Search based structural Test data Generations: A survey/ A current state of Art

Ruchika Aggarwal, Nanhay Singh

Abstract— The most significant phase of development cycle of a software focuses on testing. Testing requires set of test cases which further helps in finding the program faults. This process is in itself a tedious task, so to ensure that this process becomes more feasible some optimization algorithms are to be applied. Meta-heuristic search techniques gives way to have different coverage criterion based on which we generate the test cases. This paper addresses the current state of art in the field of structural test data generation.

Index terms- Search-based, Meta-heuristic, Test data generation, Coverage criterion

1 INTRODUCTION

Software testing aims at revealing program faults by executing the software with selected inputs and evaluating the correctness of the program behavior and output against the expected ones [1].*Search based software testing* was first proposed by Miller et al [2] in 1976. Search based software testing comes under the broad domain of search based software engineering. *Search based software engineering* (SBSE) was first introduced by Harman and Jones in the year 2001. *Search based structural test data generation* is based on the concept of optimization using meta-heuristic search techniques having coverage criterion based on the structure of the Program under test (PUT). Some of the vastly applied optimization algorithms include hill climbing, simulated annealing and genetic algorithm to name a few.

Broadly, search based techniques has been applied to a plethora of testing problems, some of which are as follows: functional testing, non functional testing, service based, exception testing, combinatorial testing, SPL, temporal testing, integration testing, regression testing, stress testing, interaction testing, mutation testing, security-safety testing, test prioritization and state machine testing.

Test data generation which, when done manually by individuals is tedious and incurs a major chunk of the total cost and indubitably it is an error prone task. It is also interesting to know that as the major chunk of the total cost incurred on software projects is assigned to software testing; similarly it would not be hard to believe that approximately the similar proportion of papers of SBSE are Search based software testing.In order to procure the papers relevant for the work conducted, we have navigated our search through the online repositories which includes the reputed publishers like IEEE explore digital library, Springer Online Library, Elsevier Online Library, ACM Portal and Wiley having “Test Data Generation” and “Search based” and “Meta-heuristic search” keywords in their title or abstract. Further,we have excluded the papers which do not apply meta-heuristic search techniques. The study is restricted to structural testing. There are a lot of extensive studies on SBST but our research motivation lies in the fact that what are the criterias based on which studies are conducted to analyze the factors that guide them.

This paper intends to summarize and classify the current state of art in the field of Search Based Structural Test Data Generation. It also focuses on:

i) Classification of the coverage criterion
ii) Need of an appropriate criteria for maximum coverage
iii) Open challenges in the field of SBSTDG
iv) Overview of the recent tools used in the field of SBSTDG

This study further tries to unveil myths related to coverage criteria, currently followed in the research of SBSTDG

2 RELATED WORK

The recent advancements in the field of SBSTDG is backed up by an extensive research work which dates back as early as 2001, wherein Harman and Jones introduced SBSE to advocate testing goals can be

achieved by the application of optimization algorithms(MHST).The early surveys done specifically on Search Based software test data generation was first presented by McMinn in the year 2004, in which the author has very clearly and aptly discussed the basic need of applying meta-heuristic search techniques (MHST) for automation in contrast to the earlier done laborious tasks done manually by the testers. Till 2004, this was potentially the most promising survey done in this field. The author had successfully covered all the possible area on which the MHST have been applied, particularly the coverage of structures of PUT as part of SBSTDG.Therest of the conducted surveys in the field of SBTDG are by Afzal et al(2009) zeroing on non- functional properties of TDG. Harman et al (2009), this survey covers the broader perspective of search based software engineering Ali et al (2010) is based on detailed empirical study on Search based test data generation.

