IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 5, Issue 10, October 2016

Application of Genetic Algorithms in Software Testing Techniques

Dr. V. Sangeetha[1], T. Ramasundaram[2]

Head, Assistant Professor, Department of Computer Science, Periyar University College of Arts & Science, Pappireddipatti, Dharmapuri, Tamilnadu, India[1]

Assistant Professor, Department of Computer Science, Sri Vijay Vidyalaya College of Arts & Science, Nallampalli, Dharmapuri, Tamilnadu, India [2]

Abstract: The ultimate aim of the software testing is to deliver a quality and reliable software product to end user. To ensure software quality, we need an effective software testing, but it is not an easy job, we have to face certain issues like an effective generation of test cases, prioritization of test cases and so on. To overcome these issues, various techniques and methodology have been proposed. At present, automatic test case generation using evolutionary algorithm has been the area of interest for many researchers. Genetic Algorithm (GA) is one of the evolutionary algorithms whi1ch produce an optimal solution to any problem. In this paper, we are going to briefly discuss applications of genetic algorithm in various software testing techniques.

Keywords: Genetic Algorithm, Software Development Life Cycle, Software Testing, Test Data

I. INTRODUCTION TO SOFTWARE TESTING

Software Testing is an activity in Software Development Life Cycle (SDLC) where the errors remaining from all the previous activities must be detected. Hence, Software testing performs a vital role in SDLC for ensuring software quality and reliability. During testing, system's behaviour is monitored, so that we determine whether or not there is a failure. Testing can only reveal the presence of faults, not their absence [1].

Verification and validation processes can also be used to checking the software that whether or not it meets its requirement specification and the functionality expected by the user. Verification is made to ensure that the software meets specification and is related to structural testing whereas validation is related to the functional testing and is made by executing software under test [2]. The ultimate goal of verification and validation processes is to establish confidence that the software system is 'fit for purpose' [3].

There are different kinds of software testing techniques. Broadly, there are two basic types of testing techniques: Black box testing and White box testing. Black Box Testing is also called functional testing because this testing is only concerned with the functionality of the software being developed.

White box testing is also called structural testing in which only concern is internal structure of the software. In white box testing, path testing considers 1.Control flow testing - it tests all possible paths of the control flow graph. 2. Data flow testing - during testing, it tests the definitions of variables and their subsequent use.

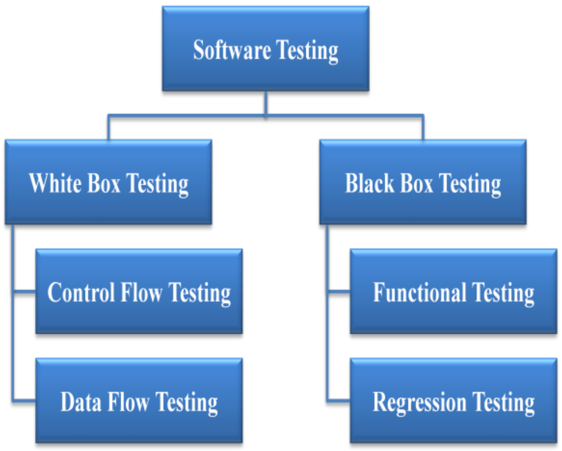

Fig.1 Basic Types of Software testing

Testing can be done either manually or automatically by using tools. As per as quality, performance, and cost of software development are a concern, it is found that automatic testing is better than manual. However, very few automatic test case generating tools are available today.

Various types of techniques have been proposed for generating test cases automatically. Recently, a lot of work is being done for automatic test cases generation using soft computing techniques like fuzzy logic, neural networks, Genetic Algorithm, and evolutionary computation providing keys to the problem areas of software testing. Genetic Algorithm often gives an optimal solution to all type of problems. Genetic Algorithm is an emerging methodology for automatic test case generation for various

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 5, Issue 10, October 2016

types of testing techniques. In this paper, various software testing techniques which perform using Genetic Algorithms are presented.

II. INTRODUCTION TO GENETIC ALGORITHMS

Genetic algorithm (GA) is a kind of evolutionary algorithms. It is a general purpose and robust optimization technique based on the way nature evolves species using the natural selection of the fittest individuals. The possible solutions to the problem are represented by a population of chromosomes.

A chromosome is a string of binary digits and each one digit that creates a chromosome is called a gene. This initial population can be totally random or can be created manually using the greedy algorithm. The pseudo code of a basic algorithm for GA is as follows [4].

