Test Suite Minimization Technique: A New Perspective Based on Genetic Algorithm for Object Oriented Programming

Vineeta Mishra[1]* Mohd Haroon[2] Shish Ahmad[3]

[1]Department of Computer Science & Engineering, Integral University, Lucknow

[2]Department of Computer Science & Engineering, Integral University, Lucknow

[3]Department of Computer Science & Engineering, Integral University, Lucknow

Abstract – Software testing is a very broad area, which involves many other technical and non-technical areas, such as specification, design and implementation, maintenance, process and management issues in software engineering. This report provides an automated approach to select test cases for object oriented program by using genetic algorithm. The given approach takes less time and meets adequacy criteria for example coverage. There are many approaches available for automatic selection of test cases for procedural pro- gram but the selection of test cases for object oriented program is challenging. The pro- posed approach enhances the previously available approaches and shows the good result.

Keywords: Software Testing, Test Case, Genetic Algorithm

- X -

1. INTRODUCTION

Software testing is an essential part of software development life cycle (SDLC). As the size of software and complexity of software increase day by day so that it is not possible to analyze the whole software by traditional method.

The process of testing is an expensive and time consuming in software development. It can consume at least 50% of the total costs involved in developing software (Beizer, 1990). To reduce the time and produce the improved quality software, we use the automation of the testing process. Ideally, testing software guarantees the removal of errors in the software, but in reality, it only reveals the presence of software errors but never guarantees their absence. If the software is not fully tested and released in the market then it gives the customer dissatisfaction. Today, software is used in many fields like in aerospace, in medical and many more. These fields involve the human life so that the testing of the software is necessary. It will take a life long time if one has to check the source code of a program or a huge man force. It is a kind of complex, labor-intensive, and time consuming work; so that one of the solution for testing the software before the deadline is automation of the software testing. The one of the automation technique for software testing is section of the test cases from the large set of test cases. We are giving an approach for selecting the test cases for object oriented program using the genetic algorithm.

The one of the basic problem in software industry is that there is always a chance for improvement of the software. The software improvement is required when there is:

- Change in requirements and code is modified according to changed requirement.
- New feature is added to the software according to the new requirement.
- Defect fixing in the previous version of the software.
- Try to improve the performance of the software.

When the software is modified then it is required to test the software fully or partially. For the large software, fully retesting of the software is very time consuming so instead of re-executing the entire test suite, it is better to select part of test suite to be run. To select the test case for the software from the entire test cases is known as selection problem for software testing.

In this report, we are giving the solution for selecting the test cases from the set of large test cases. We are using the genetic algorithm for selecting the test cases for object oriented program.

Following this introductory Section 1, Section 2, gives the brief introduction to testing, define the types and approaches used by testing, regression testing and genetic algorithm. Various operators and steps of genetic algorithm are explained. The importance and necessary issues of GAs are described also explains the advantages of genetic algorithm. How to calculate coverage of the object oriented program. The tools used for the pro- posed approach. It gives the brief idea about the tool. Section 3, provides the complete description of the proposed approach. Complete procedure of the generation of test cases and test suite than reduce the size of test suite by using genetic algorithm. Section 4, provides the conclusion and future work in the related area.

II. PROPOSED APPROACH

The whole proposed approach divided into two stages:

1. Firstly, we generate the test cases for object oriented program using genetic algorithm and making test suite.

2. Secondly, we minimize the size of test suite by applying genetic algorithm.

Generation of Test Cases

The steps for generating test cases are given below:

1. The java source code is given as the input to the Branch Instrument or module.

2. The Branch instrument or module produces the instrumented version of the original class and perform the following tasks on the class under test (CUT).

(a) Parsing the source code: Reads the source code of the CUT and classifies each statement (i.e., determining its type).

(b) Determines whole information about the constructor, method signature.

(c) Now, all the method of the given source file stored in an array.

3. Initially, the population or chromosomes are initializing randomly by binary value (either 0 or 1) with the random generator.

4. In the chromosome, 1 shows that particular method consider while testing and 0 shows that particular method not considering while testing.

5. The fitness value for each chromosome is calculated as:

(a) The test case generator takes each chromosome and decoded chromosome and converted into constructor and method invocations.

(b) In the test case the assertions are added. These assertion has two fields first one is expected value and the second one is value generates by the method.

(c) The test case is executed using JUnit tool. The JUnit execute the test case and find whether the test case passed or failed.

(d) If the test case executed by the JUnit is passed successfully then the coverage of the program is calculated by EMMA tool. The coverage value assign as the fitness of the chromosome.

1. Now, we select the chromosome by using one of the selection methods that is roulette wheel selection technique.

2. If the number of iteration completed then print the population and we stop the algorithm. Else the next generation is calculated by using selection, mutation and crossover and again go to step 5.

3. The final population shows the test cases. Now, combine all the test cases to make a test suite and calculate the fitness of the test suite.

