ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 2, February 2016

Survey of Optimization Techniques for Test Suite Optimization in Regression Testing

Sara Amjad[1], Kuldeep Jaiswal[2]

M.Tech Student, Dept. of Computer Science and Engineering, BBD University, Lucknow, India[1]

Assistant Professor, Dept. of Computer Science and Engineering, BBD University, Lucknow, India[2]

ABSTRACT: This paper aims to find optimization techniques for test suite optimization (TSO) in Regression testing(R/T). We have studied various optimization techniques and algorithms in order to find the technique that should be used for test suite optimization (TSO) in regression testing. The techniques like hill climbing, greedy algorithm, additional greedy algorithm,2-optimal greedy algorithm, genetic algorithm, particle swarm optimization(PSO) are studied and compared using different parameters. These techniques are compared on the basis of their code coverage area, computational effort and execution time.

KEYWORDS: Algorithms, Optimization techniques, Regression testing, Test suite optimization.

I. INTRODUCTION

Every time software is modified it is required to be tested. Regression testing is a black box testing technique that is used to identify bugs that have emerged at the time of software modification. Regression testing depends upon quality of test suites. A good quality test suite leads to early fault detection and hence improves quality of regression testing. Test suite optimization techniques are used for prioritization, selection and minimization of test cases in a test suite. Hill climbing, greedy algorithm, additional greedy algorithm,2-optimal greedy algorithm, genetic algorithm, particle swarm optimization(PSO) are optimization algorithm.

Particle swarm optimization is a optimization Technique inspired by social behaviour of bird flocking. In case of particle swarm optimization the next position of a particle depends on its local best position and the global best position of the swarm.

The Genetic algorithm is an optimization algorithm that uses cross over and mutation operator to find the global best solution. Genetic Algorithm include following steps:

1. Initialization-Create initial set of randomly generated particles.
2. Fitness calculation-Calculate fitness for each randomly generated particle.
3. Selection-Select portion of existing population for generating new population. The selection is done using fitness value.
4. Applying mutation and crossover- Genetic operator mutation and crossover are applied to generate new set of population by using selected portion of existing population such that new set of population should have higher frequency than parent population.

PSO includes following steps:

1. Create random population of particle.
2. Find fitness value of each particle in the population.
3. If particles current position is better than its previous position make it personal best (pbest). For first iteration the initial position of the particle will be particles best position.
4. Find the global best (gbest) particle .
5. Find the new position of the particle with the help of personal best and global best position of the particle. Change in position of particle is represented as velocity update.

- Vi(t+1)= w*Vi(t)+c1*rand*(pbest-p)+c2*rand*(gbest-p)
- newP(t+1)=P+Vi(t+1)

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 2, February 2016

Vi(t+1)-update velocity
C1-Weight of local information
C2-Weight of global information
Rand- 0<=rand<=1
pbest- Particle personal best position
gbest- Particle global best position

6. Move particle to their new position.
7. Follow the step 3 to 6 .
8. We stop when no more better particles are found in next 50(or according to the problem space) iteration.

Test case prioritization: It is the process of improving sequencing of test cases in a test suite. Consider a test suite T consisting of n number of test cases. The ordering of test cases in the Test suite T can be done in P(n) (Permutation of n) ways. Test case prioritization aims to find the best orders of test cases in the test suite that can lead to early fault detection and can provide better code coverage area.
Test case selection: It is the process of selecting subset of the test suite. The subset of test suite is generated depending on the programs that are affected by the software modification. Test case selection reduces the number of test cases to be executed and thus reduces the execution time of the test suite.
Test suite minimization: It is the process of removing redundant and obsolete test cases in the test suite. It helps to improve the rate of fault detection.

II. LITERATURE SURVEY

Search Algorithms for Regression Test Case Prioritization:
In this paper five optimization algorithms are applied and compared on the basis of block coverage, statement coverage, decision coverage. The algorithms are applied on C programs Print_tokens, Print _tokens2, Schedule, Schedule2, Space, Sed. Print_tokens, Print _tokens2, Schedule, Schedule2 are small programs with few hundred lines and Space and Sed are large programs. Five optimization algorithms are[1]-
Greedy algorithm: A Greedy Algorithm is an implementation of the 'next best' search philosophy. It works on the principle that the element with the maximum weight is taken first, followed by the element with the second highest weight and so on, until a complete, but possibly sub–optimal solution has been constructed. Greedy search seeks to minimize the estimated cost to reach some goal. This paper explains with example that the greedy approach is not always good for test suite prioritization.[1]
Additional Greedy Algorithm: It is also a kind of greedy algorithm. In this algorithm the maximum weighted element are selected from the part of program that is not consumed by previously selected element. This paper explains with example the condition when different sequencing of test cases are obtained for the same test suite in case of additional greedy algorithm[1].
2-Optimal Greedy algorithm: The K–Optimal approach selects the next K elements that, taken together, consume the largest part of the problem. In the case of K–Optimal Additional Greedy, it is the largest remaining part of the problem that is selected. When k>3 the increase computational time is very high as compare to increase in software quality. The 2-Optimal Greedy approach is fast and effective.[6][1]
Hill climbing: Hill climbing is simple and easy to implement and execute but it provides works with local best solution and does not guarantee global best solution. [1]
Genetic algorithm: The genetic algorithm starts with randomly generated population. Fitness value is calculated for each individual of the population. In this paper coverage is considered as fitness value and the population consists of test suite with different ordering of test cases. The algorithm uses mutation and crossover operator. The algorithm works finding global best solution.[1]
Result of prioritization of test cases using different algorithms show that there is not much difference between the code coverage obtain by applying these algorithms on small programs. In case of large programs it is seen that only genetic algorithm guarantees global best solution in all condition.[1]
Genetic Algorithm for regression test case prioritization using code coverage:
In this paper genetic algorithm is used for prioritizing test cases in a test suite. Genetic Algorithm is applied manually on test suites of twenty test cases in order to prioritize test cases in the test suite on the basis of complete code