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

Fig. 2 The Pseudo code for basic GA Algorithm

GA uses three operators on its population which is described below:

A. Selection

To determine how individuals are preferred for mating based on their fitness values is done by selection scheme. First, the fitness values can be defined as the capability of an individual to survive and reproduce in an environment.

Fitness values calculated using fitness function proposed in the algorithm. Weights are used to find the relative contribution of a path to the fitness calculation. In consequence, more weight is assigned to a path which is more “critical”.

Selection scheme generates the new population from the old one, thus starting a new generation. Every chromosome is evaluated in present generation to determine its fitness value. This fitness value is used to select the better chromosomes from the population for the next generation. The fitness function is using here is

$$F = \sum_{i=1}^{n} wi$$

Where, wi = weight assigned to i^th edge on the path under consideration

The algorithm works by assigning weights to the edges of Control Flow Graph on the basis of the importance of path in which the edge lies. Higher weights are assigned to the edges of the path corresponding to the critical section of the code for example branch statements, loops, control statements etc. for which testing is necessary. After all the fitness function values are intended, the possibility of selection pj for each path j, so that

$$pj = Fj / \sum Fj$$

n= initial population size

$$ck = \sum_{j=1}^{k} pj$$

Then cumulative possibility ck is measured for each path k with an equation [5][6].

B. Crossover or Reproduction (Recombination)

After selection, the crossover operation is applied to the selected chromosomes. It involves an exchange of genes or sequence of bits in the string between two individuals. Crossover happens according to a crossover possibility pc, which is an adjustable parameter. For each parent selected, generate a random real number r in the range [0, 1]; if $r < pc$ then select the parent for crossover. After that, the selected data are formatted randomly. Each pair of parents generates two new paths, called offspring.

The crossover technique used is one point crossover done at the midpoint of the input bit string. After crossover, the mutation operator is applied to a randomly selected subset of the population [5].

C. Mutation

Mutation is performed on a bit-by-bit basis. Mutation alters chromosomes in small ways to introduce new good traits. It is applied to bring diversity in the population. Every bit of every chromosome in the offspring has an equal chance to mutate (change from “0” to “1” or from “1” to “0”), and the mutation occurs according to a mutation possibility pm, which is also an adjustable parameter.

To perform mutation, for each chromosome in the offspring and for each bit within the chromosome, generate a random real number r in the range [0, 1]; if $r < pm$ then mutate the bit.

III. GENETIC ALGORITHM IN SOFTWARE TESTING TECHNIQUES

In this section, we will discuss in detail about the applications of Genetic Algorithm in various testing techniques [7]. Software testing is an optimization problem to minimize the number of test cases and minimize the time, cost and effort. And also increase the quality of the software.

A. Applications of GA in White Box Testing

White Box Testing is used to test internal structure of a program. It includes statements, conditional statements, loop statements. Structural testing aims to achieve the test cases that will force the desired coverage of different structures. In some of the research work discuss the code coverage, data flow testing, control flow testing and mutation testing using Genetic Algorithm.

1).Control Flow Testing or Path Testing:
Praveen Ranjan Srivastava and Tai-Hoon Kim [8], has been proposed a technique for identifying the most critical path clusters in a program using the genetic algorithm to generate test cases. This approach uses a weighted CFG (Control Flow Graph). Path testing searches a suitable test case that covers every possible path in the program to find out the errors. If the program has loops, then there will be an infinite number of the path.

To cover each path, we need a large number of test cases, it becomes computationally impractical. Since it is impossible to cover all paths in the program, the path testing selects a subset of paths to execute and find test cases to cover it. CFG selects an independent path for a new set of statements or condition. While testing, every independent path must traverse at least once.

S.Keshavarz and Reza Javidan [9], Proposed a new technique using Genetic Algorithm to generate test data. In coverage path testing, a test data is good data if causes to an independent traversal of a path. The main worry about the software testing is automatic and ordered generation of data is mandatory and sufficient for testing. Data is a mandatory and sufficient only if it causes a traversal on an independent path. For that, we offer a systematic and automated procedure to generate data necessary and sufficient test of a program based on program control flow graph and the covered aim of critical edges.

Yeresime Suresh and Santanu Ku Rath [10], Worked on automated test data generation using GA. Here the test data defined as the population in GA. In initial population, each individual bit string (chromosome) is a test data. This set of chromosomes is used to generate test data for feasible basis paths. The procedure for generating test data for feasible basis paths using GA is coded using MATLAB. It randomly generates the initial population, evaluates the individual chromosome based on the fitness value and applies the GA operations such as selection, crossover, and mutation to produce next generation. This iterative process stops when the genetic algorithm finds optimal test data.

