SOFT COMPUTING BASED APPROACHES FOR TEST DATA GENERATION: A SURVEY

Naveen Shaillu [1]
Research Scholar
Baddi University of Emerging Science and Technology, Baddi, India
Amar Singh [2]
Baddi University of Emerging Science and Technology, Baddi, India

Abstract: Testing is a method of realizing a program with a plan to discover the software bugs. Software testing is complex and expensive process, usually consuming minimum 35% of total development time along with 50% of the total development cost. The key step of software testing is automated test data generation (ATDG). On the basis of test data suitability criteria, ATDG scheme creates best set of test data automatically. The objective of this contribution is to review the major techniques in identifying the automated test data generation. The main aim of this paper is to look into the many researches in the ATDG, various soft computing techniques and hybridization of these soft computing techniques.

Keywords **– Software Testing, Test Data, Automatic Test Data Generation (ATDG), Soft Computing.**

I. INTRODUCTION

Software testing is a component that is costing lots of money in software development and maintenance [1]. In software testing, test data generation is labour –intensive component. Test data is a documented data that is basically used to test the software program. Test data generation is a technique that helps to find program inputs which specify the testing requirement. Automated test data generation is a technique which helps to make software development and maintenance process cost and time effective process. Because it reduces time and cost require for software development and maintenance. There are two techniques which are used for automated test data generation such as functional and Structural testing. Structural testing are further divided into sub categories which includes different kind of approaches and implementation methods such as Random approach, Path-oriented approach, Goal-oriented approach, Static method, Dynamic method, Hybrid method, Intelligent Approach [2].

Structural Testing: These techniques are based on the core structure of a program, where test cases are selected so that structural components are covered. Examples of these techniques are statement testing, decision testing, path testing, condition testing, dataflow testing, structured testing, and domain testing [3]. Additional structural techniques could be based on different approaches and implementation methods.

Functional Testing: This type of testing is used where test cases are derived from the program's specification and it deals with the functionality of the software. Examples of these techniques are equivalence partitioning, boundary value analysis, random testing and functional analysis-based testing [3].

Random Approach: Random test data generation approach is basically consists of producing inputs at an unsystematic way until a useful input is not found. This approach is rapid and simple but might be a poor selection with complex programs and with complex adequacy standards [4].

Path-oriented Approach: Path –oriented approach is typical approach used for creating CFG (control-flow graph). In this method, at first a graph is generated first and consequently, by using the graph a particular path is selected. With the help of a technique such as symbolic assessment (in the *static* case otherwise it is called function minimization) test data is generated for that path in the end [5, 6]. In symbolic execution variables are used instead of actual values while traversing the path. The path-oriented approach might face the difficulties when generating paths/graphs, traversing test data through branches and predicates (infeasible path problem), and although difficulty of data types.

Goal-oriented Approach: In this approach test-data is chosen from the existing pool of candidate test data to implement the certain goal, such as a statement, no matter how the path is taken [5]. This approach includes two basic steps: to recognize a set of statements (respective branches) the covering of which involves covering the measure; to produce input test data that perform every chosen statement (respective branch) [6]. Two distinctive approaches, 'Assertion-based' and 'Chaining approach' are acknowledged as goal oriented. In the first case assertions are introduced and then resolved while in the second case data requirement investigation is carried out. Usually the goal-oriented approach considers problems of goal selection and choice of adequate test data.

Intelligent Approach: According to Pargas et al [5] the intelligent test-data generation approach frequently depend on sophisticated study of the code to guide the examine for new test data. However in the author's view this method can be lengthy up to the intelligent investigation of program requirement as well. With the planned lengthy capability this approach will lie between functional and structural testing.

Static Methods: These are the testing methods implemented for study and testing of system illustrations such as the requirement documents, design diagrams and the software source code, either manually or automatically, without truly executing the software [7, 8]. In additional words, the static methods do not involve the software under test to be performed. They typically use symbolic implementation to achieve constraints on input variables for the precise test principle. Outcomes to these controls symbolize the test-data [9]. Implementing a program symbolically means that instead of using actual values, 'variable substitution' is used.

