ISSN: 2321-9653

INTERNATIONAL JOURNAL FOR RESEARCH IN APPLIED SCIENCE AND ENGINEERING TECHNOLOGY (IJRASET)

A Fault Analysis based Prioritization Specific Regression Test Path Identification Approach using ACO

Ms Sunita, Ms Mamta Gulia

School of Engg. &Science B.P.S.M.V Khanpur Kalan

Abstract— Testing is a process that never ends even after the software delivery. During the Acceptance testing if some changes are required in the software in terms of inclusion or exclusion of some module, In such case it is required to test the software again. But it is not feasible to test all cases. Now the work is to test only the required module instead of testing all. This whole concept is presented by Regression Testing. But while selecting the test module numbers of available factors are there. In this proposed work we will assign the prioritization to the test cases based on the number of faults in that module initially and the module critically level to the fault. According to this fault oriented module analysis the prioritization will be assigned to the test cases. An optimized approach is defined to perform for regression testing. The optimization is here performed to generate the regression test sequence using ACO approach. In this paper work an integrated model will used that will combine all these approaches and generate the optimal test sequence based on test case prioritization. The work will also design a hybrid approach for the generation of test case prioritization

Keywords—Regression Test Path, ACO (Ant Colony Optimization).

I. INTRODUCTION

Regression testing is selective retesting of a system or component to verify that modifications have not caused unintended effects and that the system or component still complies with its specified requirements. Regression tests are a subset of the original set of test cases. These test cases are re-run often, after any significant changes (bug fixes or enhancements) are made to the code. The purpose of running the regression test case is to make a "spot check" to examine whether the new code works properly and has not damaged any previously-working functionality by propagating unintended side effects. Most often, it is impractical to re-run all the test cases when changes are made. Since regression tests are run throughout the development cycle, there can be white box regression tests at the unit and integration levels and black box tests at the integration, function, system, and acceptance test levels.

II. GOOD TEST

1. The fault ratio must be reduced
2. No reputation of test cases
3. The complexity of the software system should be reduced.
4. Test cost should be minimum
5. Should be conducted by expert

III. SIGNIFICANCE OF WORK

The presented work is about to derive the optimal sequence of the test case generation under different prioritization approaches. The presented work will provide a cost effective sequence of the test case generation and the analysis. The work is also effective to analyse the software under different point of views so that the effectiveness of the code and test cases can be obtained.

The path testing is a very beneficial white box testing that play an important role to select the actual code test sequence as well as to perform the code optimization. The code optimization is actually performed in terms of detection of unusable code from the test sequence. There are number of aspects according to which the code sequence is decided. Better the test sequence is estimated, better the testing will be performed. It will also affect the testing code and test case reusability. The proposed work

is the implementation of approach that will work the basis of different priotization vectors and to generate the optimal sequence of test case generation.

IV. SOURCE OF DATA

As we know to perform the work related to regression testing we need some code as the case study to identify the tes sequence. Such kind of software model can be taken from cases studies used by different authors in previous results This kind of data will give two benefits. First we need not t write the input code or activity diagram and will save ou effort as well as the cost. The existing work will give th benefit of reliability. We can either take some dummy value to represent the test case or can use some case study used by earlier researcher.

V. METHODOLOGY

In this present work, an optimized approach is defined t perform for regression testing. The optimization is her performed to generate the regression test sequence using ACO approach. Here the work is defined to generate the test case under priority based analysis of available test case specification In this work, different approaches are considered to assign th cost to the test cases. The work also includes prioritizing the tes cases so that the optimal test sequence will be generated.

The work is defined in a layered model. At the initial stage, the software system is defined in terms of smaller modules. Once the modules are described, the identification of the test cases associated with particular module is done. A software module can be associated with one or more test case. Once the test cases are identified, the next work is to perform the test cost analysis using different approaches. One of such approach is to perform the estimation base of fault analysis. The software fault analysis is defined as the classification of the test case criticality as well as relative module criticality. A test case is defined with higher priority if it is representing some fatal error or the fault. If the test is defining some normal fault then the priority will be medium. If the test case is resolving some warning then the priority will be lowest. Once the priorities are assigned to these test cases, the cost estimation is also done based on priority analysis.

After assigning the priorities of these test cases, the next work is to generate the possible test sequence. This sequence pool work as the initial data set for ACO process on which the optimization is performed for generation of the sequence.