Gotlieb in 1990 had discussed two different approaches for automatic test data generation:1) Path oriented2) Goal oriented .As this paper shows, the problem of automatically generating test inputs is hard. For example, even the most basic activities, such as seeking to cover a branch in the code involve reachability questions that are known to be undecidable in general (Weyuker, 1979). The testing community has therefore focused on techniques that seek to identify test sets that cover near optimal sets of branches in reasonable time. Many of these techniques are covered in other sections of this paper. This section is concerned with the area of Search-Based Software Testing (SBST). SBST is a branch of Search-Based Software Engineering (SBSE) (Harman and Jones, 2001), in which optimization algorithms are used to automate the search for test data that maximises the achievement of test goals, while minimising testing costs. There has been much interest in SBST, leading to several recent surveys. This section presents some emerging challenges and open problems for the development of this exciting research agenda. SBST is the process of generating test cases (or often the inputs of test cases) using search-based algorithms, guided by a fitness function that captures the current test objective.

SBST has been applied to a wide variety of testing goals including structural (Harman and McMinn, 2010; McMinn et al., 2012; Michael et al., 2001; Tonella, 2004), functional (Wegener and Bühler, 2004), non-functional (Wegener and Grochtmann, 1998) and state-based properties (Derderian et al., 2006). Search-based approaches have been developed to address a wide and diverse range of domains, including testing approachesbased on agents (Nguyen et al., 2009), aspects (Harman et al., 2009),interactions (Cohen et al., 2003), integration (Colanzi et al., 2011; Briand et al., 2002), mutation (Harman et al., 2011; Zhan and Clark, 2005), regression (Walcott et al., 2006; Yoo et al., 2009), stress (Grosso et al., 2005) and web applications (Alshahwan and Harman, 2011).

In all approaches to SBST, the primary concern is to define a fitness function (or set of fitness functions) that capture the test objectives. The fitness function is used to guide an algorithm, which searches the space of test inputs to find those that meet the test objectives. Because any test objective can, in principle, be re-cast as a fitness function, the approach is highly generic and therefore widely applicable (as the foregoing list of testing applications demonstrates). There are many different search-based algorithms to choose from, though much of the literature has tended to focus on evolutionary algorithms (Harman, 2011).

There are several surveys of these aspects of SBST (Afzal et al., 2009; Ali et al., 2010; Harman et al., 2009; McMinn, 2004, 2011). In these surveys the reader can find more detailed treatments of the work on SBST for non-functional properties (Afzal et al., 2009), Empirical evidence regarding SBST (Ali et al., 2010), as well as overviews of techniques (Harman et al., 2009; McMinn, 2004; McMinn et al., 2012). Therefore, in this section, we do not seek to provide yet another ‘overview’ of SBST. Rather, we focus on some of the exciting and challenging avenues that lie ahead for future work in this rapidly growing research and practitioner community. In describing these future directions we seek to consider work which is already underway as well as more ‘blue skies’ directions for open challenges that could yield major breakthroughs. For example, we consider work underway on co-evolution and management of oracle cost as well as work on hybridising SBST with other test data generation techniques, such as Dynamic Symbolic Execution (DSE), a topic covered in more detail elsewhere in this paper. The oracle problem is important because the automation of testing requires automation of the checking of outputs as well as the generation of inputs. Co-evolution is interesting and

important because it fits so well the way in which the testing process operates, as we shall see. In all these emerging areas we can expect more work in the immediate future. We also consider open challenges such as the problem of migrating from generation of test cases to generation of testing strategies using search and optimising the insight that can be gained from SBST.

3 STRUCTUAL TEST DATA GENERATION

Structural test data generation can be broadly classified into: Static and Dynamic. Static STDG: it is based on the structure of the program under test. It comprises of symbolic execution and domain reduction. Constraint based testing encompasses domain reduction. Dynamic STDG: it necessitates the execution of the program under test.

Based on this concept the test data generation is categorized on several coverage criteria like: Branch coverage, path coverage, decision statement coverage, data flow coverage , control flow coverage, statement, condition coverage etc. Random testing, application of local search, object oriented approach, chaining approach are intentionally excluded out of this paper.

4 SEARCH BASED STRUCTUAL TEST DATA GENERATION

Test data generation in software testing is the process of identifying program input data which satisfy selected testing criteria.[Korel 1190]. The approach has been applied to several types of testing, including functional and non functional testing, mutation testing, regression testing, test case prioritization, and interaction testing. However the most studied form of the search based testing has been *structural test data generation.[Lakhotia 2009]*

Search Based optimization algorithms has been extensively applied for the automation of test suites in various testing techniques. Genetic algorithm are applied for statement path, branch and fault based testing. Simulated annealing is applied for exception condition testing.

