MULTI-OBJECTIVE GENETIC ALGORITHM FOR REGRESSION TESTING REDUCTION

Ravneet Kaur

M.Tech scholars Department of Computer Science and Engineering, GIMET, Amritsar, Punjab, India

Abstract

Regression Testing is type of testing which is used to cut off the directly price associated with testing of different modules. Generally regression testing performs on test cases so that the resource utilization should be very low. Test cases in it are challenging task to achieve and suitable test cases for regression testing are difficult to judge and process. Optimization of various testing processes is done by optimization algorithms such as Genetic and Ant colony which usually provide solution to the good regression testing. Still these algorithms lack of some features which are required for better optimization of test cases in regression testing. The regression testing is the most expensive phase of the software testing, regression testing reduction eliminates the redundant test cases in the regression testing suite and saves the cost of the regression testing. In our proposed work, we will focus on optimization of regression testing with multi-objective genetic algorithm which will cover parameters like simplicity and complexity for test cases for regression testing. The complexity and simplicity for test cases will be judged and according to a common fitness function threshold we will proceed with optimization of the regression testing phases. Finally the paper evaluates the basic genetic algorithm for optimizing the test cases based on execution time; implement the multi-objective genetic algorithm with simplicity and complexity of the test cases along with execution time for test case prioritization for regression testing.

Index Terms: *Genetic Algorithm, Regression Testing Reduction, Test Cases, and Fitness Function etc…*

----------***----------

1. INTRODUCTION

Regression testing is any type of software testing that seeks to uncover software errors by retesting a modified program. The intent of regression testing is to provide a general assurance that no additional errors were introduced in the process of fixing other problems or modifications of software. Regression test suites are often simply test cases that software engineers have previously developed, and that have been saved so that they can be used later to perform regression testing. Re executing all the test cases require enormous amount of time thus make the testing process inefficient. For example, one industrial collaborator reports that for one of its products of about 20,000 lines of code, the entire test suite requires seven weeks to run. Prioritizing test cases provide the opportunity to maximize some performance goals or effectiveness. One of the performance goals may be rate of dependency detection among faults. During software testing, pragmatic experiences show that independent faults can be directly detected and removed, but mutually dependent faults can be removed if and only if the leading faults have been removed. That is, dependent faults may not be immediately removed, and the +fault removal process lags behind the fault detection process. For example, if software takes limited number of inputs and after functioning generates several types of outputs then a single fault in input module may generate a large number of faults in output module if they are not mutually independent. So, in regression testing if the test cases that reveal the faults of output module execute first and test cases reveals faults of input module executes later then it will be delayed and in many cases will take long time to detect the original cause of output faults.

1.1 Unit Regression

Unit Regression will be done during Unit testing. Unit Regression is done only when a defect is fixed in a module, if that module has relationship with other sections within that Module, then we perform Unit Regression. We will be testing only the fixed part of the module.

1.2 Partial Regression

Partial regression is when regression will be done after Impact Analysis and we would go for Partial regression means complete Regression suite will not be executed.

2. GENETIC ALGORITHM

Genetic algorithm is a random searching method that has a better optimization ability and internal implicit parallelism. It can obtain, and instruct the optimized searching space and adjust the searching direction automatically through the optimization method of probability. Genetic algorithm is a random searching method. After the first population is generated, it evolves better and better approximate solutions using the law of survival of the fittest from the generations. An individual is chosen in every generation based on the fitness of different individuals in certain problem domains. A new population representing a new solution set is

produced when different individuals combine, cross, and vary by genetic operators in natural genetics.

