ISSN:0975-9646

Genetic Based Prioritization Regression Testing

Quience Gulati
M. Tech (Computer Science)
Amity School of Engineering and Technology
Noida, India

Arun Prakash Agarwal
Dept. of Computer Science
Amity School of Engineering and Technology
Noida, India

Abstract— When previously developed software is modified, there are chances that there may be lot of errors, which may lead to failure of the software system. The cost required to test every module of the system will be very high and even after that, testing every part of the system is not possible as there is limited time given to this phase. Regression testing is very common these days as it helps in testing the modified and changed part, also reducing time and cost. The number of test cases is reduced making use of the fitness function of genetic algorithm. Also the test cases are assigned priorities, which help in testing the more critical part on priority. A dynamic approach can be used to achieve the same.

***Index Terms*—Genetic algorithm based regression testing, test case prioritization, regression testing optimization, software testing.**

I. Introduction

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 number of available factors are there. The estimation cost calculated and reserved for maintenance phase is exceeded by 70% of the total cost of the software. So to enhance the efficiency, regression testing is used. Regression testing not only helps in reduction of time and cost but it also helps in checking the quality of the software system. Regression testing may be corrective or progressive, which is solely dependent on the effect of changes and modifications made. It helps in eliminating the redundant test cases[1].

After the modifications have been made in the software system, there is a possibility that there might exist a number of errors. The part of the system, where modification has been done is tested for errors as it is more prone to errors. Regression testing when used along with genetic algorithm, helps in reduction of test cases which indeed helps in reduction of the testing cost. An evolutionary approach is used Genetic algorithm uses fitness function to reduce the test suite. The fitness function is used to calculate the fitness value of the test case population. As a result only the fit tests remain in the reduced test suite. The selection amongst the population is done followed by 2 point crossover, on the test suite.

Though selection of the test suite can be made using different ways and approaches, algorithms can be made to achieve the same. Either the heuristic approach or evolutionary approach can be used according to the requirement of the tester. But evolutionary approach using the genetic algorithm gives better results. Also a dynamic approach can be used as and how required by the tester.

It was proved by Md. Imrul that prioritized test cases are far more effective than the non-prioritized test cases[3]. According to this fault oriented module analysis the prioritization will be assigned to the test cases. The main objective is to avail fault prone software after the regression testing. In this fuzzy improved genetic approach is defined to perform regression testing.

II. Literature Review

DR Shaheen and Dr Kosba[2] have proposed an improved test suite prioritization technique of structural software testing by increasing the test code coverage by maximizing the number of coverage items. It is done using multiple control flow based coverage criteria to save time and cost. The test cases were taken as genes and the test suite was considered as chromosome. A new fitness function was used in genetic algorithm, in which weights were assigned to the test cases, fault severity and the rate of the fault. They had compared their work using average percentage of fault detected (APFD) and better results were shown by new approach. So a new fully automated test case prioritization technique was proposed by them that can be easily used by the software testers.

Md. Imrul Kayes[3] proposed a metric to measure the effectiveness of test case prioritization in regression testing. The analysis can be done on prioritized or non-prioritized test cases with the help of the proposed metric. A new test case prioritization algorithm has been implemented for prioritizing the test suite so as to maximize the detection of fault at the time execution of test cases that have been prioritized. The test cases which have severe faults are executing first and the test cases with less severe faults are executed later depending upon the business need. It was proved that prioritized test cases were more effective than non-prioritized test cases in detecting dependency among faults.

Ghinwa Baradi and Nashat Mansour[4] compared the five regression testing algorithms and proved that out of slicing, incremental, firewall, genetic, simulated annealing algorithms, incremental algorithm would be the best choice as it is comparatively effective and faster than the rest. The comparison was carried out keeping both quantitative and qualitative parameters. The qualitative criteria included precision, execution time, number of test cases, inclusiveness. The quantitative criteria includes user's parameter settings, testing of global variables, type of maintenance, type of testing, level of testing, type of approach. The advantages and disadvantages of all the five algorithms were explained separately. Simulated annealing and incremental algorithms can be used for appropriate selection of the test cases. The algorithm selection is dependent on the requirements of the software tester.

