Optimize Testing Process after Generating the Behavioral Representation of System

Gurbakash Phonsa
Department of Computer Science and Engineering
Lovely Professional University, Phagwara

Kapil Kumar Bansal
Deptt of Computer Science and Engineering,
SRM University,
Noida, - India.

Rajwinder Sidhu
Department of Computer Science and Engineering, Lovely Professional University,
Noida, - India

ABSTRACT
The generation of test cases is the most challenging part of software testing phase for the software tester. Test cases are such that as it unable the detection of errors. To determine how much software testing is enough is a difficult job. Testing is advisable to identify the risk against the cost of additional testing effort. To assign priorities to test cases will guarantee that the main tests have been done. The main objective of software testing is to identify defects and correcting those defects improves Software Quality. This paper represents a noble approach to automatic generation of test cases. In this approach, UML sequence diagram is the source to generate the test cases by convert it to sequence diagram graph to further generate the test paths. Then the test paths are prioritizing by using the Genetic Algorithm along with Tabu Search Algorithm.

Keywords
UML, Sequence Diagram, Genetic Algorithm, Tabu Search algorithm, Software Testing, Test case generation.

1. INTRODUCTION
Software Engineering is the process of design, test, implement, operation and maintenance of the software by using the systematic and quantifiable approach. Software Engineering is a layered structure consists of process, methods and tools in order to control the process of software development [12]. It also concerned with various ways in which to produce operational software in expected costs and time. Software development life cycle process is used to develop the software. The verification and the validation of the developing software can be examined by the software testing phase. Software testing is crucial as correctness, completeness and quality of produced software is identified. The objective of the testing is to spot errors causing faults and resulting failures of software.Thus, software testing is an approach of finding errors in softwareby executing the software or the system in order to fix the software before the final delivery of the product. Test Cases are implemented in order to test the behavior of the system under a specified functionality. Hence, test casegeneration is essential to test the processes test cases in a particular software. Sketch diagram basedtechnique, Specification-based technique and Source code-based technique are widely known test case generation techniques[1]. Sketch diagram-based techniques are used to generate test cases from UML model such as Use Case diagram and State diagrams [6]. This technique leads to automatic genaration of test cases and saves time and resources.

UML is a design language used to model an application structures, behaviors and business processes. It provides the special facility to developers to develop and build computer applications. Thus UML specifies, construcst and document the system [3] [5]. Functional Model provides the system's functionality from the user's viewpoint. Use case diagrams fall under this category of Functional Model. Object Model includes Class Diagram and is used for defining the system structure by using the objects, functions, member data, operations and associations. Dynamic Model reveals the internal system behaviour [8]. It includes Sequence Diagram, Activity Diagram and State Machine Diagram. Sequence Diagram represents the internal behaviorial aspects of the system. The diagram is used to display how object interactions under given circumstances. Sequence diagram is an interaction diagram; illustrating the interaction among the processes, objects by means of transfer messages from one object to another with respect to time. Sequence diagram meke use of top to bottom approach from top to bottom with given time span. Objects interacts with one another by sending messages to each other; it invokes an operation of that particular object. As soon as, the object receive the message,the invoked operation, begins to execute [4].

2. OBJECTIVES
Decrease the effort and time by automated generation of test cases.

Generate the optimal Test case for the path testing with the optimization algorithm.

To improve the design quality of the system by implementing the optimization algorithm approach.

Use the hybrid optimization algorithm approach to generate test case from UML sequence diagram.

3. SCOPE OF STUDY
In the software development life cycle, testing phase is the most vital phase to detect the bugs in the software before final delivering of theproduct to the customers. By this approach, test case prioritization can be achieved based on the path testing which definitely reduces the factors such as cost and recourses. Further, it also increases the reliability and usability of the developed system. To generate the required and most vital test cases from thesequence diagram also improve the efficiency.

4. METHODOLOGY USED
A sequence diagram is converted to sequence representation graph. In SRG, nodes and edges are named as S1 to Sn and e1 to em respectively [6].