5 SEARCH BASED META-HEUSITIC SEARCH ALGORITHM

The basic phenomena in any approach is to frame the design in a way that it achieves the aim of test data generation by applying meta heuristic techniques on the input domain of the program under test.They are a set of generic algorithms that are used to find optimal or near optimal solution to problems that have large complex search spaces [5]. Ali et al [6] in their systematic review have classified MHS into local and global meta-heuristic search techniques. The necessity of applying MHS on test data generation lies in the fact that generation of test data is an optimization problem. Thus, we require to pragmatically re-formulate the test cases generated so as to get the best optimal set from the current huge set of possibly generated test cases.

Now, simple search strategies may be easily handled by applying the local MHS, but the complex search problems can't be easily tackled as they might step into the local optima. Therefore, to avoid getting stuck in the local minima problem, global MHS has the effective and more promising results in complex search problems [7]. The most common MHS algorithm applied in Search based software testing are evolutionary algorithms, simulated annealing, hill climbing, ant colony optimization and particle swarm optimization.

Local MHS: Hill climbing, Random search, Tabu search, Iterated local search

Global MHS: Evolutionary algorithms which includes Genetic algorithm, genetic programming and evolution strategies.

The study can be further categorized for structural testing as statement testing, Branch based testing, Path testing and Fault based testing.Phil McMinn in his survey paper 2004 has classified the techniques on the basis of objective function under the following category: 1) coverage oriented, 2) coverage oriented, 3) branch-distance oriented, 4) control oriented and 5) combined approach.According to Harman et al 2009, the most vastly studied field is believed to be structural testing under the domain of SBSE. He also addressed that the most considered coverage criteria is branch coverage and been targeted in the maximum no of papers. He has also clearly stated for the formulation of any search based optimization problem we need to define :

1) the choice of the representation of the problem,

2) the definition of the fitness function.[Harman et al(2009)]

Fitness Function: Any meta heuristic search technique requires a formulation of an objective function, the main goal of a fitness function is to direct the search in to promising and unexplored input domain to find the optimal solution. These two ingredients can lead us to locate optimal or near optimal solution.

1) Hill climbing: It is based on local search.Despite the local maxima problem, hill climbing is a simple technique that is both easy to implement and surprisingly effective in many SBSE problems.[
2) Simulated annealing: Simulated Annealing (SA) can be thought of as a variation of hill climbing that avoids the local maxima problem by permitting moves to less fit individuals. The approach has found application in several problems in SBSE .
3) Evolutionary testing: This optimization algorithm falls under the category of global search.The applicability of evolutionary algorithms in the domain of structural testing is known as evolutionary testing. Tonella, initiated to juxtapose both search based and object oriented software testing problems.Search based test data generation in cordinance with the object oriented systems gets complicated due to internal states as the objects are inherently state based.

   3.1) Genetic algorithms: Genetic Algorithms (GAs) use concepts of population and of recombination [245]. Of all optimisation algorithms, genetic algorithms have been the most widely applied search technique in SBSE, though this may have largely been for historical or sociological reasons, rather than scientific or engineering reasons.
4) Memetic Algorithm : It is a hybrid optimization algorithm comprising of both global and local search. It was introduce by Wang and Jeng. It requires smaller population size in contrast to genetic algorithms. It a combination of genetic algorithm and hill climbing.

There is another type of search known as ***random search***, it doesn't holds a fitness function to it therefore it is not considered as a meta heuristic search technique

7 FLOWCHART OF GENERIC SCHEMANTIC VIEW TO BE CONSIDER LATER

Type of testing	Structural testing,model based testing, mutation testing, temoral testing, exception testing, regression testing, configuration testing, stress testing etc.
Fitness Function objectives	Max cohesion, minimum coupling
Coverage criterion	Branch coverage, data flow coverage, decision coverage, path coverage, statement coverage.
Search techniques	Hill climbing, greedy search, genetic algorithm, simulated annealing, Tabu search, memetic algorithm etc.
Properties considered	Functional , non functional

Table 1 Classification scheme for SBTDG literature.

