Volume 5, Issue 4, April 2015 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

An Overview of Automatic Test Data Generation and Meta-heuristic Search Techniques

Amita Gautam
Departmentof Computer Science & Engineering
Amity School of Engineering & Technology, Amity University,
Uttar Pradesh, India

Abstract—The size and complexity of the software are dramatically increasing everyday, which in turn makes software testing a tedious and costly task. So as to reduce the high cost of manual software testing and to improve the quality of testing process require its automation. With automation large scale testing can become practical and scalable. Automatic Test Data Generation is currently serving as an important research field. As the software is constrained by its size and complexity, metaheuristic search techniques provide solution to this problem. This paper provides a brief overview about automatic test data generation and meta-heuristic search techniques.

Keywords— Automatic test data generation; Meta-heuristic search techniques; Coverage criteria.

I. INTRODUCTION

Software testing is considered to be an expensive & time-consuming component which accounts almost 50% of software system development resources [1] [2] [3]. Software testing is performed with the intent of finding as many faults as possible, while executing a program. It is a kind of review that's conducted to provide customers with the information regarding quality of product or service which is being tested. The most labor intensive part of the software testing procedure is the generation of test data, to satisfy the testing requirements. Automating the testing process is a crucial concern since it will provide more systematic approach to testing [4], thereby reducing the analysis cost as well.

A number of automatic test data generation technique have been developed so far [2]. Pargas [5] classified these techniques as random test data generator, path-oriented test data generator intelligent test data generator and goal-oriented test data generator. Input data is being generated by the test data generator for the target program in a way that these input data satisfy a particular testing objective. Further meta-heuristic search techniques for automatic test data generation are serving as an important research area for researchers. Applying meta-heuristic search techniques for automatic generation of test data solves many problems regarding automation of test generation. The size and complexity of software served as major constrain in this area. So meta-heuristic search techniques, which are applied to structural and functional testing utilize heuristics to get solutions for combinatorial problems. Such problems can be classified as.NP-complete or NP-hard. For generating the test data, the transformation of test criteria toobjective function is required. Search-Based Software Test Data Generation is just a small part of Search- Based Software Engineering [6] [7].

Rest of the paper is organized as follows. Section 2 discusses about automatic test data generation. Section 3 briefs about important meta-heuristic search techniques. Section 4 describes various coverage criterions. Section 5 presents the conclusion of the paper.

II. AUTOMATIC TEST DATA GENERATION

Before going to automatic test data generation, let us discuss about test data generator systems [15]. This consists of three parts – a path selector, program analyzer and test data generator.

A. Test Data Generator

In this system, program analyzer provides all kind of information pertaining to program, control flow path, data dependence graph etc. Whereas the path selector identifies paths for which input values will be derived by the test data generator. Finding input values which will traverse paths provided by selector, is our major goal. This is accomplished in two steps:

- Find the path predicate for the path.
- Path predicate is solved in terms of input variable.

Output of this will result in a system of $i(n)$equalities describing the formation of input data which traverses the path. On such systems various search methods can be applied for the solution [8] [9] [10]. Basically three approaches are followed for construction of test data generator: randomly generated test data, generate test data for specific path or for unspecific path. All these approaches lie under random, goal oriented and path oriented test data generation.

B. Random Test Data Generation

Random testing is simplest of all, as it could be used by any type of program to generate input values. Since data types such as string, integer or heaps is just a stream of bits. We can randomly generate bit of streams for a function that takes string as argument.

*Limitations:*Random test data generation is not a good performer when it comes to coverage. As it hardly relies on probability, it has less chance to identify semantically small-faults [11], thus it is providing high coverage. As it creates large amount of test data without any information about testing objective, generator often failed to identify data which satisfies the objective stated by the testing process.