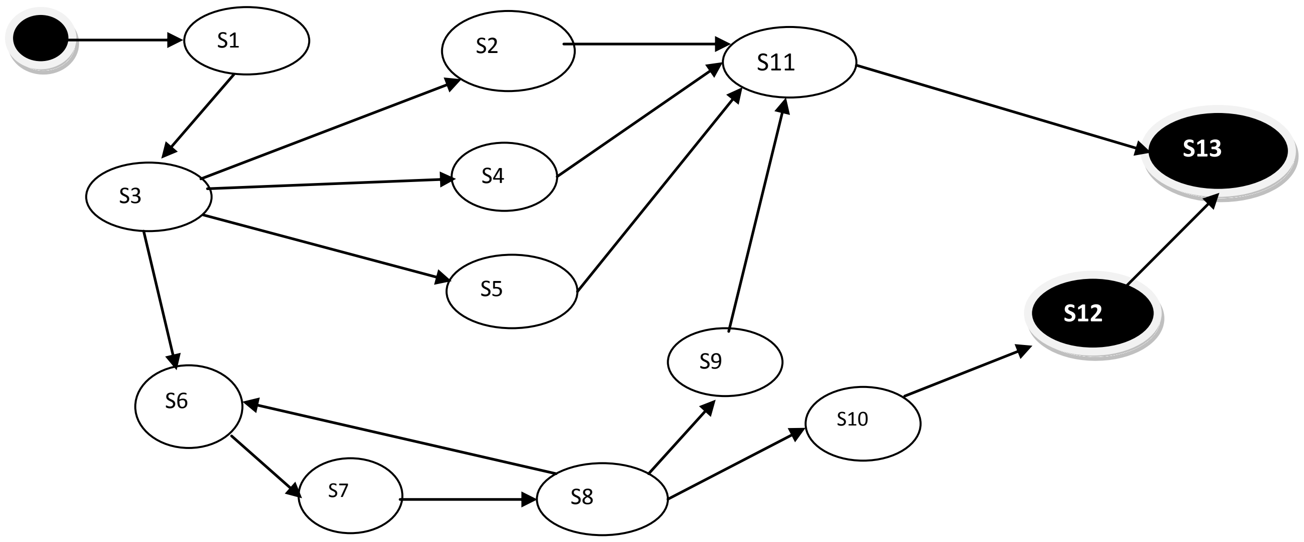

Fig 1: Graph for all possible paths with termination point

Table 1.Stack based weight assignment to nodes of SRG.

Nodes	**K(stack)**	**Size(s)**	**Weight=S (max)-k**
S13	8	9	9-8=1
S11 or S12	7	8	9-7=2
S9 or S10	6	7	9-6=3
S13 or S8	5	6	9-5=4
S13 S11 or S11or S7	4	5	9-4=5
S11 S4 or S5 or S6	3	4	9-3=6
S2 or S3	2	3	9-2=7
S1	1	2	9-1=8
S0	0	1	9-0=9

Further the Stack based complexity can be generated and find out the complexity for each node in a SRG.

Find out the decision node by looking to SRG graph mention above:
Three decision nodes are in SRG which are S1, S3 and S8

Convert it into 2 bit format

Like as:	2bit format
S1= e1, e2	00, 01
S2=e5, e6, e7	00, 01, 10
S8=e12, e13, e14	00, 01, 10

Table 2. Table for decision nodes

S.NO	**S1**	**S3**	**S8**
1	E1	E5	E12
2	E1	E5	E13
3	E1	E5	E14
4	E1	E6	E12
5	E1	E6	E13
6	E1	E6	E14
7	E1	E7	E12
8	E1	E7	E13
9	E1	E7	E14
10	E2	E5	E12
11	E2	E5	E13
12	E2	E5	E14
13	E2	E6	E12
14	E2	E6	E13
15	E2	E6	E14
16	E2	E7	E12
17	E2	E7	E13
18	E2	E7	E14

5. PROPOSED CHANGE

The existing approaches mention in various research papers used on the activity diagram, state diagram, sequence diagram and applied different complexity based upon BASICIF, STACKBASED and CYCLOMATIC COMPLEXITY. The optimization Genetic Algorithm is applied to find out the optimal test case [5]. To solve the problem of loop condition TABU SEARCH algorithm is applied to find out the test case[7] which is optimal for testing in path testing [2]. The UML sequence diagram is used by replacing the activity diagram and further applying the technique mention in [3] to convert it into Sequence Diagram Graph. After that the total complexity is generated by addition of Stack Based complexity and IF complexity. The last step is to combine the optimization algorithm like Genetic algorithm and Tabu Search algorithm.

