International Journal of Advancement in Engineering Technology, Management & Applied Science
Volume1, Issue 1 June 2014 ISSN No: 2349-3224

Title
A study of Regression Testing and their techniques

Devanshu Sharma, Kapil Sharma

C.C.S University, India

Abstract:

Regression testing is an expensive, but important process. Unfortunately, there may be insufficient resources to allow for the re-execution of all test cases during regression testing. Regression testing is usually performed by running some, or all, of the test cases created to test modifications in previous versions of the software. Many techniques have been reported on how to select regression tests so that the number of test cases does not grow too large as the software evolves.

Keywords: Regression Testing, Test case prioritization, Techniques

Introduction:

It is one of the most critical activities of software development and maintenance. Whenever software is modified, a set of tests are run and the comparison of new outputs is done with the older ones to avoid unwanted changes. If new output and old output match it implies the modifications made in one part of the software didn't affect the remaining software It is not appropriate to re-execute every test case for a program if changes occur [1].Regression testing must be conducted to confirm that recent program changes have not adversely affected existing features and new tests must be conducted to test new features. Testers might rerun all test cases generated at earlier stages to ensure that the program behaves as expected. However, as a program evolves the regression test set grows larger, *old* tests are rarely discarded, and the expense of regression testing grows. Repeating all previous test cases in regression testing after each minor software revision or patch is often impossible due to the pressure of time and budget Constraints. On the other hand, for software revalidation, arbitrarily omitting test cases used in regression testing is risky. In this paper, we investigate methods to select small subsets of effective fault-revealing regression test cases to revalidate software.
Many techniques have been reported in the literature on how to select regression tests for program revalidation. The goal of some studies [1, 3, 13] is to select every test case on which the new and the old programs produce different outputs, but ignore the coverage of these tests in the modified program. In general, however, this is a difficult, sometimes undesirables, and problem Others [5, 8, 10, 15, 18,] place an emphasis on selecting existing test cases to cover *modified*

program components and those may be affected by the modifications, i.e., they use coverage information to guide test selection. They are not concerned with finding test cases on which the original and the modified programs differ. Consequently, these techniques may fail to select existing tests that expose faults in the modified program. They may also include test cases that do not distinguish the new program from the old for re-execution. In this paper we defined different techniques.

Regression test selection

The problems of regression test selection can be solved by prioritizing test cases. Regression test Prioritization techniques reorder the execution of a test suite in an attempt to ensure that defects are revealed earlier in the test execution phase [11].

Regression Test Selection Techniques

A variety of regression test selection techniques have been described in the research literature. Rothermeland Harrold [ZO] describes several families of techniques; we consider three such families, along with two additional approaches often used in practice. We here describe these families and approaches, and give a representative example of each - we utilize these representative examples in our experimentation.

Minimization Techniques

These techniques attempt to select minimal sets of tests from T that yield coverage of modified or affected portions of P. One such technique requires that every program statement added to or modified for P‘ be executed (if possible) by at least one test in T.

Safe Techniques

These techniques select, under certain conditions, every test in T that can expose one or more faults in PI. One such technique selects every test in T that, when executed on P, exercised at least one statement that has been deleted from P, or at least one statement that is new in or modified for P‘.

Dataflow-Coverage-Based Techniques

These techniques select tests that exercise data interactions that have been affected by modifications One such technique selects every test in T that, when executed on P, exercised at least one definition use pair that has been deleted from P', or at least one definition-use pair that has been modified for P'.

Ad Hoc / Random Techniques

When time constraints prohibit the use of a retest-all approach, but no test selection tool is available, developer soften select tests based on "hunches", or loose associations of tests with functionality. One simple technique randomly selects a predetermined number of tests from T.

