Approaches for Generating Test Cases Automatically to Test the Software

Paramjit Kaur, Rupinder Kaur

Abstract - Testing the software is very important phase in software development life cycle. So to test the software automatically is the best way to test the software because it consume less time where testing software manually is consuming process. To test the software automatically, test case generation is the best way. One way to generate the test cases is with the help of UML diagrams. In this paper we study the various techniques to generate the test cases from the UML diagrams to test the software automatically. Different tools are used to generate the UML diagrams. A new method is also proposed which will help in testing software automatically by generating test cases. Techniques in which different approaches are used like model based approach, novel approach, approach in which genetic algorithm is implemented as data mining technique are used to generate the test cases automatically, function minimization technique used for finding the minimum predicate function. The different tools like AGEDIS, TGV and GOTCHA are used for test generation. These approaches have been proposed to bring out all possible test cases of given object diagram.

Index terms: object diagrams, test cases, UML, genetic algorithm, data mining, tabu search algorithm.

I. INTRODUCTION

Software testing is an important activity in the software development because software has played an important role in our lives both socially and economically. While developing the software, the software organizations spend near about 50% of their budget in testing related task. It provides the efficiency of the software and the correctness of the software. Testing can be done either manually or automatically. Testing automatically is best way to test software because it consume less time and give accurate result than manual testing. One way to test the software is by generating the test cases which is a common approach. After generation of test cases, a particular technique is applied on it. Test data can be designed either manually or automatically. Test case are also generate from the UML using object diagrams. UML is unified modeling language that is used to create visual models of a software system. These models can help to create designs and to permit analysis and review of these models. There are different types of diagram in the UML. Test cases are generated through UML diagrams. Different techniques are used which uses different approaches to generate the test cases from object diagram.

All the possible test cases are generated to find the optimal result. Approach like model based approach which uses the genetic algorithm’s crossover technique which is useful to generate the test cases after the completion of the design phase and error could be detected at an early in stage in software development life cycle, novel approach in which tabu search algorithm is used which obtain the possible valid test cases, another novel approach uses genetic algorithm which help to improve the design quality, find faults and reduce software development time. The test cases are generating from the sequence diagram using sequence dependence table (SDT). Different tools for test generation are used like AGEDIS. TGV, GOTCHA are used for developing the test cases automatically to test the software in efficient manner and find the bugs from it in less time. All these techniques are used to test the software automatically by generating test cases.

II. RELATED WORK

This section discusses the related work done on generating the test cases automatically to test the software. Different techniques are considered with different method to generate the test cases. Each of techniques used have is own advantages and disadvantages. By applying different technique optimal result is obtained.

M.Prasanna et al. (2009) presents that to test the software, test cases generation is best way. The test cases are derived by analyzing the dynamic behavior of the objects due to internal and external stimuli [5]. Test cases can be generated with the help of UML diagrams. Researchers use model based approach in which genetic algorithm’s crossover technique is apply on the class diagram and the traversal is done by the depth first search(DFS) algorithm. This tree structure approach coupled with genetic algorithm shows that it is capable to reveal 80% faults in unit level and 88% faults in integration level. They couple the genetic algorithm with mutation testing to check the effectiveness in the testing process which shows 80.3% of effectiveness. The result shows methodology is useful to generate test cases after the completion of the design phase and error could be detected at an early stage in software development life cycle.

A.V.K Shanth et al. (2011) have proposed another model based approach in which the concept of data mining is used in which the evolutionary genetic algorithm technique is apply on the class diagram and generate the test cases [7]. Researcher study that Data mining technique is implemented to generate the optimal test cases automatically by which human and cost efforts are minimized [7]. They show that evolutionary genetic algorithm yields optimal valid test case than with only genetic crossover operator, after applying
depth first searching algorithm. The advantages are that specification-based testing uses information derived from a specification to assist testing as well as to develop programs. The model-based approaches are mainly used to generate test cases by applying the different techniques and produce the optimal result from them. Since test cases are obtained is valid one so it is not mandatory to evaluate manually.

G.Mohan Kumar, A.V.K.Shanthi (2012) researchers used some novel approaches to test the software at the initial stage itself which will easy for the software testers to test the software in the later stage [12]. In their study they use the tabu search algorithm to generate the test cases automatically from the object diagram. Here they take the sequence diagram. The experiment results show that this method has better performance. All the possible test cases are generated and validated by prioritization. test cases which are generated can be used as test suite for path testing for application. This approach can reveal all path’s for software to be developed and also obtained test cases valid once, which avoids validation test case because of fitness criteria [12].

Sangeeta sahbwal, Ritu sibal, Chayaniaka Sharma (2011) consider in their paper another novel based approach in which testing efficiency is optimized by applying the genetic algorithm on the test data. For requirement change, a stack based approach for assigning weights to the nodes object diagram is proposed [8]. Here the object diagram that is considers are activity and state chart diagram. In this they convert first activity diagram into CFG and state chart diagram into SDG and then stack based approach is applied. The test paths are generating from the activity diagram and state chart diagram. In this novel approach the research uses the genetic algorithm which is apply on the sequence diagram. In the first sequence diagram is generated and then from the sequence diagram. Sequence dependency graph is generated and genetic algorithm is apply on it. In their study the found that the approach used is significant to identify location of the fault in the implementation and thus reduce the testing efforts [13]. The proposed approach makes use of IF model and genetic algorithm to find the path to be tested first.