2). Data Flow Testing:
Moheb R. Girgis, Ahmed S. Ghiduk, and Eman H. Abd-Elkawy[11]. Worked on automatic test path generation based on two proposed GA-based and PSO-based techniques that cover the all-uses criterion for the program under test. These two techniques do their search by constructing new paths from previously generated paths that are evaluated as effective test paths. Then, we presented a GSO-based technique that effectively combines the proposed GA-based and the PSO-based techniques to improve the individual's score for natural selection of the fitness and for good knowledge sharing at the same time.

In each iteration of the proposed GSO algorithm, the population is divided into two parts and they are evolved with the two techniques respectively. They are then recombined in the updated population, that is again divided randomly into two parts in the next iteration for another run of genetic or particle swarm operators.

Na Zhang, Biao Wu and Xiaoan Bao [12], Proposed a method for generating test data automatically using Multi-Population Genetic Algorithm. The algorithm defined the concept of external pressure which as the degree of competition between individuals. Fully considering the influence of coverage, branch condition and degree of competition between individual species of three aspects, and give different weights, we design a fitness function to evaluate the merits of the individual species.

Janvi Bandlaney, Rohit Ghatol, Romit Jadhwani [13], Presented a paper on an introduction to data flow testing. In his paper, they generated the idea of a control flow testing. According to them, control flow diagrams are a keystone in testing the structure of software programs. By examining the flow of control between the various components, they designed and selected test cases. Data-flow testing is a control-flow testing technique which also examines the life cycle of data variables. The main goal of their paper is to discuss the concept of data-flow testing and applying it to a running example.

B. Application of GA in Black Box Testing

Black box testing is which testing the functionality of software and software full fill their specification and user requirement. In some research performed, functional testing and regression testing using Genetic Algorithm.

1). Functional Testing
Francisca Eanuelle [14] has presented a GA-based technique to generate good test plans for functionality testing in an unbiased manner to avoid the expert's interference. The motivation behind this work is to prove that the GA is able to generate good test plan although the

best sequence of the test plan is unknown. The test plan or test sequence totally relies on the experts or the people who understand the application well.

The emphasis is given on the fact that “an error in a program is not necessarily due to the last operation executed by the user but may have been due to a sequence of previously executed operations that leads an application in an inconsistent state”.

In other words, as a sequence of operations is executed, the state of inconsistency is non-decreasing or a problem in a software application is directly proportional to the level of inconsistency of the state in which application is. In this work, the operation of large granularity has been chosen so that the sequence of operation that leads application to the inconsistent state can be identified.

Ruilian Zhao, Shanshan lv [15], used the neural network and GA for the functional testing. The neural network is used to create a model that can be taken as a function substitute for the SUT. The emphasis is given on the outputs which exhibit the important features of SUT than inputs. In that case, test cases should be generated from the output domain rather than input domain.

The feed forward neural network and backpropagation training algorithm are used for creating a model. The neural network is trained by simulating the SUT. The outputs generated from the created model are fed to the GA which is used to find the corresponding inputs so that automation of test cases generation from output domain is completed.

In this paper, inputs to the GA are the function model generated from the neural network, a number of input variables n, range of input variables that is upper [n] and lower [n], population size, maximum iteration number, goal output g, maximum fitness function f, crossover probability and mutation probability.

The fitness function is defined as max where c is the actual output and the g is the goal output of the SUT. The population is evaluated by applying GA.

The difference between goal output and the actual output of SUT using the neural network is used for calculating fitness value of the individuals in the population. If fitness value exceeds or reaches the maximum fitness value, then search stops and the current individual is taken as the test inputs for the corresponding outputs.

2). Regression Testing
N. Kaushik, M. Salehie, L. Tahvildari, and S. Li, M. Moore [16], proposed a paradigm called Dynamic Prioritization which involves changing the order of test cases during the testing process. Since the test case pool changes through the development cycle, the list of prioritized test cases would change as well.

A. Kaur, S. Goyal [17] proposed a new Genetic Algorithm to prioritize the regression test suite is introduced that will prioritize test cases on the basis of complete code coverage. The genetic algorithm would also automate the process of test case prioritization.

A. Jiang, Y. Mu, Z. Zhang [18], Selects the test cases that can test part of changes and then do the reduction for these selected test cases. In addition to the methods mentioned in this section, a large number of methods were proposed in the past. A good survey of test case prioritization methods, as well as algorithms for optimal test sequence analysis, can be found in [19] and [20].