C. Goal Oriented Test Data Generation

Goal oriented approach [13] provides better results than random generation in terms of providing assistance towards a definite set of paths. The generator generates the input that will traverse the given unspecific path, which enables the generator to find input for any path. This reduces the chances of encountering comparatively infeasible paths. Two methods using this technique are as followed:

*1) Chaining Approach:*It uses data dependence for finding solutions to branch predicates. In this approach chain of nodes are identified that crucial for the execution of goal nodes.

2) *Assertion Oriented Approach:*Here certain conditions called assertions when executed are supposed to be held, or else there is an error either in program or in the assertion. These assertions are inserted in to the code either manually or automatically.

D. Path Oriented Test Data Generation

Path oriented generation is similar to goal oriented test data generation, apart from the use of specific path. In this generator is not provided an option of selecting among a set of paths, rather just single path.

III. META-HEURISTIC SEARCH TECHNIQUES

Following three important Meta–heuristic Search-Based Techniques are described below:

A. Hill Climbing

Hill climbing is a local search [12] that proceeds from randomly chosen starting point from search space. This point is considered as an initial solution which goes on improving through hill climbing algo. Neighborhood of initial point is investigated and a move is made to replace the current solution if better solution is found in terms of fitness improvement. This process is continued until, no improved neighbors (solutions) are found for present solution. There are two options: in next ascent hill climbing, the investigation is made to find the first neighbor that provides an improved fitness. In steepest ascent hill climbing, the complete neighborhood set is evaluated to find the solution with maximum increase in fitness. In case no fitter neighbor is explored, than the search terminates and a locally optimal solution is found.

Metaphorically speaking, this progressive improvement is exactly like climbing of hill in the search landscape. Now the problem with this approach is that the hill sited by the algorithm may provide locally optimal solutions, and may be worse than globally optimal in the search space. An extension to hill climbing algorithm is achieved by integrating sequences of "restarts" involving diverse initial solutions, to test more of the search space and reduce this problem. Hill climbing is a simple approach which is effective and easy to implement.

B. Simulated Annealing

Simulated annealing is a kind of variation of hill climbing which allows probabilistic acceptance of less fit individuals. This approach permits for less restricted movements around the search space. Simulated annealing is replication of metallurgical annealing, in which heated material allowed to cool slowly there by making it stronger. Temperature fall reduces the freedom of movement among atoms. But in earlier stages when metal was hot, atoms explode different energy states. As the search progresses, probability of acceptance of less fit solution changes. This probability is a drop in fitness function and a temperature parameter simulates the metal temperature in metallurgical annealing. The probability of taking an unfavorable move is condensed by the effect of cooling on 'the simulation of annealing'.

C. Genetic Algorithm

Basic concept of GAs was first described by Holland [14]. GAs has been applied to a variety of problems involving search of optimization. GAs revolves around the principle of "survival of the fittest" hence, drawing inspiration from the natural search and selection processes. The basic idea of algorithm beginswith the initialization of random population of individuals. In this, each individual presents a potential candidate solution of a given problem. Each individual in the environment uses fitness function to evaluate its adequacy and quality. After this, the selection of the solutions in the current population occurs, these selected solution are going to serve as parents in the next generation. This selection requires evaluation of solution for their fitness as parents. This means the solution with high fitness values are likely to be selected. Now these selected individuals are combined through crossover operation, which results in a new generation of population. The process crossover takes two individuals that swap chunks of data at randomly selected position. Mutation process is applied, to introduce slight changes into a small fraction of population. This process is continued until the population evolves to form a solution to the problem, or until the maximum number of iterations occurred.

```
Initialize (population)
Evaluate (population
While (stopping condition not satisfied) do
{
    Selection (population)
    Crossover (population)
    Mutate (population)
    Evaluate (population)
}
```

IV. COVERAGE CRITERIA

The following coverage criterions [16] are considered in Search-Based Software Testing.

A. Statement Coverage

Statement coverage is achieved when each statement in the program is executed at least once.

B. Decision Coverage

Decision coverage is also know branch coverage, sees the extent to which all outcomes of branch statements are covered by test suits.