The work is here defined to generate the optimal test case sequence under different constraint analysis so that the regression testing cost will be reduced. Another method for assigning the test cost is here defined under user perspective analysis. The flow of the work is defined here in figure 1.

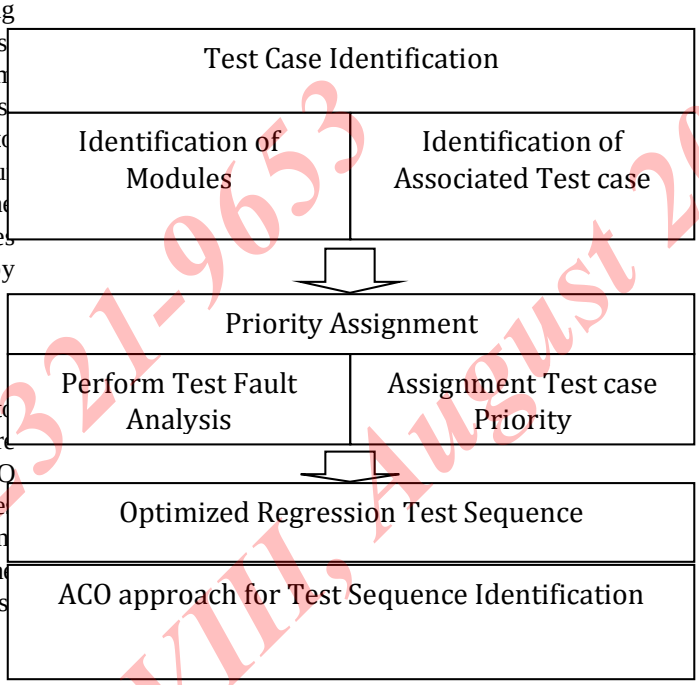

Figure:1 work flow

VI. ALGORITHM

The proposed work is about to find an optimal regression test sequence. To find the test sequence, at first the possible path analysis is performed using ACO approach. Once the identification of the possible sequence is defined, the optimization is performed. The work is here defined under cost and priority analysis. The cost analysis and priority analysis is performed under fault analysis.

Algorithm: Fault Based

1. Split the software program in N number of software

modules

2. Identify the test cases associated with each software modules.

3. Identify the faults over each module and based on it assign the software cost to these software modules

4. Based on the cost analysis prioritize the test cases.

5. Collect the possible test cases over the flow diagram and use it as the population set for ACO process.

6. Generate M Ants at random Position

7. For i=1 to MaxIterations

[Repeat steps 8 to 9]

8. Perform Ant Analysis to Generate analyse the Test cost

9. Build the Test Sequence under cost Analysis

10. Identify the test sequence for optimal sequence

11. If (Exist (Path))

{

12. Go to Step 7

}

13. Present new path as the Result Sequence

14. Exit

ALGORITHM: MODULE INTERACTION BASED

1. Split the software program in N number of software modules

2. Identify the test cases associated with each software modules.

3. Identify the module interaction analysis and based on it assign the software cost to these software modules

4. Based on the cost analysis prioritize the test cases.

5. Collect the possible test cases over the flow diagram and use it as the population set for ACO process.

6. Generate M Ants at random Position

7. For i=1 to MaxIterations

[Repeat steps 8 to 9]

8. Perform Ant Analysis to Generate analyze the Test cost

9. Build the Test Sequence under cost Analysis

10. Identify the test sequence for optimal sequence

11. If (Exist (Path))

{

12. Go to Step 7

}

13. Present new path as the Result Sequence

14. Exit

ISSN: 2321-9653

VII. ANALYSIS OF PROPOSED WORK

As the general case we have assigned the random cost to each test case and perform the analysis based on this random cost assignment. The output driven based on this assignment is shown as under.

