A Review: Study of Test Case Generation Techniques

Itti Hooda
Research Scholar
M.D.U. Rohtak, Haryana

Rajender Chhillar, Ph.D.
HOD- Computer Science and Applications
M.D.U. Rohtak, Haryana

ABSTRACT

Software testing is very labor intensive task for developing software and improving its quality. According to some researchers and software professionals,50% of the time, cost and effort are spent on software testing.To test software, generating test cases is the most important task.Testing can be done either manually or automatically by using various testing tools. In today’s scenario software are testedautomatically with the help of tools as it is a fast and accurate process of testing software. Although various testing tools are available in market and are used by testers to test the software and to generate test cases and test data automatically.There are various techniques available for generating test cases like fuzzy logic, finite state machine, neural networks, genetic algorithms, soft computing, genetic programming, evolutionary computation and many others.This paper presents various test case generation methods, test case minimization, selection, and prioritization and evaluation techniques.This paper also focuses on various test case prioritization and selection techniques that help the test engineers to schedule and rank the test cases to reduce the total effort, time and the cost.

Keywords

Test cases, UML Diagrams, GeneticAlgorithm,Software under Test (SUT), Extended Finite State Machine(EFSM)

1. INTRODUCTION

Software testing is most essential and integral activity in software development life cycle. This activity is performed by using a systematic approach .The testing process is a method of executing a program on a set of test cases and comparing the actual results with the expected results. Test cases are usually derived from software artifacts such as specifications and design. Testers initially go through the requirement document to understand requirements and specifications. After this is done, they start preparing test cases using test case template. A basic test case template should have details like Test Steps, Test Data, Expected Result, Actual Result, Pass/Fail and comments. It should have other details like Pre-conditions, assumptions, requirement numbers, date, tester name etc. Test Cases are generated automatically using testing tools OR testers have to manually write test cases. There are various Test Case Generation techniques used to generate test cases. The two main approaches are given [R.Mall] to automate the test cases. The first approach is to designing the test cases from requirements and design specifications. The second approach is to design test cases usingcode. Although generating test cases from coding is quite complex task therefore the first approach is mostlyadopted. This process also helps the test engineers in findingand analyzing the problems and faults with the designed system. The purpose of this paper is to study the test case generation techniques by using the various existing techniques. The various phases of test case life cycle are Test case generation, test case selection, test case minimization, test case prioritization and evaluation.

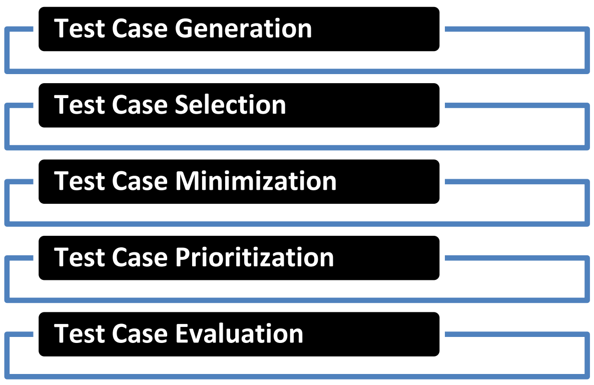

Figure 1 Test Case Life Cycle

This paper is divided into four sections. The first section depicts on various test case generation techniques using various UMLdiagrams, test case generation based on GUI,goal driven and others techniques for generating test cases. The second section focuses on the test case selection and minimization. The third section describes the test case prioritization& evaluation.Finally the fourth section summarizes the conclusion for reflecting the acceptance and relevance of these techniques and the future work.

2. SOFTWARE TEST CASE GENERATION TECHNIQUUES

The total cost, time and effort required for overall testing depends on total number of test cases.A test case is a set of inputs given to the software or application to get the pre-specifiedoutput. The effort basically depends on the size of the application and the number of test cases.R.P. Mohapatra and Jitendra Singh describe the step by step method for test case generation technique.