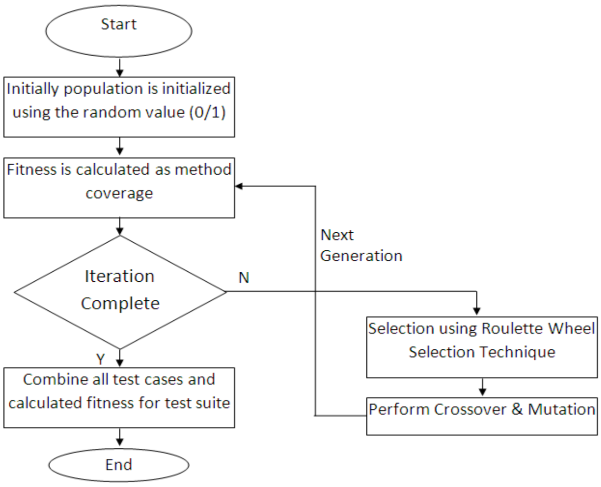

Start
Initially population is initialized using the random value (0/1)
Fitness is calculated as method coverage
Iteration Complete
N
Y
Next Generation
Selection using Roulette Wheel Selection Technique
Perform Crossover & Mutation
Combine all test cases and calculated fitness for test suite
End

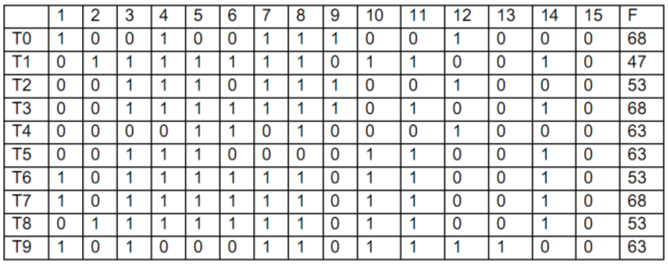
Figure 8: Flow Chart of Test Case Generation.

[Neelam Gupta, Aditya P. Mathur, and Mary Lou Soffa, *"Automated test data generation using an iterative relaxation method"*, In ACM SIGSOFT Sixth International Symposium on Foundations of Software Engineering(FSE-6), pages 231-244, Orlando, Florida, November 1998]

III. EXPERIMENTAL RESULT & DISCUSSION

Below table 1 shows the result for test case generation with population size = 10 and iteration = 10:

Table 1: The generated test cases with fitness value.



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	F
T0	1	0	0	1	0	0	1	1	1	0	0	1	0	0	0	68
T1	0	1	1	1	1	1	1	1	0	1	1	0	0	1	0	47
T2	0	0	1	1	1	0	1	1	1	0	0	1	0	0	0	53
T3	0	0	1	1	1	1	1	1	1	0	1	0	0	1	0	68
T4	0	0	0	0	1	1	0	1	0	0	0	1	0	0	0	63
T5	0	0	1	1	1	0	0	0	0	1	1	0	0	1	0	63
T6	1	0	1	1	1	1	1	1	0	1	1	0	0	1	0	53
T7	1	0	1	1	1	1	1	1	0	1	1	0	0	1	0	68
T8	0	1	1	1	1	1	1	1	0	1	1	0	0	1	0	53
T9	1	0	1	0	0	0	1	1	0	1	1	1	1	0	0	63

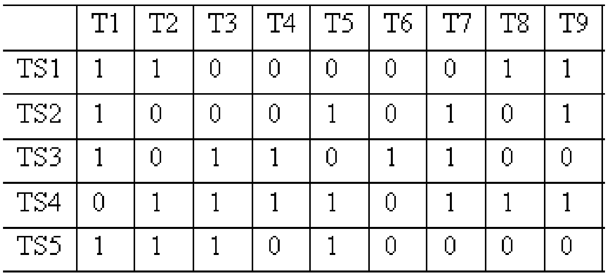
Now, we find the test suite by combining all the test cases generated above.

Table 2: The test suite.

	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	95

Below table 3 shows test suite that are minimized by genetic algorithm:

Table 3: The minimized test suite.



	T1	T2	T3	T4	T5	T6	T7	T8	T9
TS1	1	1	0	0	0	0	0	1	1
TS2	1	0	0	0	1	0	1	0	1
TS3	1	0	1	1	0	1	1	0	0
TS4	0	1	1	1	1	0	1	1	1
TS5	1	1	1	0	1	0	0	0	0

The minimized test suite found as TS5 with size = 5 contains test cases T1, T2, T3, T5, T10.

[Roy P. Pargas, Mary Jean Harrold, and Robert Peck, *"Test-data generation using genetic algorithms: Software Testing"*, Verification & Reliability, 9(4): pp. 263-282.]

IV. CONCLUSION

In this thesis, we have primarily focused on test case selection of object-oriented soft- ware automatically. We have also explored the technique and application of evolutionary algorithm like genetic algorithm to the automate the selection of test case for testing.

Our approach is suitable for object oriented program; however it requires improvement for large and complex program with real world problems. We works primarily on test case selection and its automation process but it is not fully automated as we have to insert assertion manually and the initially sequence of chromosome given manually and hence further development is essential.

V. REFERENCES

B. Beizer (1990). *Software Testing Techniques*, Van Nostrand Reinhold, 2nd edition.

C. Ramamoorthy, S. Ho, and W. Chen (1976). *"On the automated generation of program test data"*, IEEE Trans. Software Eng, vol. SE-2, no. 4. PD. pp. 293-300.

J.W. Duran and S. Ntafos (1984). *"An Evaluation of Random Testing"*, IEEE Transactions on Software Engineering, 10(4): pp. 438-444.

L. Clarke (1976). *"A System to generate Test Data and Symbolically Execute Programs "*, IEEE Trans. Sofrware Eng., vol. SE-2, no. 3, pp. 215- 222.

Neelam Gupta, Aditya P. Mathur, and Mary Lou Soffa (1998). *"Automated test data generation using an iterative relaxation method"*, In ACM SIGSOFT Sixth International Symposium on Foundations of Software Engineering(FSE-6), pages 231244, Orlando, Florida, November.

R. M. Poston (1996). *Automating Specification-Based Software Testing*, IEEE Computer Society Press.

Roy P. Pargas, Mary Jean Harrold, and Robert Peck (1999). *"Test-data generation using genetic algorithms: Software Testing"*, Verification & Reliability, 9 (4): pp. 263-282.

Corresponding Author

Vineeta Mishra*

Department of Computer Science & Engineering, Integral University, Lucknow

E-Mail – winihill@gmail.com