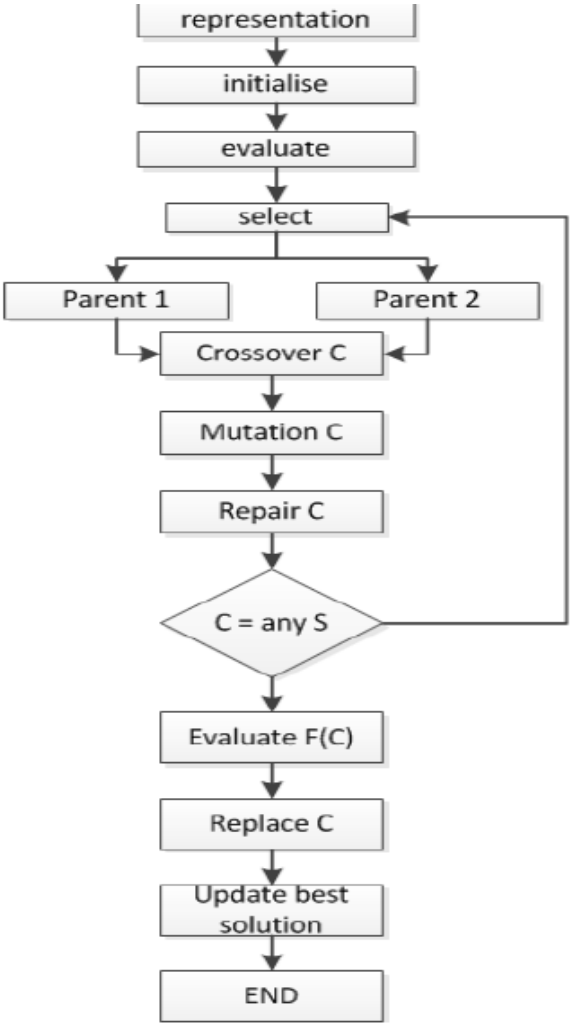

Chart -1: Genetic Algorithm

2.1 TEST CASES

Test cases are the combination of testing units for software or any coding. Normally the complete regression testing of the codes take many resources. To cut off the resources for the development phases, code is divided into various groups so that partial groups can be tested on the bees of the essential testing for partial groups. Now partial groups used to test which save resources. Further for more resource saving and accuracy, priority is provided to the groups on bases of various attributes to reduce further cost of testing.
The genetic and multi genetic are similar in their approach but difference lies in the performance of the multi objective. The comparison can be done on the bases of fitness function of the genetic process.

1. The simplicity of the process can be obtained with accordance to accessing process of the test case. In our research, we have processed the test cases for testing reduction. Number of test cases will be used for the breaking of testing affords. Test cases will be set as input and according to the fitness functions we will calculate the simplicity by find the chances of occurrence of the particular test case. As occurrence possibility is more than simplicity considered to be good for test case.

2. In case of complexity, we will calculate the process time for the test case processing, if in a single iteration, the time taken is more and in other iteration time taken is less for same test case then we will consider it complex in processing

2.2 Related Work

Our research will start with study of test case prioritization process for various types of codes and find the existing process for regression testing available. Our research will start with developing a language code (probably in java) and then to test it for errors. Total regression testing process will be done on it. Once done, we will find the test cases for complete code which will be used as test case in further testing. Fitness function of the multi-objective genetic algorithm will be decided on the individual fit of the test case. We will try to reduce the regression testing resources by reducing the test cases to have reduced test suites for testing. Less test suite will be used for regression testing so that resources can be saved. We will start with simple regression strategies such as testing of whole code without use of any test case. After that we will find simple test case scenario which will save resources and improve efficiency of bug finding also. Finally for providing better test case selection process, we will process the set of test cases created so far with multi objective genetic algorithm which will consider the optimized filtering of the test cases.

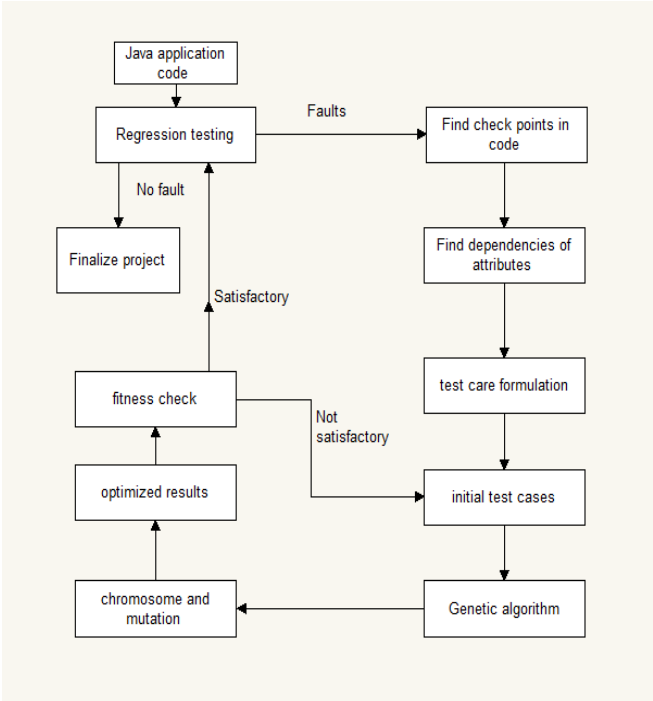

Chart-2: Fault Finding Algorithm

CONCLUSIONS

This paper proposed a multi-objective genetic algorithm for regression testing reduction. The multi objective genetic algorithm overcomes the short comes of genetic algorithm. This paper will focus on optimization of regression testing with multi-objective genetic algorithm which will cover parameters like simplicity and complexity for test cases for regression testing. The complexity and simplicity for test cases will be judged and according to a common fitness function threshold we will proceed with optimization of the regression testing phases.

REFERENCES

[1] Liang You, Yansheng Lu, ''A Genetic Algorithm For The Time-Aware Regression Testing Reduction Problem'', 8th International Conference on Natural Computation, ICNC 2012.

[2] R. Savenkov,"How to become a software tester", Roman Savenkov Consulting, 2004.

[3] 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.

[4] Christof Budnik, Siemens," Software Testing, Software Quality and Trust in Software-Based Systems", Siemens.com.

[5] Md. Imrul Kayes, "Test Case Prioritization for Regression Testing Based on Fault Dependency", IEEE International Conference of Software Engineering, February 2012.

[6] 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.

[7] 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.