1. The first step is to find all possible constraints from start to finish nodes. A constraint is a pair of algebraic expressions which dictate conditions of variables between start and finish nodes.
2. To reduce the test cases, the highest value is assigned to the variable having maximum value and the lowest value is assigned to minimum value within its specified range.
3. After this, the constant value is assigned to the given variable at each node in the defined path.
4. Finally table is created that includes all possible test cases.

Leung and White categorizetest cases into five classes as reusable, retest able, obsolete, structural, new specification and new structural test cases.

Reusable test cases only execute the parts of the program that remain unchanged between two versions. Retestable test cases execute the parts of a program that have been changed in another program. Obsolete Test Cases can be rendered obsolete because their input/output relation is no longer correct, due to changes in specifications and they don't test what they were designed to test due to modifications in the program.

There are some methods of test case generation that depends on the application like test case generation for web application, object oriented application, structured based systems, UMLapplications, applications based on evolutionary and genetic algorithms and many others. Software is tested by using some set of inputs and its success depends on the expected outputs derived from the test case conditions.Throughout the years, several different testing have been proposed for generating test cases.

2.1 UML Diagrams

Test cases can be derived from use cases and also be derived from system requirements.Samuel et al [1] presented a novel technique of test case generation based on UML sequence diagrams. They used the edge marking dynamic slicing method which is applied on MDG(Message Dependence Graph) for creating the slices. These slices help in prediction of sequence diagram by which the test data is generated.Some use case models arealso used for generation of test cases. The behavioral and interaction diagram based test case generation method in which various use case diagrams are used. The activity diagrams are used to generate the test case scenarios. Use case Diagram Graph (UDG) is a graph for deriving test cases that shows all the possible use cases in the SUT. Baikunth proposes the method of generating the test cases for object oriented programs from UML collaboration and activity diagrams. To find the fittest (sub -optimal) test cases, genetic algorithms is used that satisfies the test case adequacy criteria.The adequacy criteria is used to show the adequacy of the test case related to statement coverage, branchcoverage, path coverage, condition coverage etc in terms to find the faults in the program.

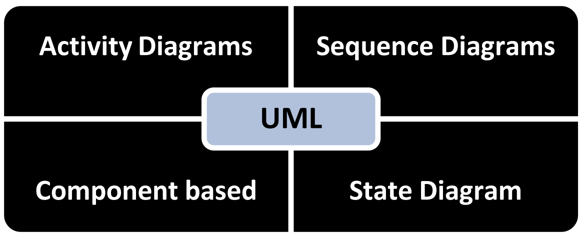

Figure 2 UML Notation Based Test Case Generation Techniques

2.2 Critical Path Method

T.Y.Chen et al[5] proposed a framework for generating test cases i.e. critical path method. His method helps the software tester in formation of test cases by refining functional choices.[27]Ratna Raju describes the designing of test cases,It is necessary to know the purpose of the test along with the hardware and software requirements and the description on how to perform the task. The success criteria depend on the expected and the actual results for the particular test.A test case generator "Auto Test" is also used to generate and run random tests for twenty seven classes from a usually used Eiffel Library.

2.3 Code based Test Generation

[A .Mathur] Test cases can be generated directly from the code which is called as code- based test generation. Thistechnique is used primarily while enhancing existing tests based on test adequacy criteria. This method supports regression testing to reduce the size of test suite, or prioritize tests. Four types of inputs are taken where the first input p' the program to be tested, the program p from which p' has been derived by making changes and existing test suite T for p and some run time information obtained by executing p against T. A fully automatic valid test cases generation scheme is given by using testing technique that satisfies the transition path coverage criteria [A.V.K Shanti].

2.4 GUI based Test Case Generation Technique

[Imran Ali Quershi and AamerNadeem] thoroughly presented a survey of test case generation methods that are primarily useful for GUI testing.