a) The obtained Test Sequence of this random cost assignment is given as

3 9 4 1 10 7 8 5 6 2

b) The cost driven from the on given approach is given as

Cost = 11.0547

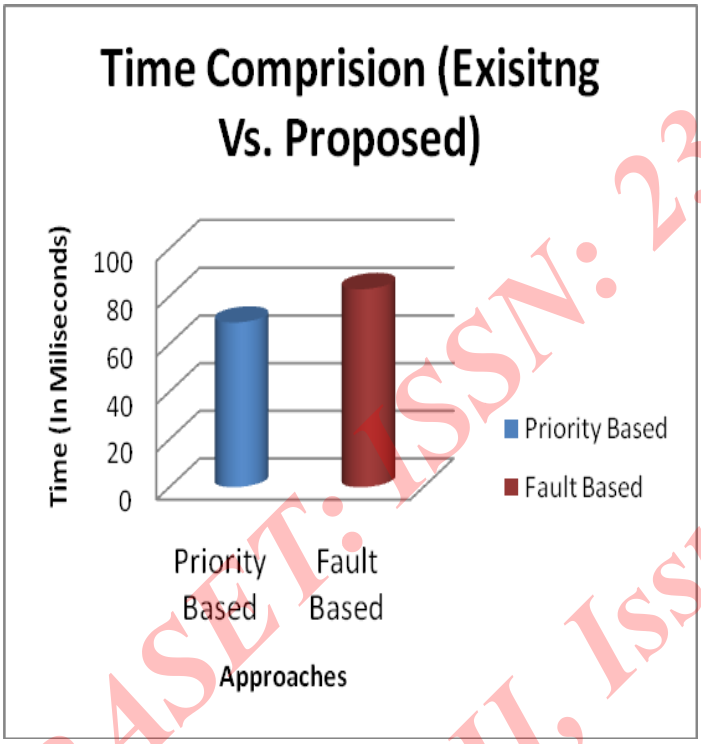

Fig:2 Cost Analysis (Proposed Vs. Existing)

Here figure is showing the time based comparative analysis between the priority based test path analysis and the cost based test path analysis. Here x axis represent these two approaches and y axis represents the time taken by the approach. The time is given in milliseconds. We can see that the fault based analysis is less efficient then priority based analysis.

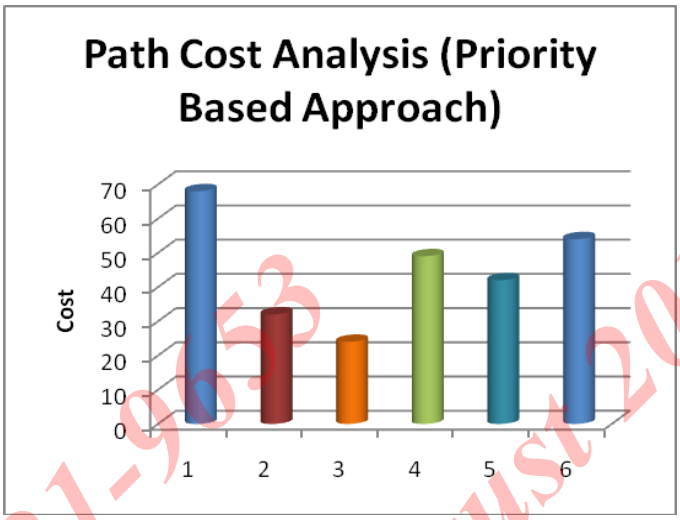

Fig: 2 Cost Analysis (Priority Based Approach)

Here figure is showing the comparative analysis of different cost testing path cost in case of priority based test path estimation. Here we have considered the priority between 1 and 3. Case 1 represents when all priority are 1, case 2 represents when all priorities are 2 an case 3 represents when all priorities are 3. Case 4 represents the priorities assigned in ascending order and case 5 represents the priority assignment in descending order. Case 6 represents the random assignment of the priorities. Here y axis represents the cost of test path.

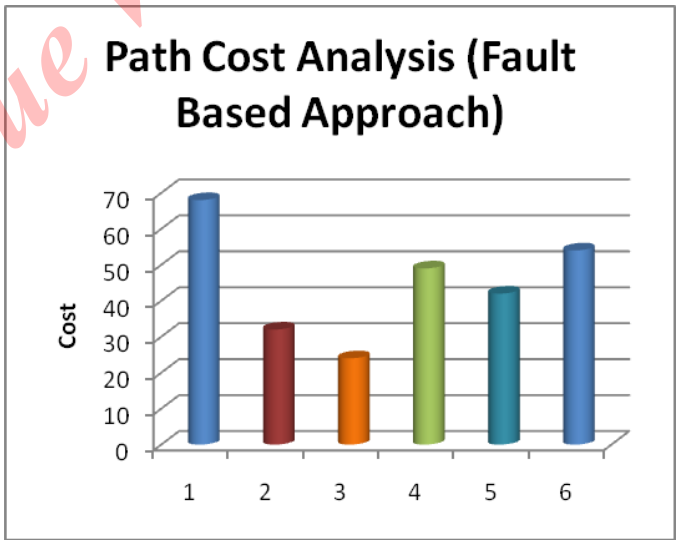

Fig:3 Cost Analysis (Fault Based Approach)

Here figure is showing the comparative analysis of different cost testing path cost in case of fault based test path estimation. Here we have considered the fault criticality between 1 and 3. case 1 represents when all fault levels are 1, case 2 represents when all fault levels are 2 an case 3 represents when all priorities are 3. fault levels 4 represents the priorities assigned in ascending order and case 5 represents the fault levels assignment in descending order. Case 6 represents the random assignment of the fault criticality. Here y axis represents the cost of test path.