IV. CONCLUSION

Various techniques have been proposed for test case generation; however, no one could achieve the best performance for every piece of code. Test case generation becomes an optimization problem today. So, there are scope remains open for applying some more technique to achieve a better result.

A Genetic algorithm is one such optimization technique. In this paper, the applications of genetic algorithm in various software testing techniques have been discussed. This will pave the path for further work in this direction.

REFERENCES

[1] Pankaj Jalote, An integrated approach to software engineering, 3rd edition, Springer
[2] Paul C. Jogersen, Software testing: A craftsman approach. 3rd edition, CRC presses, 2008.
[3] Ian Somerville, Software engineering, 9th edition, Pearson, 2011.
[4] Goldberg, D.E, Genetic algorithms: in search, optimization and machine learning, Addison Wesley, M.A, 1989.
[5] Klir, G.J., Yuan, B.: Fuzzy sets and fuzzy logic: Theory and applications. Prentice-Hall Inc., Englewood Cliffs (1995)
[6] Last, M., Eyal, S.: A Fuzzy-based lifetime extension of genetic algorithms. Fuzzy sets and systems 149(1), 131–147 (2005).
[7] Chayanika Sharma, Sangeeta Sabharwal, Ritu Sibal, Software testing techniques using genetic algorithm, IJCSI International Journal of Computer Science Issues, Volume 10, Issue 1,2013, 1694-0784
[8] Praveen Ranjan Srivastava and Tai-hoon Kim, Application of genetic algorithm in software testing, International Journal of Software Engineering and Its Applications Volume 3,Issue 4,2009
[9] S. Keshavarz and Reza Javidan, Member, IACSIT, Software quality control based on genetic algorithm, International Journal of Computer Theory and Engineering, Vol. 3, No. 4, August 2011
[10] Yeresime Suresh and Santanu Ku Rath, A genetic algorithm based approach for test data generation in basis path testing, The International Journal of Soft Computing and Software Engineering [JSCSE], Vol. 3, No. 3, Special Issue: e-ISSN: 2251-7545, March 2013.
[11] Moheb R. Girgis, Ahmed S. Ghiduk, and Eman H. Abd-Elkawy, Automatic data flow test paths generation using the genetical swarm optimization technique, International Journal of Computer Applications (0975 – 8887) Volume 116 – No. 22, April 2015.
[12] Na Zhang, Biao Wu and Xiaoan Bao, Automatic generation of test cases based on multi-population genetic algorithm, International Journal of Multimedia and Ubiquitous Engineering Vol.10, No.6 (2015).
[13] Janvi Bandlaney, Rohit Ghatol, Romit Jadhwani, An introduction to data flow testing, NCSU CSC TR-2006.

[14] Francisca Emanuelle et. al., “Using genetic algorithms for test plans for functional testing”, 44th ACM SE proceeding, 2006, pp. 140 - 145.
[15] Ruilian zhao, shanshan lv, “Neural network-based test cases generation using genetic algorithm” 13 IEEE international symposiums on Pacific Rim dependable computing. IEEE, 2007, pp.97 - 100.
[16] N. Kaushik, M. Salehie, L. Tahvildari, S. Li, M. Moore (2011) “Dynamic prioritization in regression testing” IEEE fourth international conference on software testing, verification and validation workshops, pp: 135-138
[17] A. Kaur, S. Goyal (2011) “A genetic algorithm for regression test case prioritization using code coverage”, International journal on computer science and engineering, vol. 3, pp:1839-1847
[18] A. Jiang, Y. Mu, Z. Zhang (2010) “Research of optimization algorithm for path-based regression testing suite”, 2nd IEEE International workshop on education technology and computer science, pp:303-306
[19] R. Kavitha, N. S. Kumar (2010) “Test case prioritization for regression testing based on severity of fault” (ijcse) international journal on computer science and engineering Vol. 02, No. 05, pp: 1462-1466
[20] S. Raju, G. V. Uma (2012) “Factors oriented test case prioritization technique in regression testing using genetic algorithm” European Journal of Scientific Research, Vol.74, No.3, pp. 389-402.

BIOGRAPHY

T.Ramasundaram was born in Dharmapuri, Tamil Nadu (TN), India, in 1982. He received the B.Sc.egree in Computer Science from the Periyar University, Salem, Tamilnadu, India, in 2003 and the M.Sc. degree in Computer Sciencefrom the Periyar University, in 2005, and M.Phil degree from the Periyar University, in 2009. He is currently pursuing the Ph.D. degree with the Department of Computer Science, Periyar University. His research interests include Software Engneering and data mining.