1. The first method of generating test cases for GUI responsibilities using complete interaction sequence
2. Finite state testing and analysis of graphical user interface.
3. On the test case definition for GUI testing.
4. Automation of GUI testing using a model driven approach.
5. Developing cost effective model based technique for gui testing.
6. A model-based approach for testing GUI using hierarchical predicate transition nets.
7. Generating test cases from the GUI model.
8. Using GUI run time state as feedback to generate test cases.

[Qureshi] describes all above test case generation methods for basically GUI applications and presents a clear picture about the functionality of each technique and also presents the comparison table in which each technique has been compared by using some evaluationparameters.

2.5 Dynamic Path Testing and Evolutionary Technique

It is a dynamic path testing technique that generates test cases by executing the program with different possible test case values. Korelalso proposed an approach of test data prioritization based on path testing. The test cases are prioritized according to the shorter path by applying random testing method[11].J.Wegener, A.Baresel and H.Sthamer focused on generating test cases by using several structural test coverage testing criteria using evolutionary approaches like Genetic Algorithm.Which generally use themeta heuristic search based evolutionary approach for test case generation [33]. They also propose the test case generation for C language by using random testing technique. This method is required at the time of coding a program. The UML diagrams are quite helpful in testing these types of software.

2.6 Graph Traversal Algorithm

Test cases can also be generated by traversing from parent root to child node following breadth first OR depth first in a graph or tree. When all the nodes in a path from parent to child node are traversed, then that is considered as one test case. All nodes should be covered to make sure all flows in an application are covered. One flow is considered as one test case.

2.7 Genetic Algorithm

The most recent research is being done on test case generation, minimization and evaluation using Genetic Algorithm. A GA is an optimization technique which can be applied to various problems. It uses survival of the fittest technique, where the best solution survives. The two main requirements of GA are (a) an encoding used to represent a solution of the solution space(b) An objective function that is a fitness function which measures the goodness of a solution. [40]

This includes two testing techniques i.e. mutation and crossover testing. Mutation includes mutants which are changed function to be applied on any logic or test case to generate new test case. Crossover testing is also used to generate test cases which involves crossing over of two different use cases or test cases to generate a new use case.

3. TEST CASE SELECTION AND MINIMIZATION

Test case selection is a method of selecting a subset of test cases from a test suite to reduce the time, cost and effort in software testing process. It is very much similar to test suite minimization technique. The test suite minimization technique is based on metrics like coverage measured from a single version of the program under test. The difference between these two techniques depends upon the changes made in SUT.The test cases are selected according to the changesmade between the previous and the current version of the SUT.

3.1 Model based Technique and Extended Finite State Machine (EFSM)

In Model based technique, the subset of the requirements gets modeled by using formal notations such as specifications of the subset of the requirements. Leon [9] presented an empirical comparison of four different techniques for filtering large test suites-test suite minimization, prioritization by additional coverage, cluster filtering with one cluster sampling, failure pursuit sampling. Rothermal[35]also define a technique for test suite minimization where the size of the test can be reduced by eliminating redundant test cases from the test suite. Therefore minimizationmethod is also called as test suite reduction.Leung and White present thefirst systematic approach to regression testing and test cases.Tallam et.al performed two types of reduction on the lattice.Lattice is a natural representation that supports the identification of the test cases. Test suite optimization problem is solved by implementing the model based test suite optimization technique.In this technique the evolutionary based algorithms are used for optimizing the test suite .Another technique used to reduce the total number of test cases are Extended Finite State Machine(EFSM) .It is basically used to reduce the regression test suites.

4. TEST CASE PRIORITIZATION AND EVALUATION

Test case prioritization is a method of scheduling and ranking the test cases from multiple test suites of software. There are many approaches to schedule and rank the test cases.Each and every test case is assigned some priority but sometimes there may be some issues arises when multiple test cases have the same priority or the weights. Sometimes problemoccurs in prioritizing these multiple test suites.There are two methods to overcome these issues.