Retest-All Technique

This technique reuses all existing tests. To test P', the technique "selects" all tests in T. [17]

Genetic Algorithm

GA is a powerful search and optimization algorithm, which are based on the theory of natural evolution. In GA, each solution for the problem is called a chromosome and consists of a linear list of codes. The GA sets up a group of magi nary lives having a string of codes for a chromosome on the computer. The GA evolves the group of imaginary lives (referred to as population), and gets and almost optimum solution for the problem. The GA uses three basic operators to evolve the population: selection, crossover, and mutation. Genetic algorithm was developed by John Holland-University of Michigan (1970.s) to provide efficient techniques for optimization and machine learning applications through application of the principles of evolutionary biology to computer science. It uses a directed search algorithms based on the mechanics of biological evolution such as inheritance, mutation, natural selection, and recombination (or crossover). It is a heuristic method that uses the idea of .survival of the fittest. In the genetic algorithm, the problem to be solved is represented by a list of parameters which can be used to drive an evaluation procedure, called chromosomes or genomes. Chromosomes are typically represented as simple strings of data and instructions. In the first step of the algorithm, such chromosomes are generated randomly or heuristically to form an initial pool of possible solutions called first generation pool. In each generation, each organism (or individual) is valuated, and a value of goodness or fitness is returned by a fitness function. In the next step, a second generation pool of organisms is generated, by using any or all of the genetic operators:

selection, crossover (or recombination), and mutation. Pair of organisms is selected for to survive based on elements of the initial generation which have better fitness. In other words, the organisms that have relatively higher fitness than other organisms in the generation are selected to survive. Some of the well-defined organism selection methods are roulette wheel selection and tournament selection. After selection, the crossover (or recombination) operation is performed on the selected chromosomes, with some probability of crossover *(Pc)*-typically between 0.6 and 1.0.Crossover results in two new child chromosomes, which are added to the second generation pool. The crossover operation is done by simply swapping a portion of the underlying data structure of the chromosomes of the parents. This process is repeated with different parent organisms until there are an appropriate number of candidate solutions in the second generation pool. In the mutation step, the new child organism's chromosome is randomly mutated by randomly altering bits in the chromosome data structure. The aim of these is to produce a second generation pool of chromosomes that is both different from the initial generation and hence have better fitness, since only the best organisms from the first generation are selected for surviving. The same process is applied for the second, third, generations until an organism is produced which gives a solution that is "good enough". The algorithm starts with an initial set of random solutions called the population. Each individual in the population, known as chromosome, represents a particular solution of the problem. Each chromosome is assigned a fitness value depending on how good its solution to the problem is. After fitness allotment, the natural selection is executed and the 'survival of the fittest chromosome' can prepare to breed for the next generation. A new population is then generated by means of genetic operations: cross-over and mutation. This evolution process is iterated until a near-optimal solution is obtained or a given number of generations is reached.

Fitness function:

In order to identify the best individual during the evolutionary process, a function needs to assign a degree of fitness to each chromosome in every generation. So in order to determine whether the assumed region of the input image is a face or not, The fitness value of the possible face region is computed by means of similarity.

Selection:

Selection operator is a process in which chromosomes are selected into a mating pool according to their fitness function. Good chromosomes that contribute their gene-inherited knowledge to breed for the next generation are chosen.

Cross-over:

This operator works on a pair of chromosomes and produces two offspring by combining the partial features of two chromosomes. Here we will have to learn single point cross-over, two point cross-over and uniform crossover operators.

Mutation:

This operator alters genes with a very low probability. The components of the genetic algorithm explained above can also be summarized as below [16].

Conclusion:

Regression testing is the process of retesting the modified parts of the software and ensuring that no new errors have been introduced into previously tested code. In addition to validating added functionality, regression testing compares the behavior of a new version to the old version to check that no faults are introduced. When the outputs produced by two versions are different, it implies faults have been introduced into the system. Steps are then taken for the removal of these faults.

Regression test selection techniques reduce the cost of regression testing by selecting an appropriate subset of the existing test suite based on information about the program, modified version and test suite. Test suite minimization technique lower costs by reducing a test suite to a minimal subset that maintains equivalent coverage of the original test suite with respect to a particular test adequacy criterion.