S. Nachiyappan, A. Vimaladevi and C. B. SelvaLakshmi[1] have used genetic algorithm as evolutionary approach for reducing the test cases. It was noticed that to reduce the test cases, the coverage was not compromised with, which was examined using bar graph. A highly adaptive method was obtained based on block coverage value and using genetic operators, selection-roulette wheel, mutation and crossover-2 point crossover. Depending upon the fitness value-good or bad the test case is selected.

Liang You and Yansheng Lu[5] proposed a genetic algorithm for the time aware regression testing reduction problem. The time-aware regression testing reduction problem is discussed along with the related work done on this. The main objective of this problem is to minimize the execution time of the remaining test cases after the removal of the redundant test cases. The evaluation of the genetic algorithm to examine the efficiency is using eight example problems.

Praveen Ranjan Srivastava[6] focused upon scheduling the test cases so as to improve the performance of regression testing. Explaining the four methodologies of regression testing: retest all, regression test selection, test suite reduction and test case prioritization. It was proved that prioritized test cases show better results than the non prioritized test cases. Analysis of prioritized and non-prioritized was done using average percentage of fault detected (APFD).

III. Basic Flow

If we have constant test cases set that is test suites for regression testing and they are executed irrespective of the number and type of bug fixes than it may not uncovers an as-yet undiscovered errors. In such a case our testing objectives will fail because a good test always intent to find errors. And it is unusual for any organization to expand between 30-40 percent of the total project efforts on testing. The efforts spend on executing test cases for regression testing can be minimized if analysis is done to find out- "what test cases are relevant and what are not".

Using regression testing, the tester gets confidence that atleast the critical faults will be detected and thus can be removed after identification. Hence the efficiency of the software system is increased. Using Regression testing, depending upon the requirements the tester can use any of the methodologies amongst retest all, regression test selection, test suite reduction, test case prioritization. Figure 1 shows a basic model of regression testing process, from the selection of the test cases to execution of the test cases.

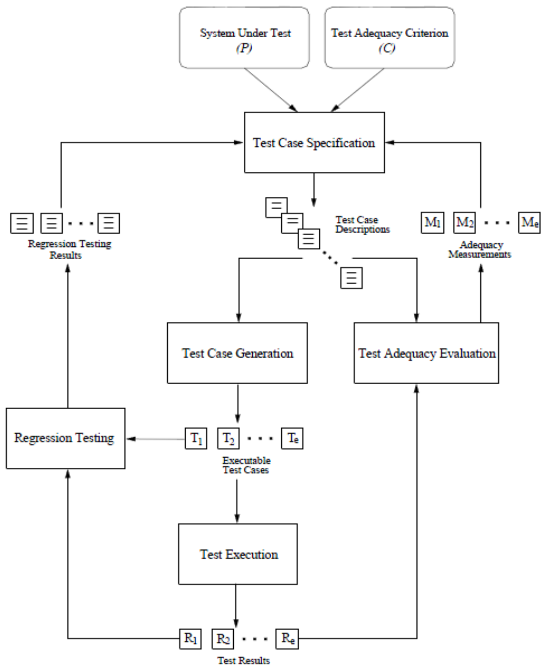

Fig. 1. Model of Regression Testing Process

The basic software flow analysis is here defined under the fault analysis. The work flow is shown here under:

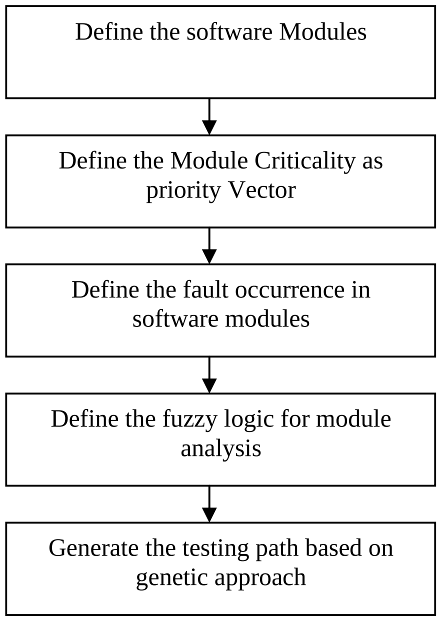
Define the software Modules
Define the Module Criticality as priority Vector
Define the fault occurrence in software modules
Define the fuzzy logic for module analysis
Generate the testing path based on genetic approach

Fig. 2. Software Testing Process Flow

The figure above explains the flow of testing. The software system is first divided into modules so as to make it less complex. To every module the criticality is defined as priority vector, then in these modules the probability of occurrence of fault is defined. The module is then analyzed using fuzzy logic. The testing path is generated accordingly, based on the genetic approach.