4.1 Multiple Test Suite prioritization (MTSP) method

[Jiripong] describes the MultipleTestSuiteprioritization (MTSP) method, which is used to prioritize the test cases from multiple test suites. The entire program is divided into number of test suites and these test suites contains multiple testcases. The test cases are prioritized according to weight and rank that are used for testing the program.Rothermal[19] considered nine approaches for prioritizing a set of test cases and reported results. He also presented different approaches for revealing the faults to improve the software quality. Regression testing is the process of testing the software against those changes in the existing software.The four methods of regression testingare-reset method, regression test selection method, test suite reduction and test case prioritization method.Shin Yoo[38]presented various test case prioritization approaches that are based on some criteria.

The first approach is Distribution-based approach, in which the test case profiles are distributed based on the dissimilarity metric. Using this metric, the clusters of these test cases are prepared according to their profiles.The test cases having similar profiles get clustered into one redundant group of test cases and other groups indicate the unusual conditions that cause the test case failures.

The second approach is Human based approach which is based on Case-Based Reasoning(CBR) in which a Rankshoot algorithm is taken that selects test cases according to their ranks provided.Shin and Harman [38] gives some other approaches where the test cases get prioritized according to the probability of test case selection methods.The test cases are selected based on some factors like cost, length of test cases etc.The history based approach is associated with the clusters based on some pervious artifacts that are obtained by matrix analysis. Some test cases areprioritized according to the software requirements of the customers.

4.2 Model based approach

Theother approach is model based approach, where the relevant test cases are assigned into high and low priority test cases based onthe designed model.[Jiripong] describes an experiment’s design, measurement metrics and results in order to determine the most recommended test case prioritization method.

The test cases are evaluated to assess and compare the suitable test cases to test the software. To evaluate the test cases, the following test case prioritization techniques areused :(1) Prepare experiment data, (2) Run the test suites prioritization method, (3) Evaluate results.

Some measurement metrics are also used in this experiment are :(1) Percentage of high priority reserve effectiveness, (2) Size of acceptable test cases, (3) Total prioritization time.These methods help in finding the minimum number of test cases for testing software.The table of test case generation technique ranking is also shown in that paper based on these prioritization techniqueswith the help of Random approach,Hema’s approach, Alexey’s method, MTSSP and MTSPM methods of test case generation.

5. CONCLUSION

There are various test case generation techniques available, which may vary from application to application, could be based on specifications, design and programming language, however this paper described most prominenttest case generation techniques and alsoexplained test case life cycle phases like test case selection, minimization, prioritization and evaluation. Testing results solely depends of test execution and test execution is done based on test cases. If test cases could be optimized in such a way that test execution would be done in minimum amount of time with maximum coverage and find maximum bugs, then that would improve quality of software. This is only possible if testers optimize test cases in best available manner to generate most effective test cases, evaluate them and further minimize and select best test cases using a firm mechanism. The further research is required to implement most recent algorithm like Genetic Algorithm in combination of some other techniques like UML, traversal algorithms on Object oriented programming to optimize test case generation and to improve overall software testing process, which in turn will improve quality of software. A new software testing technique would be proposed after implementation of combination of Genetic Algorithm and other existing algorithms.

6. REFERENCES

[1] Philip Samuel, Rajib Mall, A Novel Test Case Design Technique Using Dynamic Slicing of UML Sequence Diagrams, e-Informatica: Software Engineering Journal, Vol 2, Issue 1, 2008.

[2] Chan, W.K.,Chen, T.Y., Tse, T.H.,An Overview of Integration Testing Techniques for Object Oriented Programs, In Proceedings of the Second ACIS Annual International Conference on Computer and Information Science(ICIS 2002), Mt. Pleasant, Michigan(2002).

[3] Mathur.A et al, Foundations of Software Testing:TestSelection,Minimization, and Proiritization for Regression Testing, Purdue University, October 25, 2010.

[4] QuershiT.Imran Ali, Nadeem Aamer, GUI Testing Techniques:A Survey, International Journal of Future Computer and Communications, Vol.2, No.2, April 2103.

[5] J.B. Goodenough and S.L.Gerhart, Toward a Theory of Test Data Selection, In Proceedings of the International Conference on Reliable Software, 1975, pp.493-510.