Dynamic Method: In this method, Instead of using variable backup, these methods execute the software under test with some, feasibly randomly particular input [6]. By detecting the program flow the system can handle whether the projected path was taken or not. In case of an undesirable response the system backtracks to the node where the flow took the inappropriate path. Using different types of search methods the flow can then be changed by control the input in a way so that the accessible branch is taken.

Hybrid Methods: As the name suggests, a hybrid method is the combination of static and dynamic methods. This combination could be in the form of side-by-side use, probably resolving some tasks by using static methods and some by using the dynamic methods, or by both methods.

II. LITERATURE REVIEW

Tahbildar et al [10] were giving an outline of automatic test data generation and obtain the basic ideas associated to automated test data generation research. They were also labelling different execution techniques with their relative advantages and disadvantages, the future challenges and difficulties of test data generation were clarified and also define the area where more focus is essential for making automatic test data generation more effective in industry.

Bogdan Korel [11] presents another method for generation of test data which is based on tangible implementation of the program under test, function minimization methods, and energetic information flow study. In this method, Test data are established for the program using real values of input variables. When the program is performed, the program performance flow is checked. If during program performance an unwanted execution flow is observed then function minimization search algorithms are used to spontaneously locate the values of input variables for which the designated path is navigated. In addition, energetic data flow analysis is used to define those input variables answerable for the unwanted program performance, leading to important speedup of the search process.

Chauhan et al [12] proposed a metaheuristic technique spontaneously generation of test data particularly for web-based application. It is combination of GA (genetic algorithm) and fuzzy logic for automation test data generation. And then compare this combine technique with the earlier proposed hybrid GA-ACO technique. The results of new approach GAFLA is better than GA-ACO as in some cases, it has higher coverage ratio % than the GA-ACO.

Pargas et al [5] introduce a Generate Data technique by using genetic algorithm for automated test data generation. Generate Data technique is executed in a tool TGen. In TGen similar processing was used to recover the presentation of search. With the usage of TGen, an unsystematic test data generator, called Random too was performed. Both TGen and Random were in use, to observe the production of test data for statement and branch coverage.

Gupta et al [13] offered a method to generate automatic test data for object oriented software with the use of genetic algorithm. This technique helps to evaluate the suitability of both possible and impossible test cases. The projected approach shows that GA is valuable in decreasing the quantity of impossible test cases by producing test cases for object oriented software.

Chawla et al [14] offered a hybrid approach of PSO (particle swarm optimization) and GA (genetic algorithm) for automated test data generation for procedural and object oriented programs. And then compare this combine technique with the earlier proposed hybrid technique PSO and GA. PSOG performs well on all the tested classes for all other strategies under evaluation.

Singla et al [15] introduce a technique which is combination of GA and PSO, that's why it is called GPSCA (Genetic-Particle Swarm Combined Algorithm) which is used to produce spontaneous test data for information flow analysis by using dominance concept among two nodes. The performance of the planned method is examined on quantity of programs having diverse scope and difficulty. The recitation of Genetic-Particle Swarm Combined Algorithm is associated to both Genetic and Particle Swarm Optimization algorithms.

Lu et al [16] introduce an approach for producing test data for extended finite state machine (EFSM) paths as condition analysis. In this approach, GA is used to produce test data for EFSM paths. They analyze the Kalaji's fitness function, catch out some difficulties of it, then offer a new fitness function merging the branch distance with uncovered condition ratio, and compare their proposed method with Kalaji's method.

Gupta et al [17] presented a program implementation based method to produce test data for branch analysis. This method energetically changes to a path that deals fewer resistances to generation of an input to force implementation over it to reach the known branch. This method is appropriate to programs with common nonlinear terminologies. It can handle altered programming language features. This method is fit for automation and has prospective to deliver a useful solution to the test data generation problem.