References:

[1] W. Eric Wong, J. R. Horgan, Saul London and Hira Agrawal ,"A Study of Effective Regression Testing in Practice",8th*International Symposium on Software Reliability Engineering*,pp.264-274,_Albuquerque, NM ,1997.
[2] Zengkai Ma and Jianjun Zhao, "Test Case Prioritization Basedon Analysis of Program Structure", *15th Asia-Pacific Software Engineering Conference*, pp.471-478, Beijing, 2008
[3] Walid S. Abd El-hamid, Sherif S. El-etriby, and Mohiy M.Hadhoud "A General Regression Test Selection Technique",*World Academy of Science, Engineering and Technology,*2010.
[4] Gregg Rothermel, Roland H. Untch and Mary Jean Harrold,"Prioritizing Test Cases For Regression Testing", *IEEETransaction on Software Engineering*, Vol. 27, pp.929-948, 2001.
[5] Walid Said Abd El-Hamid, Sherif Said El-etriby and Mohily Mohammed Hadhoud, "Regression Test Selection Technique for Multi-Programming Language", *7th International Conference on Informatics and Systems (INFOS)*, pp.1-5, Cairo, 2010.
[6] Bing Jiang, Yongmin Mu and Zhihua Zhang, "Research of Optimization Algorithm for Path-Based Regression TestingSuit",*2nd International Workshop on Education Technology and Computer Science*,Vol.-2,pp.303-306,Wuhan, 2010.
[7] Linda Badri, Mourad Badri and Daniel St-Yves, "Supporting Predictive Change Impact Analysis: A Control Call Graph Based Technique", *12th Asia-Pacific Software Engineering conference*,pp.9, 2005.
[8] Yuming Zhou, Baowen Xu, Jianjun Zhao and Hongji Yang,"ICBMC: An Improved Cohesion Measure For Classes",*Proceedings of the International Conference on Software Maintenance*, pp.44-53 , 2002.
[9]Edward B. Allen and TaghiM. Khoshgoftaar, "Measuring Coupling and Cohesion: An Information-Theory Approach",6th*International Software Metrics Symposium*, pp.119-127, Boca Raton, FL , Aug 2002.
[10] Gregg Rothermel and Mary Jean Harrold, "Analyzing Regression Test Selection Techniques", *IEEE Transaction on Software Engineering*, Vol. 22, pp.529-551,Aug1996.
[11] Michael Ruth and ShengruTu,"A Safe Regression Test Selection Technique for Web Services",*2nd International*
Conference on Internet and Web Application and Services, pp.47,_Morne ,2007.
[12] Khin Haymr Saw Hla, Young Sik Choi and Jong Sou Park,"Applying Particle Swarm Optimization to Prioritizing Test Cases
for Embedded Real Time Software Retesting ",*IEEE 8thInternational Conference on computer and Information Technology Workshopes*, pp.527-532 , 2008.
[13] N Gnanasekaran and Vineet Banga, "Automated Regression Testing Framework ", *NIIT Technology Ltd.*, pp.1-19, July02nd,2007.
[14] Lionel C. Briand, Jie Feng and Yvan Labiche, "Using Genetic Algorithms and Coupling Measures to Device Optimal Integration Test Orders", *IEEE Transaction on Software Engineering,*pp.43-50,Ischia,2002.

[15]Gregg Rothermel and Mary Jean Harrold, “Empirical Studies of a Safe Regression Test Selection Technique”, *IEEE Transactions on Software Engineering*, Vol. 24, pp.401-419,June 1998.
[16]http://www.talkorigins.org/faqs/genalg/genalg.html
[17] Todd L. Graves et al.,“An Empirical Study of Regression Test Selection Techniques” *Proceedings of the International Conference on Software Engineering*, pp.188-197,1998.
[18] Manoj, Proceedings of the National Conference on Advances in Computational Intelligence,pp.448-451