Lakhotia et al. proposed a multi objective approach for structural test data generation keeping in mind the objective of maximum coverage

Reference	Fields	Tools	Application
Arcuri et al[]	Object orient	Evosuite	Parameter Tuning

	ed TDG		
Shamshiri et al[]	Object oriented TDG	Evosuite	GA and Random Search
Boyapati et al[]	TDG	Korat	Java framework
Briand et al[]		SimCoTest	
Fraser et al[]	TDG	Evosuite	Branch coverage
Galleoti et al[]	TDG	Evosuite + dse	Branch coverage, NA
Lakhotia et al[]	TDG	AUSTIN	Branch coverage
Lakhotia et all[] 2010	TDG	CUTE, AUSTIN	Branch coverage
Lakhotia 2013	TDG	AUSTIN	
Mairhofer[] 2011	TDG	RuTeG	Statement coverage
Michael et al[]2001	TDG	GADGET	Code coverage
Baresi and Miraz	TDG	TestFul	Statement coverage, Branch coverage
Lakhotia et al[] 2008	TDG	CUTE	Branch Coverage
Harman et al 2007	TDG	IGUANA	Multi objective Branch Coverage

Table 2 Survey on advancements of SBSTDG based on Tools

S.No	Test Level	Fault Type (test adequacy criteria)	MHS Type	Reference
1.	White box	Data flow	PSO	Nayak et al[]
2.	Gray box	Statement coverage	EA (novelty search)	Boussaa et al[]
3.	White box	Path coverage	GA	Ahmed at al[]
4.	Unit testing	Branch coverage	Not specified	Alshraideh et al[]
5.	White Box(Not specified)	Branch coverage	Symbolic Execution(Alternating variable and GA)	Baars at al[]
6.	NA	Data flow coverage	SA, GA	Bueno et al[]
7.	NA	Condition-Decision Coverage	PSO	Jia et al[]
8.	NA	Path Coverage	Cellular automata	Bhasin et al[]
9.	White box	Branch coverage	HC, EA, MA	McMinn et al[]
10.	White box	Multi-objective Branch coverage	EA, GA	Harman et al[]2007
11.	NA	Branch coverage	Scatter search	Blanco et al
12.	White box	Branch coverage	GA	Cao et al[]
13.	White box	Branch coverage	Memetic algorithm	Fraser et al[] 2015
14.	Data flow testing	Data flow coverage	GA	Girgis et al[]
15.	White box	Path Coverage	GA, SA	Ghani et al[]
16.	White Box	All-uses Criteria	Hybrid GA and PSO	Girgis et al[]
17.	White Box	Path Coverage	Constraint solving	Gotlieb
18.	White Box	Path coverage	EA	McMinn et all[]2006
19.	White Box	Multi objective	EA, GA	Harman et all[]

		branch coverage		
20.	White Box	Branch Coverage	EA,GA, HC	Harman and McMinn[]2007
21.	White Box	Branch Coverage	EA	McMinn et al[] 2012
22.	White Box	Not mentioned	PSO	Jiang et al[] 2013
23.	White Box	Code coverage	GA, PSO	Koleejan et all[]
24.	White Box	Not specified	EA	Lammermann[] 2005
25.	White Box	Path Coverage	GA	Lin and yeh 2001
26.	White Box	Code coverage, branch, statement	GA	Michael et al[]2001
27.	White Box	Branch coverage	GA	Miller et all[]2005
28.	White Box	Branch coverage	GA	Paachauri and srivastava[]
29.	White box	Control flow	GA	Pargas et al[]1999
30.	White Box	Edge/condition coverage	GA	Sofokleous and andreou[] 2007
31.	White Box	Feature pairwise coverage	GA	Wang et al
32.	White Box	Control flow and data flow coverage	EA	Wegener et al[]
33.	White Box	Branch coverage	PSO, GA	Windisch et al[]
34.	White Box	Data flow coverage	NA	Vivanti

