ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Vol. 4, Issue 1, January 2016

Survey on Automatic Test Data Generation Tools and Techniques for Object Oriented Code

Pranali Prakash Mahadik, Prof.Dr. D. M. Thakore

Research Scholar, Dept. of Computer Engg., Bharati Vidyapeeth Deemed University, Collage of Engineering, Pune, India.

Professor, Dept. of Computer Engg., Bharati Vidyapeeth Deemed University, College of Engineering, Pune, India.

ABSTRACT: Automating the process of software Testing can reduce the cost of testing. This survey gives overview of test data generation tools and techniques. There areseveral methods which are capable of generating test input automatically based on the source code of the program under test. Survey paper mentioned the description in brief about test data generation technique like Random selection, Search-based techniques and Symbolic execution based techniques. Survey focuses on the problem of how to choose the most appropriate tool that will fulfill developer requirements consisting of level of automation, cost requirement, language support, etc. The aim of survey is to provide the categorization of all tool based on methods used, Type like commercial or academic, programming language support, input, output, testing technique and general information of tools, target domain, that useful for a developer to make a decision on which tool to use.

KEYWORDS: Test date generation; Test cases; Random testing; Search-based testing; automated unit testing.

I. INTRODUCTION

Software testing is process of find the faults and it is important phase of software development life cycle which is very time consuming and tedious process, accounting for more than 50% of the total software cost and resources [1]. The test data generation is most expensive part of software testing.
In software testing selecting the test input is challenging tasks. There are some methods which are capable to automatically generating test input based on the source code of the program under test. Several methods used for test generation based on code, for example:

- Random selection: random selection also called adaptive random test generation give good results.
- Code annotations: in this method if the code is annotated with pre/post conditions, these can drive selection of test input.
- Search-based techniques: search algorithms like genetic algorithm are used for representing coverage of some kind.
- Symbolic execution based techniques: it includes dynamic symbolic execution, concolic testing, etc. [2].

Automating the process of Software testing can reduce the overall cost. Generally, developer perform unit testing. There are many tools are available for automating the unit testing process, Which perform task like test data generation and test cases generation detecting known bugs in the code, etc. tools. Some examples available are Jtest, C++ tester,AgitarOne,codeProAnalytix,Randoop,JWalk,Jcute,CATG,EvoSuite,JTExpert,Symbolic PathFinder,T3,JCrasher,PET,GRT,etc.these tool have different aspects like programming language support, technique used, input type, Output produced.

It is difficult for a developer to choose appropriate tool, as each tool have different efficiency and effectiveness level. Efficiency and effectiveness level of tool are two different things; efficiency of tool is related to robustness and execution time of tool, while the effectiveness of the tool be determined by different factors such as type of errors it can find, test cases type or output it can produce and quality of generated output. Therefore, this survey work help developers in the process of selecting appropriate tool based on their requirements and develop some new tools which have additional properties and find out more errors.

The survey paper begins with introduction of the various test data generation techniques. Then it attempts to give an overview of available academic research project and commercial tools which provide automated support for unit testing. Then the categorization of tools provided based on different parameters like methods used ,Type like commercial or academic, programming language support, input, output, testing technique, domain, etc. use of automated unit testing tool reduce the task of developer at some extent because it is very difficult to work on both development and testing of software module [3].

In this survey we studied 15 automated test data generation tools and categorized them based on the literatures available. This survey provides maximum research information automation tool could help developers to find and to choose suitable tool. the list of all current available automated unit testing tools which is provided together with brief description of some additional properties that can provide help to developers or organizations developer to choose appropriate tool and also helpful for development of new tool.

II. Methods used for code-based test generation

In this section code-based test generation methods based on the available literature describe its functioning in brief.

A. *Random Test Generation:*

Random test generation process also known as adaptive test generation it produce good result in test data generation. It can select input randomly until inputs are found [4]. Random test generation is easy and fast method to generate test data. It can generate massive number of test cases automatically. It introduces randomness in software testing process which reflects irregularity of system environment. This technique may not have acceptable test coverage. This technique may fail to find test data to satisfy the requirements because information for the test requirements is not merged. The several disadvantages of this method are such as it is suitable only for simple programs. it does notgenerate test cases using available information, and many sets of values may produce same observable behavior.Random testing is mostly used in combination with other techniques. Mostly to produce an early test case from which other can be used to generate subsequent input values. Random test data generation is used most of the time used as a benchmark because it is easy to implement.

