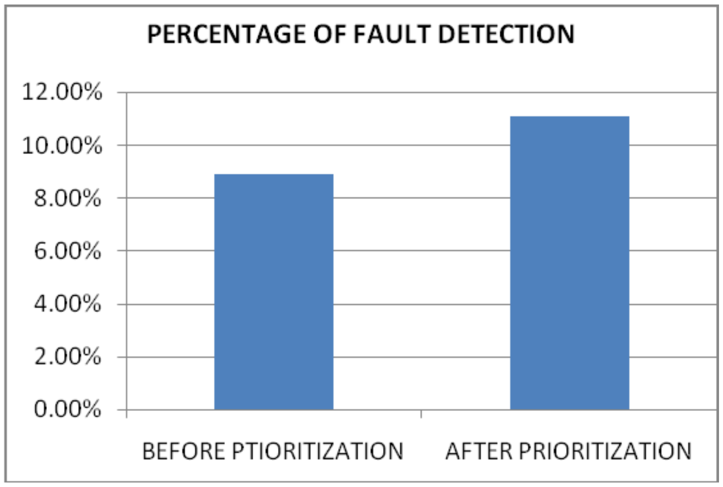


Fig 4.2 fault detection found before prioritization and after prioritization

V. CONCLUSION

Graphs prove that prioritized case is more effective. A prioritized test suite which covers more than one coverage criteria is considered to be a stronger coverage goal than a test suite which covers single coverage criteria.

The comparison of proposed work and related study [1] is done and performance of the proposed work is found better in term of fault detection in testing as shown in fig4.1 for java and fig4.2 for C++ above. Fault Detection is comparatively high after prioritization of test case by using multiple criteria based merging technique.

REFERENCES

[1] R. Savenkov, How to become a software tester. (Roman Savenkov Consulting, 2004)

[2] Alexey G. Malishevsky, Joseph R. Ruthruff, Gregg Rothermel, Sebastian Elbaum, "Cost-cognizant Test Case Prioritization," Technical Report TR-UNL-CSE-2006-004, Department of Computer Science and Engineering, University of Nebraska–Lincoln, Lincoln, Nebraska, U.S.A., March 2006.

[3] S. Elbaum, A. Malishevsky, and G. Rothermel, "Prioritizing test cases for regression testing," Proc. The 2000 ACM SIGSOFT International Symposium on Software Testing and Analysis, Portland, Oregon, U.S.A., August 2000, 102–112.

[4] S. Elbaum, A. Malishevsky, and G. Rothermel,"Test case prioritization: A family of empirical studies*,"* IEEE Transactions on Software Engineering, vol. 28(2), 2002, pp. 159–182.

[5] Huang,Chin-Yu, Lin,Chu-Ti, Software reliability analysis by considering fault dependency and debugging time lag. IEEE Transactions*,* vol. 55(3), 2006, pp. 436-450.

[6] D. Jeffrey and N. Gupta, "Test case prioritization using relevant slices," Proc. Computer Software and Applications Conference*,* 2006, 411–420.

[7] B. Qu, C. Nie, B. Xu, and X. Zhang, "Test case prioritization for black box testing," Proc. Computer Software and Applications Conference, July 2007, 465–474.

[8] B. Korel, G. Koutsogiannakis, and L. H. Tahat, "Model-based test prioritization heuristic methods and their evaluation," Proc. International Conference on Software Maintenance, 2007, 34–43.

[9] B. Korel, L. Tahat, and B. Vaysburg, "Model based regression test reduction using dependence analysis," Proc. International Conf. on Software Maintenance, 2002, 214–223.