Table 3 Papers based on application of MHST in SBSTDG

- *Ruchika Aggarwal is currently working as assistant professor in Aryabhatta college, University of Delhi and pursuing Ph.D degree program in Computer Science, GGSIPU, Delhi India, PH-9971918446. E-mail: ruchika.aggarwal1989@gmail.com*
- *Nanhay Singh is currently working as associate professor in AIACTR , Delhi, India, PH-9971594480. E-mail:nsingh1973@gmail.com*

CONCLUSION

Although search based structural test data generation promises extremely good results by finding out the optimized results of test data that helps in meeting structural coverage criteria such as brach coverage, path coverage, decision coverage etc. But afer carrying out this extensive survey: the need of implementing the most optimum stopping criteria in different domains of fault type coverages in accordance with the used meta-heuristic search technique was analysed. The diiferent tools for test data generation have already been proposed by different researchers like Austin, Cute etc.

REFERENCES

1. Harman, Mark, YueJia, and Yuanyuan Zhang. "Achievements, open problems and challenges for search based software testing." *Software Testing, Verification and Validation (ICST), 2015 IEEE 8th International Conference on*. IEEE, 2015.

2. McMinn, Phil. "Search-based software test data generation: A survey."*Software Testing Verification and Reliability* 14.2 (2004): 105-156.

3. Lakhotia, Kiran, Mark Harman, and Phil McMinn. "A multi-objective approach to search-based test data generation." *Proceedings of the 9th annual conference on Genetic and evolutionary computation*. ACM, 2007.

4. Harman, Mark, et al. "The impact of input domain reduction on search-based test data generation." *Proceedings of the the 6th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on The foundations of software engineering*. ACM, 2007.

5. Tracey, Nigel James. *A search-based automated test-data generation framework for safety-critical software*. Diss. University of York, 2000.

6. Harman, Mark, et al. "Optimizing for the number of tests generated in search based test data generation with an application to the oracle cost problem."*Software Testing, Verification, and Validation Workshops (ICSTW), 2010 Third International Conference on*. IEEE, 2010.

7. Alshraideh, Mohammad, and Leonardo Bottaci. "Search-based software test data generation for string data using program-specific search operators."*Software Testing, Verification & Reliability* 16.3 (2006): 175-203.

8. Harman, Mark. "Automated test data generation using search based software engineering." *Automation of Software Test, 2007. AST'07. Second International Workshop on*. IEEE, 2007.

9. Ali, Shaukat, et al. "A search-based OCL constraint solver for model-based test data generation." *Quality Software (QSIC), 2011 11th International Conference on*. IEEE, 2011.

10. Tracey, Nigel, et al. "A search-based automated test-data generation framework for safety-critical systems." *Systems engineering for business process change: new directions*. Springer London, 2002. 174-213.

11. Zhan, Yuan, and John Clark. "Search based automatic test-data generation at an architectural level." *Genetic and Evolutionary Computation–GECCO 2004*. Springer Berlin Heidelberg, 2004.

12. McMinn, Phil, et al. "Input domain reduction through irrelevant variable removal and its effect on local, global, and hybrid search-based structural test data generation." *Software Engineering, IEEE Transactions on* 38.2 (2012): 453-477.

13. McMinn, Phil, MuzammilShahbaz, and Mark Stevenson. "Search-based test input generation for string data types using the results of web queries."*Software Testing, Verification and Validation (ICST), 2012 IEEE Fifth International Conference on*. IEEE, 2012.

14. Cao, Yang, Chunhua Hu, and Luming Li. "Search-based multi-paths test data generation for structure-oriented testing." *Proceedings of the first ACM/SIGEVO Summit on Genetic and Evolutionary Computation*. ACM, 2009.

15. Sagarna, Ramón, and Xin Yao. "Handling constraints for search based software test data generation." *Software Testing Verification and Validation Workshop, 2008. ICSTW'08. IEEE International Conference on*. IEEE, 2008.

16. Ghani, Kamran, John A. Clark, and Yuan Zhan. "Comparing algorithms for search-based test data generation of matlab® simulink® models."*Evolutionary Computation, 2009. CEC'09. IEEE Congress on*. IEEE, 2009.