B. *Symbolic Execution based techniques :*

Symbolic execution as a tool for test-data generation was first proposed by James King in 1976 [5]. Symbolic execution based technique include dynamic symbolic execution, concolic testing, etc. Symbolic test data generation techniques [6, 7] assign symbolic values to the variables and generate algebraic expressions for the numerous constraints in the program. A constraint solver is used to find out a solution for these expressions that full fills a test requirement. Symbolic execution means executing a program with symbol instead of concrete values. Assignment statements are denoted as functions of their symbolic arguments, and constraints on symbolic values are expressed in form of conditional statements. Symbolic execution can be used for many purposes, like as bug detection, verification of program, debugging, maintenance, and localization fault [8].

C. *Search-based techniques:*

Search algorithm is used to represent some kind of coverage. Search based engineering used for task like metaheuristic search technique like Tabu search, simulated annealing, genetic algorithm, etc. Search based software engineering has much applications in test data generation because the generation of software tests is an undesirable problem [9,10].search based software engineering is application of optimization technique in test data generation and solving software engineering problems. Generally optimization refers to finding the best possible solution among all available solution. The task of test case generation is transformed into an optimization problem and it can be solved with simulated annealing or evolutionary algorithms which are meta-heuristic search techniques. Input domain of system under test represent as search space from this the test data fulfil the test objective under consideration is generated. Evolutionary testing is generally used to increase the quality of testing and provide high level of automation which reduce the cost in overall system development process. In various case studies, it has been proved that evolutionary testing capable to improve the testing process efficiency and effectiveness significantly. McMinn provided overview of different evolutionary testing [11].Advantage of SBST is that its result shows the efficiency of approach Disadvantage is it require large search space.