VIII. CONCLUSION

In this present work we have improved the existing regression path testing approach by implementing the ACO approach. A software project is the sequence of correlated code modules where some test cases are associated with each module. In this present work we have defined these test cases respective to the fault occurrence parameter. According the importance of test cases, some priority value is assigned to each test case. After this the dynamic programming approach is implemented to find the best path respective to the low cost and less chances of fault occurrence during the testing process.

ACKNOWLEDGEMENT

I would like to express my deep gratitude to Assistant Professor Ms. Sunita, my research work supervisor, for her valuable and constructive suggestions during the planning and development of this research work. Her willingness to give her time so generously has been very much appreciated.

REFERENCES

[1] Williams L., "Testing Overview and Black-Box Testing Techniques", page no.35- 59, 2006.

[2] Pressman, Roger S.,"Software engineering: a practitioner's approach" 5th edition, 2001.

[3] Last M., Eyal S., and Kandel A.,"Effective Black-Box Testing with Genetic Algorithms",2005

[4] Singh K., Kumar R., "Optimization of Functional Testing using Genetic Algorithms" ,International Journal of Innovation, Management and Technology, Vol. 1, No. 1, April 2010.

[5] Engström E., "Regression Test Selection and Product Line System Testing", Third International Conference on Software Testing, Verification and Validation, 978-0-7695-3990-4/10 $26.00 © 2010 IEEE

[6] Jin W., Orso A., "Automated Behavioral Regression Testing", Third International Conference on Software Testing, Verification and Validation 978-0-7695-3990-4/10 © 2010 IEEE

[7] Do H., Mirarab S., "The Effects of Time Constraints on Test Case Prioritization: A Series of Controlled Experiments", IEEE Transaction on Software engineering, vol. 36, no. 5, September, 0098-5589/10/ 2010 IEEE

[8] Foo K., Jiang Z., Adams B., "Mining Performance Regression Testing Repositories for Automated Performance Analysis",10th International Conference on Quality Software, 1550-6002/10 © 2010 IEEE

[9] Zhang C., *et al.* "An Improved Regression Test Selection Technique by Clustering Execution Profiles", 10th International Conference on Quality Software 1550-6002/10 © 2010 IEEE

[10] Kumar A., Tiwari S., Mishra K., "Generation of Efficient Test Data using Path Selection Strategy with Elitist GA in Regression Testing", 978-1-4244-5540-9/10 ©2010 IEEE

[11] Li B., *et al.*, "Automatic Test Case Selection and Generation for Regression Testing of Composite Service Based on Extensible BPEL Flow Graph", and 60773105, and partially by National High Technology Research and Development, 26th international conference on software maintenance, 978-1-4244-8628-1/10 ©2010 IEEE

[12] Kumar M., *et al.,"* Requirements based Test Case Prioritization using Genetic Algorithm", IJCST Vol. 1, Issue 2, December 2010.

[13] Rothermel G., Roland H.," Test Case Prioritization: An Empirical Study", International Conference on Software Maintenance, Oxford, UK, September, 1999, IEEE Copyright.

[14] Rus I., *et al.*, "Software Dependability Properties: A Survey of Definitions, Measures and Techniques". Fraunhofer Technical Report 03-110, January 2003.

[15] Yang B., Wu J., "A Regression Testing Method for Composite Web Service", 978-1- 4244-5316-0/10/$26.00 ©2010 IEEE

[16] Gu Q., Chen Q.," Optimal Regression Testing based on Selective Coverage of Test Requirements", International Symposium on Parallel and Distributed Processing with Applications, 978-0-7695-4190-7/10 $26.00 © 2010 IEEE

[17] Tao C., *et al.*," An Approach to Regression Test Selection Based on Hierarchical Slicing Technique", 34th Annual IEEE Computer Software and Applications Conference Workshops, 978-0-7695-4105-1/10 $26.00 © 2010 IEEE.