17. Windisch, Andreas. "Search-based test data generation from stateflowstatecharts." *Proceedings of the 12th annual conference on Genetic and evolutionary computation*. ACM, 2010.

18. Varshney, Sapna, and Monica Mehrotra. "Search based software test data generation for structural testing: a perspective." *ACM SIGSOFT Software Engineering Notes* 38.4 (2013): 1-6.

19. Vivanti, Mattia, et al. "Search-based data-flow test generation." *Software Reliability Engineering (ISSRE), 2013 IEEE 24th International Symposium on*. IEEE, 2013.

20. Mairhofer, Stefan, Robert Feldt, and Richard Torkar. "Search-based software testing and test data generation for a dynamic programming language."*Proceedings of the 13th annual conference on Genetic and evolutionary computation*. ACM, 2011.

21. Malhotra, Ruchika, and Manju Khari. "Heuristic search-based approach for automated test data generation: a survey." *International Journal of Bio-Inspired Computation* 5.1 (2013): 1-18.

22. Khatun, S., et al. "A Random search based effective algorithm for pairwise test data generation." *Electrical, Control and Computer Engineering (INECCE), 2011 International Conference on*. IEEE, 2011.

23. Maragathavalli, P. "Search-based software test data generation using evolutionary computation." *arXiv preprint arXiv:1103.0125* (2011).

24. Mairhofer, Stefan. "Search-based software testing and complex test data generation in a dynamic programming language." (2008).

25. Zidoune, Walid, and ToufikBenouhiba. "Targeted adequacy criteria for search-based test data generation." *Information Technology and e-Services (ICITeS), 2012 International Conference on*. IEEE, 2012.

26. Kotelyanskii, Anton, and Gregory M. Kapfhammer. "Parameter tuning for search-based test-data generation revisited: Support for previous results."*Quality Software (QSIC), 2014 14th International Conference on*. IEEE, 2014.

27. Sofokleous, Anastasis A., and Andreas S. Andreou. "Dynamic Search-Based Test Data Generation Focused on Data Flow Paths." *ICEIS (2)*. 2008.

28. Hermadi, Irman, Chris Lokan, and R. Sarker. "Dynamic stopping criteria for search-based test data generation for path

testing." *Information and Software Technology* 56.4 (2014): 395-407.

29. Shaukat, A., et al. "A search-based OCL constraint solver for model-based test data generation." *Proceedings of the 11th Int. Conf. on Quality Software (QSIC 2011)*. 2010.

30. Kanmani, S., and P. Maragathavalli. "Search-based software test data generation using evolutionary testing techniques." *International Journal of Software Engineering (IJSE)* 1.5 (2010): 10-22.

31. HARMAN, M., et al. "J. WEGENER, J. 2007. The Impact of Input Domain Reduction on Search-based Test Data Generation." *ESEC-FSE'07: Proceedings of the the 6th Joint Meeting of the*.

32. Harman, Mark. "Automated test data generation using search based software engineering (keynote)." (2007): 2.

33. Gotlieb, Arnaud, NadjibLazaar, and YahiaLebbah. "Towards Constraing-Based Local Search for Automatic Test Data Generation." *Software Testing Verification and Validation Workshop, 2008. ICSTW'08. IEEE International Conference on*. IEEE, 2008.

34. Harman, Mark, et al. "Domain Reduction for Search-Based Test Data Generation." (2007).

35. Malhotra, Ruchika, et al. "Comparison of Search based Techniques for Automated Test Data Generation." *International Journal of Computer Applications* 95.23 (2014).

36. Pachauri, Ankur. "Software test data generation using metaheuristic search based techniques." (2014).

37. Harman, Mark, et al. "Reducing Oracle Cost in Search Based Test Data Generation."

38. Wei, Y.; Meyer, B. &Oriol, M. (2010), Is Branch Coverage a Good Measure of Testing Effectiveness?, *in* Bertrand Meyer & Martin Nordio, ed., 'LASER Summer School' , Springer, , pp. 194-212 .

39. Michael, Christoph, and Gary McGraw. "Automated software test data generation for complex programs." *Automated Software Engineering, 1998. Proceedings. 13th IEEE International Conference on*. IEEE, 1998.

[1]