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 2, February 2016

coverage. An approach is used for initially selecting the suitable population and then genetic operation is applied to selected population.[5]

The paper compares random ordering, reverse ordering, optimal ordering and genetic ordering of test cases in the test suites on the basis of average percentage code coverage. It concludes that genetic algorithm can be used for prioritization of test cases in the test suite, with minimum execution time.[5]

A Comparison of Particle Swarm Optimization and Genetic Algorithm:

This paper examines that the quality of solution obtained by genetic algorithm is same as the quality of solution obtained by particle swarm optimization algorithm but the computational effort required by particle swarm optimization is less than that required by genetic algorithm.[2]

Effectiveness and efficiency test are performed to compare genetic algorithm with particle swarm optimization algorithm. These tests are carried out for eight sample problems: the Banana function, the Eggcrate function, Golinski's Speed Reducer, Reliability-based Satellite Design, 5-station, 6-station, 7-station and 8-station Radio Telescope Array Configuration.[2]

The effectiveness test is performed to compare quality of solution and efficiency test is performed to compare computational effort. The t-test hypothesis is used to compare the effectiveness and efficiency of both the algorithm. The result of the test shows that genetic algorithm and particle swarm algorithm both provide high quality solution but the computational effort required by particle swarm optimization is less than that required by genetic algorithm. [2]

Effectiveness of Test Case Prioritization Technique Based on Regression Testing:

In this paper a new approach is used for test case prioritization. The prioritization is done on the basis of six prioritization factors (customer allotted priority, developer observed code execution complexity, changes in requirements, fault impact, completeness and traceability). The prioritization factors help to assign weight to each test case. The prioritization is done on the basis of weight assigned to the test case. The prioritization technique is applied on bank application. Effectiveness is measured using average percentage fault detection metrics.[4]

This paper concludes that the value for average percentage fault detection is higher after prioritization than the value for average percentage fault detection before prioritization.[4]

Research on Optimization Scheme of Regression Testing:

This paper explains how regression testing can be improved by combining test case selection with test case prioritization. The test cases are selected by obtaining the programs that are affected by the software modification. The selected test cases are prioritized on the basis of their coverage ability and troubleshooting capability.

This paper concludes that the value of average percentage fault detection is highest when test case selection is combined with test case prioritization.[3]

III. CONCLUSION

This paper delivers review of techniques used for test suit optimization. It also gives detail about the comparison of different optimization techniques. The review can help the researchers in comparing the optimization techniques. It provides future work idea that can be used in improving regression testing. In my opinion we can use particle swarm optimization technique for test suite optimization purpose by improving the code coverage area of test suite.

This survey also shows that there is scope of future work in the field of regression testing. We can use this review for obtaining a better technique for test suit optimization in regression testing.

REFRENCES

1. Zheng Li, Mark Harman and Robert M. Hierons, "Search Algorithms for Regression Test Case Prioritization", IEEE Transaction Paper on Software Engineering ,Vol. 33 Issue 4, pp.225-237, 2007.
2. Rania Hassan, Babak Cohanim, Olivier de Weck, Gerhard Venter, "A Comparison of Particle Swarm Optimization and Genetic Algorithm.", 46th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference,pp.1-13,2005.
3. Shiming Sun, Xiuping Hou, Can Gao, Linlin Sun, "Research on optimization scheme of regression testing.",Ninth International Conference on Natural Computation,pp.1628-1632,2013.
4. Thillaikarasi Muthusamy and Dr. Seetharaman, "Effectiveness of Test Case Prioritization Technique Based on Regression Testing", , International Journal of Software Engineering & Applications, Vol. 5 No. 6,pp.113-123,2014.
5. Arvinder Kaur and Shubhra Goyal, " A Genetic Algorithm for Regression Test Case Prioritization using code coverage",
6. International Journal on Computer Science and Engineering ,Vol. 3 No. 5, pp.1839-1847, 2011.
7. S. S. Skiena. "The algorithm design manual". Springer-Verlag, New York, NY, USA,1998.