[6] Shukla.Upasana, Bharti. A.K, Gupta. D.N, Integration Testing of Object Oriented Component Using Program Slicing in the Detection of Equivalence Mutants, International Journal of Advanced Research in Computer Science and Software Engineering,Vol.3,Issue 1,ISSN:2277 128X,January 2013.

[7] McMinn.P, Search-Based Software Test Data Generation :A Survey, Software Testing Verification and Reliability, Vol.14, No.2,pp.105-156,2004.

[8] Mohapatra, R.P, Singh, Jitendra, Improving the Effectiveness of Software Testing through Test case Reduction, Word Academy of Science, Engineering and Technology 13, page 345-350,2008.

[9] Biswal, Baikuntha. Narayan et al,Test Case Generation and Optimization of Object Oriented Software using UML Behavioral Models, NIT,Rourkela, Orissa, India, July, 2010.

[10] Swain.Ranjita.Kumari, Behera. Prafulla.Kumar and Mohapatra.Durga.Prasad, Minimal Test Case Generation for Object Oriented Software with State Charts,

[11] Korel.B,L.Tahat and M.Harman, Test Prioritization using System Models, In proceedings of the 21st IEEE International Conference on Software Maintenance (ICSM 2005), page 559-568, September 2005.

[12] Wegener J, Baresel A, Sthamer H,2001,Evolutionary Test Environment for Automatic Structural Testing, Information and software technology, 43 (2001), pages 841-854.

[13] Hyunsoo Do and Gregg Rothermel, On the Use of Mutation Faults in Empirical Asssessments of Test Case Prioritization Techniques, IEEE Treansactions on Software Engineering, V. 32, No. 9, pages 733-725, 2006.

[14] G.Rothermel, R.J.Untch, C.Chir, Priortizing Test cases for Regression Testing, IEEE Transactions on Software Engineering, 27(10):929-948, October 2001.

[15] H.K.N. Leung , L. White, A Cost Model to Compare Regression Test Reduction using Dependence Analysis.In Proceedings of the IEEE International Conference on Software Maintainance(ICSM 1991), pages 201-208, IEEE Computer Society, October 1991.

[16] S.Tallam, N.Gupta, A Concept Analysis Inspired Greedy Algorithm for Test Suite Minimization, SIGSOFT Software Engineering Notes, 31(1): 35-42, 2006.

[17] PakinamN.Boghdady, NagwaL.Badr, Mohamed Hashem, TolbaF.Mohamed, Test Case Generation and Test Data Extraction Technique

[18] Nirpal. PB and Kale .KV, Using Genetic Algorithm for Automated Efficient Software Test Case Generation for Path Testing,International Journal onAdvanced Networking and Application, Vol:02, Issue:06, Pages:911-915, April 2011.

[19] Binder RV. Testing Object Oriented Systems:Models,Patterns and Tools. Addison Wesley: USA , 2000.

[20] Harman M, Kim. S.G, Lakhotia K, McMinn P, and Yoo S. Optimizing for the Number of Tests Generated in Search Based Test Data Generation with an Application to the Oracle Cost Problem, proceedings of the third International Conference on Software Testing, Verification and Validation Workshop: Paris, France, 2010.

[21] Sarma M, Kundu D, Mall R. Automatic Test Case Generation from UMl Sequence Diagrams. Proceedings of the 15th International Conference on Advanced Computing and Communications, IEEE Computer society: Washington, DC, USA, 2007.

[22] Tillmann N and De Halleux J.Pex__ White Bor . Test Generation for .NET. In proceedings of the second Conference on Tests and Proofs (TAP) : Prato, Italy, 2008.

[23] Legeard B. Model-Based Testing : Next Generation Functional Software Testing. Proceedings of practical Software Testing: Tool Automation and Human Factors Seminar. Published by :SchlossDagstuhl_ Leibniz_ ZentrumfuerInformatik : Dagstuhl, Germany, 2010.