The GA has the three main operation on population is as follow:

```
        Population Initialized
        Population Evaluate
While (condition not satisfied){
        Selection
        Crossover
        Mutation
        Evaluate        }
```

The Selection operation generates the new population from the old population based on the fittest value to better select the chromosomes from the population. The Crossover operation swaps the sequence of bits in the string into two individuals. The Mutation operation is applied which alters chromosomes in small ways to introduce new superior traits.

The TABU SEARCH has the three variables tabuin, tabuout and best. In the best variable, store the calculated initial test path value as well as store the path in tabuin. Further calculate the neighborhood test path value and compare it with value stored in best variable. Until the current value is greater than the best; do calculate the neighborhood value; if it is greater than store the current value in the best and the current path to the tabuin. Repeat until all path have been covered to get the prioritized test path.

Table 3. Test case representation using two bit format:

Number	e1	e2	e3
1	00	00	00
2	00	00	01
3	00	00	10
4	00	01	00
5	00	01	01
6	00	01	10
7	00	10	00
8	00	10	01
9	00	10	10
10	01	00	00
11	01	00	01
12	01	00	10
13	01	01	00
14	01	01	01
15	01	01	10
16	01	10	00
17	01	10	01
18	01	10	10

Finally the optimization algorithms can be applied such as Genetic Algorithm [9] [10] and TABU SEARCH Algorithm [2] and compare the result to existing approaches.

Steps for implementation

1. Draw sequence diagram.
2. Convert it into SRG.
3. Generate Stack based complexity & IF complexity.
4. After that find out the total complexity.
5. Apply optimization algorithms on generated test cases.

The next step is to find out the Basic IF complexity [5] which can be done as:

IF= fan in (a) * fan out (a)

Where a is current node in SRG. The total complexity can be generated through the addition of STACKBASED and BASICIF complexity [5].

Table 4. Find the total complexity of each node.

Node	Complexity based on pop op. , (A)	Fan in(a) X Fan out(a), (B)	Total complexity =(A+B)
S0	9	0	9
S1	8	2	10
S2	7	1	8
S3	7	3	10
S4	6	1	7
S5	6	1	7

S6	6	1	7
S7	5	1	6
S8	4	3	7
S9	3	1	4
S10	3	1	4
S11	6+5+2=13	4	17
S12	2	1	3
S13	5+4+1=10	0	10

These steps followed by us research representation. Steps are the essential for the problem formulation. The graph is prepared from the behavior representation of the system. This graph is used to generate the complexity on the particular path through the node involved in scenario. After individual complexity total complexity is calculated for the whole path. Once the complexity is known on each path, then the challenge is how we can optimize these case in an efficient way. The answer to this problem is the optimization techniques available. But we have to choose the technique which is best suited. We had studied the various technique and then conclude the best one which is not only good, but produce the efficient result in proposed scenario. While applying this technique we include some new nodes.

The below diagram is shows the context of the problem and the formulation to pursue the proposed scenario.

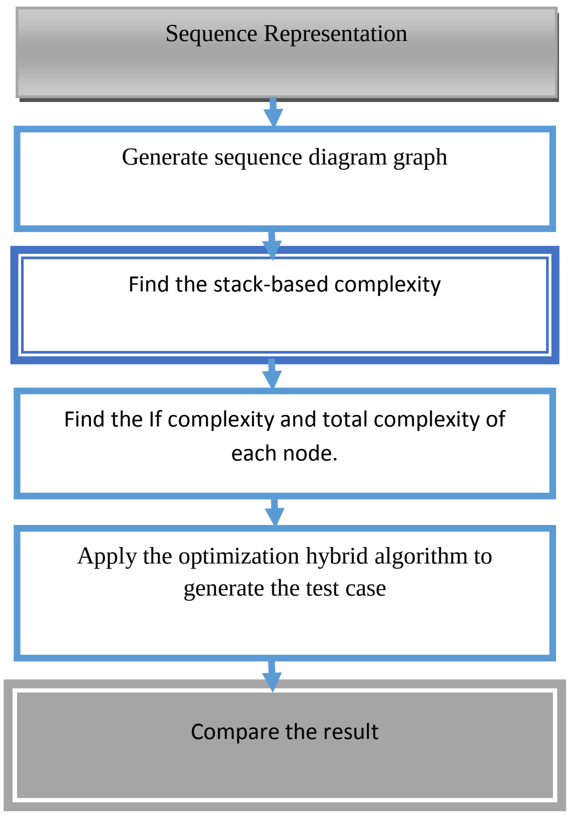
Sequence Representation
Generate sequence diagram graph
Find the stack-based complexity
Find the If complexity and total complexity of each node.
Apply the optimization hybrid algorithm to generate the test case
Compare the result