GONG et al [18] proposed a method using GA for Producing test data for path analysis and error revealing. This technique is beneficial to some real-world programs, and equated with a random method and evolutionary optimization method.

Yao et al [19] offered a scientific model for multiple paths coverage to generate test data. Formerly, a multipopulation GA with single sharing is untaken to explain the planned model. Then examined the performance of the planned technique in several test programs. The tentative results show that the proposed method can increase the productivity of generating test data for many paths coverage.

Kumar et al [20] shows that amount of test cases essential in case of Spanning tree is fewer as associate to the Dominance tree and use an optimization method which is based on natural computing theory for the generation of debauched, exclusive and consistent test cases which is named as on Swine Influenza Models Based Optimization (SIMBO). Finally, SIMBO technique compare with PSO and GA.

Blanco et al [21] shows and analyze dualistic versions of a method established on the Scatter Search metaheuristic method for the automatic test cases generation and using a branch coverage acceptability measure. The primary test case producer, called TCSS, uses an assortment property to spread the search of test cases to all branches of the program under test in order to produce test cases that cover the all branches. The another, called TCSS-LS, is an extension of the earlier test case producer which syndicates the assortment property with a limited search method that permits the growth of the search for test cases that cover the challenging branches. Their result shows good performance of their approach than many other generators.

Ahmed et al [22] proposed genetic algorithms (GA) for structural testing to generate test data automatically. They consider path coverage as the test acceptability standard. They planned a genetic algorithm-based test data originator that is, in single run, able to produce various test data to cover several target paths. Their outcomes show that their test data generator is more efficient and more effective than others.

Mehdi et al [23] present the EvoPSO algorithm using swarm intelligence paradigm. The algorithm is implemented in EvoSuite tool for the purpose of test data generation. The performance of EvoPSO has been investigated on SFIIO dataset. The performance shows that EvoPSO is efficient and can give competitive results.

Nikravan et al [24] proposed a method for path coverage analysis to generate test data. The main problem of this type testing lies in its capability to build a test suite efficiently and reduce the number of discards. They address this difficulty with a new divide-and-conquer method based on adaptive random testing approach. This method generates a lesser amount of unacceptable inputs and is capable of finding the sub-domain of several difficult constraints.

Zhu et al [25] proposed a new method to increase the productivity of generating a set of test data for several paths in Search-based Software Testing (SBST). Also, they improve the outdated grouping approach to balance the load of dissimilar calculation resources. A technique is used for gathering the constraints to decrease the search space of test data, this technique is known as symbolic execution. In order to avoid the challenges in symbolic execution, the comfortable version of constraints are projected to enhance the method. Their result shows good performance of their method in both efficiency and load balancing.

Hong et al [26] present the idea of dynamic program slicing for generating test data spontaneously; the effectiveness of creating test data can be enhanced. for generating test data, The core procedure is as follows: First, in the program, they analyze the dynamic percentage of the interest point's variable, and catch the present value of the interest point's variable; Then In the branch function, they use the technique of minimization, and monitor the change of program input.

Kumar Yadav et al [27] present a new method to create test data spontaneously for information flow analysis based on hybrid adaptive Particle Swarm Optimization - genetic algorithm (hybrid APSO-GA). This method developed to improve the drawbacks of genetic algorithm and Particle Swarm Optimization algorithms, specifically in data flow testing. A new fitness function is also intended on the basis of the idea of dominance relations, branch weight and branch distance to direct the search route more resourcefully. This approach is then paralleled with former algorithms. The outcomes demonstrate that hybrid APSO-GA provides superior results as related to GA, PSO, ACO, DE and hybrid GA-PSO.

Li et al [28] proposed an efficient automated test data generation method for shorten the time for test data generation. For this purpose, they first elaborated the basic PSO algorithm and the improved PSO, and as a search strategy, and then they focused on the path of trajectory similarity calculation method. Experiment result shows that their combination of two methods can more efficiently generate test data.