[24] Li N, Xie T, Tillmann N, Halleux J, and Schulte W. Fitness –Guided Path Exploration in Dynamic Symbolic Execution. Proceedings of the IEEE/ACM International

Journal on Automated Software Engineering (ASE'09) : Auckland, New Zealand, 2009.

[25] H.Treharne, J.Draper and Schneider S, Test Case Preparation using a prototype.

[26] Guitieuez. Javier J, Escalona .Maria. J,Mejias Manuel, Toues Jesus, Generation of Test Cases from Functional Requirements. A Survey.

[27] P.D Ratna Raju, Suresh,Cheekaty, Harish Babu. Kalidasu, object Oriented Software Testing, International Journal of Computer Science and Information Technologies, Vol.2(5), ISSN: 0975-9646, pg2189-2192, 2011.

[28] N.J.Tracey, A Search Based Automated Test Case Data Generation Framework for Safety-Critical Software, PhD thesis, University of New York, 2000.

[29] Swain. Santosh. Kumar, Mohapatra. Durga.Prasad, Mall. Rajib, Test Case Generation Based on Use Case and Sequence Diagram, International Journal of Software Engineering, IJSE, Vol.3 No.2, July 2010.

[30] Shanthi.A.V.K, Automated Test Cases Generation for Object Oriented Software, Indian Journal of Computer Science and Engineering(IJCSE), ISSN:0976-5166, Vol.2 No.4, Aug-Sept 2011.

[31] PakinamN.Boghdady, NagwaL.Badr, Mohamed Hashem, TolbaF.Mohamed,A Proposed Test Case Generation Technique Based on Activity Diagrams, International Journal of Engineering and Technology, IJET-IJENS,ISSN: 114703-5858, Vol:11, No:03, June 2011.

[32] SaswatAnand et al.,20xx, Antonio. Bertolino, J.Jenny.Li, Hong .Zhu(Editors/Orchestrators),An Orchestrated Survey on Automated Test case Generation, Journal on Systems and Software X(y), xxCyy. Proceedings of IEEE/ACM Workshop on Automation of Software Test(AST 06-AST 12): Feb 11,2103.

[33] Mishra. Manish, Mishra.Shalini, Porawal.Robins, Basic Principle for Test Case Generation Automatically, VSRD International Journal of Computer Science and Informatiom Technology, VSRD-IJCSIT, Vol.2(9), pg. 772-781, 2012.

[34] Y.G.Kim, H.S. Hong, D.H.Bae and S.D.Cha et al., Test case Generation from UML State Diagrams, Software Testing Verification and Reliability, 1999,Pages 187-192.

[35] J.A.Jones, M.J.Harrold, Test Suite Reduction and Prioritization for Modified Condition/Decision Coverage, IEEE Transactions on Software Engineering, 29(3), Pages 195-209, March 2003.

[36] Roongruangsuwan.Suipong, Daengdeg. Jirapun, Test Case prioritization Techniques, Journal of Theoretical and Applied Information Technology,2005-2010, JATIT and LLS.

[37] Gaurav.Sahni,Kestina.Rai, Software Testing Techniques for Test Case Generation, International Journal of Advanced Research in Computer Science and Software Engineering, Vol .3, Issue 9, ISSN;2277 128x, Sept 2013.

[38] Yoo. Shin, Harman. Mark, Regression Testing Minimization, Selection and Priortization: A Survey, King's College London, Centre for Research on Evolution, Search and Testing, Strand, London, WC2R 2LS, UK.

[39] David.Leon, Andy. Podgurski, A Comparison of Civerage-Based and Distribution-Based Techniques for Filtering and Prioritizing Test Cases,Proceedings of International Symposium, Software Reliability Engineering, pp.442-453, 2003.

[40] Suri.Bharti, Mangal.Isha, Regression Test Suite reduction using an Hybrid Technique Based on BCO And Genetic Algorithm, IJCSI, ISSN (Print):2231-5292,Vol.-II,Issue-1, 2.