Fig2. Representation of steps to implement

Finally applying the hybrid approach which is used to generate the test cases from the sequence diagram and prioritize the test case which provides the optimal test cases. Hence, there are various search optimization techniques which are best able to find out the optimal solution of the problem such as Calculus Based Techniques, Random Search Techniques and Enumerative techniques [6] [11].

Figure 1 depicts the steps used in the methodology mentioned above.

6. CONCLUSION

In this paper, we represented the automatic test case approach and have a brief look to each step. The input of this approach is the UML sequence diagram which converted to sequence diagram graph. Further, the description is of how to prioritize generated test cases using optimization algorithms. We implemented the GA algorithm along with Tabu search optimization algorithm. Hence, our approach to propose an algorithm that basically can be used to automate the process of testing. This approach is used to support the path testing for generating the automatic test cases. In future, the automatic tool based upon this approach is possible to build. This automatic tool will reduce cost of software development and improve quality of the software.

7. REFERENCES

[1] Sharma, N. K., and DivyaSaxena. Study of Approaches For Generating Automated Test Cases By UML Diagrams.

[2] Shanthi, A. V. K., and G. MohanKumar. "A Novel Approach for Automated Test Path Generation using TABU Search Algorithm." International Journal of Computer Applications 48 (2012).

[3] Sarma, Monalisa, DebasishKundu, and Rajib Mall. "Automatic test case generation from UML sequence diagram." Advanced Computing and Communications, 2007. ADCOM 2007. International Conference on. IEEE, 2007.

[4] Malhotra, Ruchika, and Abhishek Bharadwaj. "Test Case Prioritization Using Genetic Algorithm." International Journal of Computer Science and Informatics ISSN (2012): 2231-5292.

[5] Sabharwal, Sangeeta, RituSibal, and ChayanikaSharm. "Applying Genetic Algorithm for Prioritization of Test Case Scenarios Derived from UML Diagrams." International Journal of Computer Science Issues (IJCSI) 8.3 (2011).

[6] Keyvanpour, Mohammad Reza, HajarHomayouni, and HosseinShirazee. "Automatic Software Test Case Generation: An Analytical Classification Framework." International Journal of Software Engineering & Its Applications6.4 (2012).

[7] Sharma, Anjali, and Maninder Singh. "Generation Of Automated Test Cases Using UML Modeling." International Journal of Engineering 2.4 (2013).

[8] Sumalatha, V. Mary. "Object Oriented Test Case Generation Technique using Genetic Algorithms." International Journal of Computer Applications 61.20 (2013).

[9] Malhotra, Ruchika, and Abhishek Bharadwaj. "Test Case Prioritization Using Genetic Algorithm." International Journal of Computer Science and Informatics ISSN (2012): 2231-5292.

[10] Doungsa-ard, Chartchai, et al. "An automatic test data generation from UML state diagram using genetic Algorithm." Proceedings of International Conference on Software, Knowledge, Information Management and Applications (SKIMA). 2006.

[11] Srivastava, Praveen Ranjan, and Tai-hoon Kim. "Application of genetic algorithm in softwaretesting." International Journal of software Engineering and its Applications 3.4 (2009): 87-96.

[12] "IEEE Standard Glossary of Software Engineering Terminology," IEEE std 610.12-1990, 1990.