Lam et al [29] offered an ACO technique for creation of test sequence for state-based software testing. A directed dynamic graph is created to represent the State chart model structure of a software system under test. Using the developed ACO algorithm, a group of ants can effectively explore the graph and generate optimal test data to achieve test coverage requirement.

Zhang et al [30] proposed an approach combining the ant colony algorithm with the branch function technique to automatically generate test data based on path coverage criteria. The approach is able to search in the model built by the input domain of program, and obtains the test data which could satisfy the selected path. At last, the efficiency of the approach is validated using the triangle program.

Nayak et al [31] introduce a Particle Swarm Optimization (PSO) algorithm to test cases creation for information flow testing. They simulated both the evolutionary and swarm intelligence techniques. Their experiment result shows that PSO beats GA in 100% def-use coverage percentage.

Chen et al [32] introduce a multi-population genetic algorithm (MPGA) to generate test data for path-oriented criteria. They can use a classifier known as triangle classifier as program under test, their outcomes demonstrate that this approach can generate test data for path oriented more effectively and efficiently than simple GA based approach.

III. CONCLUSION

Test data generation is most challenging issue in software testing. This paper presented the overview of various soft computing based techniques for automatic test data Generation. The paper has given the short description of each soft computing based technique. In the literature we have found that soft computing based approaches are more appropriate than the hard computing based approaches. In this paper we have reviewed various recent soft computing based approaches to automatic test data generation namely ACO, GA, MPGA and PSO etc.

REFERENCES

[1].C.Ghezzi, M.Jazayeri, and D.Mandrioli. Fundamental of software engineer Prentice Hall, Englewood cliffs, NJ, 1991.
[2].Shahid Mahmood. A Systematic Review of Automated Test Data Generation Techniques. 2007.
[3].Mansour, Nashat; Salame, Miran. Data Generation for Path Testing. Software Quality Journal04; 12,121–136, 2004.
[4].Bogdan Korel. Automated Test Data Generation for Programs with Procedures. In proceedings of the 1996 ACM SIGSOFT international symposium on Software testing and analysis '96, 209-215/96, Vol.21, No 03, ISSTA 1996.
[5].Pargas Roy P., Harrold Mary Jean, Peck Robert R. Test-data generation using genetic algorithms. Software Testing, Verification and Reliability 9, 263–282, 1999.
[6].Akos Hajnal and Istvan Forgacs. An Applicable Test Data Generation Algorithm for Domain Errors. In proceedings of the 1998 ACM SIGSOFT international symposium on Software testing and analysis ISSTA98; Vol. 23, No. 2, 63-72/98, ISSTA 1998.
[7].Arand Gotlieb, Bernard Botella, Michel Ruether. Automatic Test Data Generation using Constraint Solving Techniques. In proceedings of the 1998 ACM SIGSOFT international symposium on Software testing and analysis98; Vol. 23, No. 2, 53-62/98, ISSTA 1998.
[8].Huey-Der Chu, John E. Dobson, and I-Chiang Liu. FAST: a framework for automating statistics-based testing. Software Quality Journal 6; 13–36, 1997.
[9]. I. Sommerville. *Software Engineering,* 5th edn (Addison-Wesley, Wokingham, 1996).
[10].Nigel Tracey, John Clark and Keith Mander. Automated Program Flaw Finding using Simulated Annealing. In proceedings of the 1998 ACM SIGSOFT international symposium on Software testing and analysis '98, 73-81/98, Vol.23, No 02, ISSTA 1998.
[11].Hitesh Tahbildar and Bichitra Kalita. Automated Software Test Data Generation: Direction Of Research. International Journal of Computer Science & Engineering Survey (IJCSES) Vol.2, No.1, Feb 2011.
[12] Bogdan Korel. Automated Software Test Data Generation. IEEE Transactions on Software Engineering. VOL. 16 NO 8. August 1990.
[13].Deepa Chauhan and Akanksha Sehgal. Automated test data generation using soft computing techniques. International Journal of Advanced Research in Computer Engineering & Technology (IJARCET) Volume 4 Issue 4, April 2015.
[14].Nirmal Kumar Gupta and Mukesh Kumar Rohil. Improving GA based Automated Test Data Generation Technique for Object Oriented Software. 3rd IEEE International Advance Computing Conference (IACC) , 249-253, 2013.
[15].Priyanka Chawla, Inderveer Chana and Ajay Rana. A novel strategy for automatic test data generation using soft computing technique. Springer, 346-363, 2015.
[16].Gongzheng Lu. An Approach to Generating Test Data for EFSM Paths Considering Condition Coverage. Electronic Notes in Theoretical Computer Science 309, 13–29, 2014.
[17].Neelam Gupta, Aditya P. Mathur and Mary Lou Soffa. Generating Test Data for Branch Coverage. IEEE, 219-227, 2000.
[18].Dunwei GONG and Yan ZHANG. Generating Test Data for both Path Coverage and Fault Detection Using Genetic Algorithms. Springer, 822-837, 2013.
[19].Xiangjuan Yao and Dunwei Gong. Genetic Algorithm-Based Test Data Generation for Multiple Paths via Individual Sharing. Computational Intelligence and Neuroscience, 2014.
[20].Sanjay Singla, Dr. Raj Kumar and Dr. Dharminder Kumar. Natural Computing for Automatic Test Data Generation Approach Using Spanning Tree concept. Elsevier, 929-939, 2016.