IV. Genetic Algorithm

The Genetic algorithm is an evolutionary algorithm and helps in optimization. It requires genetic representation and fitness function to find an optimal solution. The selection takes place from the available population using fitness function, genetic operators are applied to obtain an optimal solution, followed by termination.

A. Selection

Individuals are selected amongst the population on the basis of their fitness value. The selection can be using roulette wheel selection or tournament selection method. Roulette wheel selection is done on the basis of fitness of each individual while in tournament selection, the individuals are selected randomly.

B. Genetic Operators

The genetic operators are applied on the individuals selected using the roulette wheel or tournament selection method. The genetic operators are crossover and mutation. The crossover operator is applied on two parents to obtain an optimal solution, solution in this case is a child. There are types of crossover: 1-point crossover, 2- point crossover, uniform crossover and multipoint crossover. Mutation helps in introducing new genetic structure from one generation to the next. It is assumed that mutation provides comparatively better solutions. Mutation can be flipping of genome, interchanging, bit strings mutation.

V. Conclusion

In this paper we have examined the different ways and algorithms that can be used to reduction the test case suite. It is noticed that several companies have "constant test cases set" for regression testing and they are executed irrespective of the number and type of bug fixes. Sometimes this approach may not find all side effects in the system and in some cases it may be observed that the effort spend on executing test cases for regression testing can be minimized, if some analysis is done to find out what test cases are relevant and what are not. Study shows the improvements in algorithms going through as time passed.

Because there is a requirement of some dynamic approach that can reduce the test cases while performing the regression testing. It also requires a dynamic approach to assign the prioritization to the test cases. As a result a new sequence will be generated. The proposed approach will not only perform the inclusion of new test cases and the elimination of useless test cases. This whole process will be done dynamically. It will also assign and change the priorities of test cases dynamically depending on the use cases.

References

[1] S. Nachiyappan, A. Vimaladevi and C. SelvaLakshmi [2010], “An Evolutionary Algorithm for Regression Test Suite Reduction”.

[2] A. Ahmed, M. Shaheen and E. Kosba [2012], "Software Testing Suite Prioritization using Multi-Criteria Fitness Function".

[3] Md. Imrul[2011], “Test Case Prioritization for Regression Testing Based on Fault Dependency”.

[4] Ghinwa Baradhi and Nashat Mansour[1997], “A Comparative Study of Five Regression Testing Algorithms”.

[5] L. You and Y. Lu[2012], “A Genetic Algorithm for the Time AwareRegression Testing Reduction Problem”.

[6] Elbaum, Malishevsky, Rothermel [2002],"Test case prioritization: a family of empirical studies".

[7] David Leon, Andy Podgurski [2003],"A Comparison of Coverage-Based and Distribution-Based Techniques for Filtering and Prioritizing Test Cases".

[8] Xiaofang Zhang, Changhai Nie, Baowen Xu, Bo Qu [2007], "Test Case Prioritization based on Varying Testing Requirement Priorities and Test Case Costs".

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

[10] Wei Jin and Alessandro Orso, “Automated Behavioral Regression Testing”, 2010 Third International Conference on Software Testing, Verification and Validation 978-0-7695-3990-4/10 © 2010 IEEE

[11] Hyunsook Do, Member, Siavash Mirarab, “The Effects of Time Constraints on Test Case Prioritization: A Series of Controlled Experiments”, IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, VOL. 36, NO. 5, SEPTEMBER, 0098-5589/10/ 2010 IEEE

[12] King Chun Foo, Zhen Ming Jiang, Bram Adams, “Mining Performance Regression Testing Repositories for Automated Performance Analysis”, 2010 10th International Conference on Quality Software, 1550-6002/10 © 2010 IEEE

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

[14] Lin Chen Ziyuan Wang Lei Xu, “Generation of Efficient Test Data using Path Selection Strategy with Elitist GA in Regression Testing”, 978-1-4244-5540-9/10 ©2010 IEEE.