C. Condition Coverage

Statement in the code is executed at least once, and every condition provided in each decision has taken all possible outcomes at least once.

D. Multiple Condition Coverage

In this the test data is required to evaluate all the combinations of conditions for each decision.

E. Modified Condition Decision Coverage

Following points are considered in MCDC:

- It should not affect outcome of decision independently.
- All possible outcomes should be passed at least once.

V. CONCLUSION

Automatic test data generation is a technique that automatically generates test data for the software under test. By this software testing becomes more efficient and cost effective. Software test data generation isvery important for software testing as test data serve as one of the key factors for shaping the quality of any software testing process during its execution. Whereas Meta-heuristic Search Techniques provide high-level frameworks which incorporate heuristics to find solutions to combinatorial problems. As test data generation is an undecidable problem, applying meta-heuristic search techniques to test data generation provides promising solutions.

ACKNOWLEDGMENT

I would like to express my sincere gratitude to Dr. Deepak Arora, Head – Department of Computer Science & Engineering and Amity University for their support and guidance.

REFERENCES

[1] Roper, M., Maclean, I., Brooks, A., Miller, J., and Wood, M. Genetic Algorithms and the Automatic Generation of Test Data

[2] Whittaker, J.A. What Is Software Testing? And Why Is It So Hard? IEEE Software. February, 2000.

[3] Beizer, B. Software Testing Techniques. Van Nostrand Reinhold, New York. 1982

[4] Bashir, M.F, Banuri, S. H. K., “Automated model based software Test Data Generation System”, Proc. 4’Th International Conference on Emerging Technologies, pages 275-279, Oct.2008.

[5] Pargas, R.P., Harrold, M.J., and Peck, R.R. Test-Data Generation Using Genetic Algorithms. Journal of Software Testing, Verification and Reliability. 1999.

[6] M. Harman and B. Jones. Search-based software engineering. Information and Software Technology, 43(14):833{839, 2001.

[7] W.H.Deason,D.Brown,K.Chang,andJ.H.CrossII.Arule-basedsoftwaretestdatagenerator.IEEE Transactions onKnowledge andDataEngineering,3(1):108-117, March1991

[8] W.H.Deason,D.Brown,K.Chang,andJ.H.CrossII.Arule-basedsoftwaretestdatagenerator.IEEE Transactions onKnowledge andDataEngineering,3(1):108-117, March1991

[9] R.FergusonandB.Korel.Thechainingapproachforsoftwaretestdatageneration.IEEETransactions on Software Engineering,5(1):63{86, January 1996

[10] N.Tracey,J.Clark,andK.Mander.Automatedprogramflawfindingusingsimulatedannealing.In ProceedingsofACMSIGSOFTinternationalsymposiumonSoftwaretestingandanalysis,volume23, pages73-81,March 1998.

[11] J.O_uttandJ.Hayes.Asemanticmodelofprogramfaults.InInternationalSymposiumonSoftware Testingand Analysis (ISSTA96),pages 195{200.ACMPress, 1996.

[12] McMinn,P. (2004):Search-basedsoftwaretestdata generation:Asurvey, SoftwareTesting,Verification and Reliability, 14(2): 51, pp. 105-156

[13] B.Korel.Dynamicmethodforsoftwaretestdatageneration.SoftwareTesting, Verificationand Reliability, 2(4):203-213,1992.

[14] J.Holland.Adaptationinnaturalandartificialsystems.AnnArbor:TheUniversityofMichiganPress, 1975.

[15] Korel,B.AutomatedSoftwareTestDataGeneration.IEEETransactionsonSoftwareEngineering, Volume 16, No.8, pp. 870-879,August,1990.

[16] PrasadBokil,PriyankaDarke,UlkaShrotri,VenkateshR.(2009):AutomaticTestDataGenerationforC Programs,ThirdIEEEInternationalConferenceonSecureSoftwareIntegrationandReliability Improvement.