[21].Raquel Blanco, Javier Tuya and Belarmino Adenso-Díaz. Automated test data generation using a Scatter Search approach. Elsevier, 708-720, 2008.
[22].Moataz A. Ahmed and Irman Hermadi. GA-based multiple paths test data generator. Elsevier, 3107-3124, 2008.
[23].Mohammad Mehdi, S. Mohammad Reza, Reza Akbari, S. Ehsan Beheshtian and S. Parsa Badiei. On the Performance of EvoPSO: a PSO Based Algorithm for Test Data Generation in EvoSuite. 2nd Conference on Swarm Intelligence and Evolutionary Computation (CSIEC), 129-134, 2017.
[24].Esmaeel Nikravan, Farid Feyzi and Saeed Parsa. Enhancing Path-Oriented Test Data Generation Using Adaptive Random Testing Techniques.
[25].Li Jiao, Ziming Zhu and Xiong Xu. Improved Evolutionary Generation of Test Data for Multiple Paths in Search-based Software Testing. IEEE, 612-620, 2017.
[26].Mao yang hong and Lin ruo Qin. Application of Dynamic Program Slicing Technique in Test Data Generation. Elsevier, 355-360, 2017.
[27].Sumit Kumar, Dilip Kumar Yadav and Danish Ali Khan. A Novel Approach to Automate Test Data Generation for Data Flow Testing Based on Hybrid Adaptive PSO-GA Algorithm. Int. J. Advanced Intelligence Paradigms, Vol. 9, Nos. 2/3, 2017.
[28].Cui Huanhuan, Chen Li, Zhu Bian and Kuang Halei. An Efficient Automated Test Data Generation Method. International Conference on Measuring Technology and Mechatronics Automation, 453-456, 2010.
[29].Huaizhong Li and C. Peng Lam. Software Test Data Generation using Ant Colony Optimization. International Journal of Computer, Electrical, Automation, Control and Information Engineering Vol: 1, No: 1, 2007.
[30].Kewen Li, Wenying Liu and Zilu Zhang. Automatic Test Data Generation Based On Ant Colony Optimization. Fifth International Conference on Natural Computation, 216-220, 2009.
[31].Narmada Nayak and Durga Prasad Mohapatra. Automatic Test Data Generation for Data Flow Testing Using Particle Swarm Optimization. Springer, 1-12, 2010.
[32].Yong Chen and Yong Zhong. Automatic Path-oriented Test Data Generation Using a Multi-population Genetic Algorithm. Fourth International Conference on Natural Computation IEEE, 566-570, 2008.