Ranjit Swain, Vikas Panthi, Durga Prasad (2012) in this paper different techniques are used to generate the test cases to test the software. The functional minimization technique is also used to generate the test cases. In this technique in which the STUPEC [11] technique is used in which first predicate is selected and then predicated is transformed and then test cases are generated. The functional minimized technique is used for finding the minimum of predicate function. In this approach the test cases are generated step by step. Here the object diagram that is used for generating the test cases is state machine diagram. This approach covers much coverage like state coverage, transition pair coverage, action coverage. The numbers of test cases are minimized that achieve transition path coverage by testing the borders determine by simple prediction. It is found that test cases are generated from the object diagram by minimize the cost and time [11]. It can also handle transitions with guards and achieves transition path coverage.

L.C.Briand el al. (2008) in this paper approach the researcher proposed the method supported by the prototype tool to tackle the regression test selection problem at the design level in the context of in many cases automation is likely to help avoid human errors. The main objective has been to ensure that regression testing was safe while minimizing regression efforts. But they show that certain changes may not be visible in the design and may require additional attention during coding or special way to document them design. Another limitation is that, based on UML design information, test selection may not be a precise as if it was based on detailed code analysis. The case study that is considered has only one case of imprecision in classifying a test case as retesting.

Alessandra Cavarra, Thierry Jeron, Alan Hartman ISSTA (2002) present an architecture for model-based testing using a profile of the unified modeling language (UML). Class, object and state diagrams are used to define essential model. To generate the test cases automatically, different types of tool are available like AGEDIS test generation tool is used [4]. The AGEDIS test generation tool is based on the principles of two exiting tools that are TVG and GOTCHA. The main advantage of the AGEDIS test case generation tool is its ability to combine different test directives: coverage criteria, test purposes and test constraints. This allows the user to tune the selection of test cases with respect to the budget of the campaign [4]. A hierarchy of test suites can be constructed with the help of property that the larger the suite, the greater the coverage of the implementation. This hierarchy is particular useful in regression testing.

III. PROPOSED WORK

In this paper a new method is proposed in which the genetic algorithm’s crossover technique is applied to bring out all possible test cases. This technique will be applied on the sequence diagram. This method is coupled with mutation testing to check the effectiveness of the methodology. The traversal is done with the help of the depth first search. The work is done with class diagram which show that tree structure approach coupled with genetic algorithm reveal 80% faults in unit level and 88% faults in integration level. They couple the genetic algorithm with mutation testing to check the effectiveness in the testing process which shows 80.3% of effectiveness. The result shows methodology is useful to generate test cases after the completion of the design phase and error could be detected at an early stage in software development life cycle. It should be consider that new method show better result the previous work done.

STEPS FOLLOW IN THE PROPOSED WORK

Following are the steps that should be following to generate the test cases.

1) Using rational rose software construct a sequence diagram and store with .mdl as extension.
2) Capture the object names by parsing the .mdl file.
3) Build a tree using object names and apply genetic algorithm’s crossover technique.
4) Then convert new generated trees into binary trees. Traversal is done by depth first search method of binary trees.
5) All the valid, invalid and termination of the application can be obtained using step 5.

All these steps are following in the sequence manner and result will obtain.
Flow chart of methodology

![Flow chart of methodology](image)

**Fig.1 Flowchart**

**Fig1. Show the flowchart of the proposed method.**

**DESCRIPTION OF METHODOLOGY**

First of all the sequence diagram is drawn with the help of rational rose tool which will show the snapshot of the detailed state of the system at a point in time. We can also use another tool for drawing object diagrams. After drawing the sequence diagram, save it with extension .mdl. Then object names are capture and then new tree is formed on which genetic crossover technique is applied. The tree is formed in such a manner that the objects represents the root node and place vertical one after one and attributes are arranged in left branch and methods are arranged in right branch of tree. After applying genetic algorithm, binary tree is form by arranging nodes in left of root node of the root node in vertical order and arrange its sibling in the horizontal order. Then traverse the binary tree by depth first search. It will give all the valid, invalid and termination sequences for the given application.

**IV. CONCLUSION AND FUTURE WORK**

In this paper we see that techniques that are used for generating the test cases from the UML diagrams developers to improve the design quality find faults in the implementation early and reduce software development time. In this paper new method is also proposed in which genetic crossover algorithm is applied on the sequence diagram which can help in find more faults and increase the effectiveness through mutation testing. Techniques used show the result that methodology is useful to generate test cases after the completion of the design phase and errors can be detected at an early stage in the software development life cycle. In future work it should be consider how to generate test cases automatically through object diagram to obtain optimal result and implementation of proposed work to generate the test cases automatically to test the software consider to be give better result.

**REFERENCES**


**AUTHOR PROFILE**

**Paramjit kaur**, Research Scholar, done M.Sc(CS) from GNDU, now doing M.Tech (CSE) from Lovely Professional University, Phagwara, Punjab, India, Research Area is Software Engineering.

**Rupinder kaur**, Assistant Professor in Department of CSE, Lovely Professional University, Phagwara, Punjab, India, done B.Tech from Baba Banda Singh Bahadur Engg. College Fatehgarh Sahib(2010), M.Tech from University Institute of Engg & Tech., Punjab University(2012), Research Area is Software Engineering.