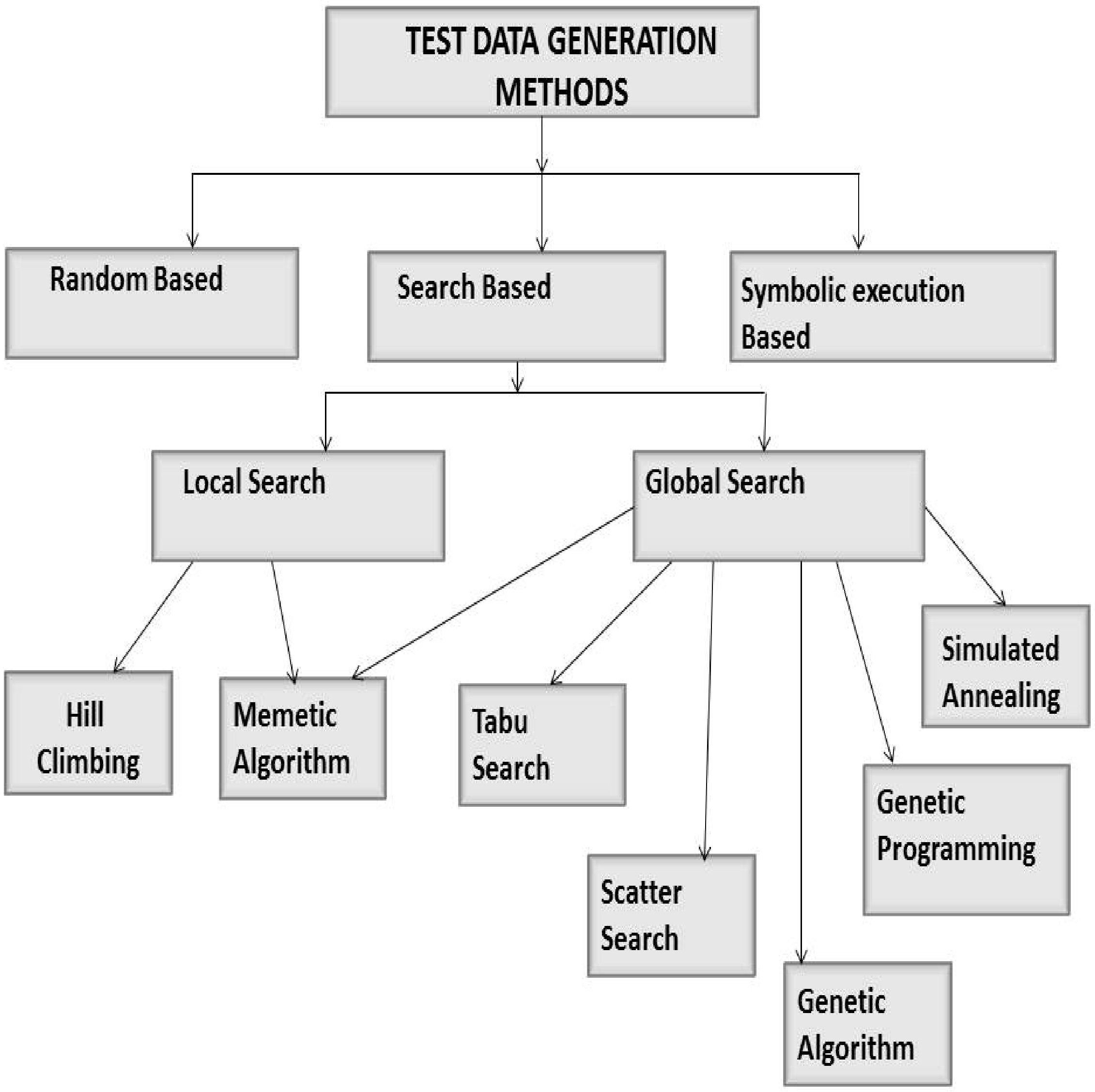

Fig.1. Test data generation methods

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Vol. 4, Issue 1, January 2016

III. AUTOMATED TEST DATA GENERATION TOOLS

We have studied 15 automated test generation tools based on the available literature which describe it in brief.

A. *JTest:*

Jtest introduce by Parasoft [12] is a solution to unit testing which has automated Unit test cases generation capabilities also perform static analysis, regression testing, code review, runtime error detection, and Design by contract also allows the developers to automatically generate unit tests for new projects as well as for legacy code.Jtest has the capability to examine Java code and unit test cases which are automatically generate. It generates the test cases based on the input method and return value in your actual class. All the test classes generated will have same class name but appended with Jtest keyword. Jtest also provide options, on the extent to which the java code requires to be analyzed for generating the test cases. If 'through checking' option is selected, Jtest produces more number of test case methods by using different mixtures of input. Jtest also reused existing test cases. Test case parameter values can be taken from an external excel file or can be internally created within Jtest. The excel file which internally produced will have different combination of values like it may be auto generated or manually added which required for a specific test case. Jtest also provide facility of performing regression testing. We can also get details of code coverage along with the test case results upon execution.

B. *Parasoft C++ test:*

Parasoft's C++ test [13] test introduce in1999.it is commercial software quality improvement tool. C++ test supports checking coding standard, static analysis, dynamic or runtime analysis, regression testing and code review automation, detection of error at run time and unit test generation. For non-primitive types C++ test always selects a constructor. Therefore, C++test is not able to produce test input that is required to be NULL. Furthermore, C++test does not attempt to modify the object any more i.e., no extra methods are called after the constructor to change the state of object. The C++test User's Guide [13] does not describe anything about the object creation strategy.

C. *AgitarOne:*

Agitar Technologies released AgitarOne in 2004 a commercial tool based on academic research results test-input generation for Java. AgitarOne is automated Junit test generation tool. It works on classes with various input data and creates observations that represent the behavior of the unit by which developers can convert to assertions.AgitarOne can be used as standalone IDE from the command-line. AgitarOne allows all developers like novice or expert to get started quickly and easily with unit testing. AgitarOne help to create a complete set of unit tests with much less manual effort.

D. *CodePro Analytix:*

In 2010 Google, Inc. bought CodePro AnalytiX from Instantiations, Inc. It flawlessly integrates into Rational Developer, any Eclipse development environment or IBM WebSphere Studio [14]. CodePro AnalytiX is a tool that supports or provides help to improve the quality of Java programs The CodePro Junit Test Case Generation tool allows you to automate the formation of complete Junit regression test cases. For each method under test CodePro AnalytiX produces input values for all available parameters, determines combinations, validates the result, computes the result of method execution under test and generates test cases in Junit format.

E. *Randoop:*

Pacheco et al. introduced in 2007 RANDOOP [15, 16]. Randoop is automated unit test generator tool for Java. It automatically generates unit tests java classes which are in Junit format. Randoop generates unit tests automatically using feedback-directed random test generation of unit tests for object-oriented programs by using method sequences.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Vol. 4, Issue 1, January 2016

Randoop executes the method sequences which capture the program and to catch bugs.Randoop, used at Microsoft internally, which used by a team of test engineers to find errors in a core. Randoop has created tests that find previously unknown errors even in mostly used libraries like Sun and IBM's JDKs. Randoop technique is highly effective because it combine randomized test data generation and test execution result.

F. *Jwalk:*

Anthony Simons create Jwalk which is a unit testing tool for the Java. Jwalk supports unit testing paradigm called Lazy Systematic which is based on the two notions of lazy specification first is the ability to infer the evolving specification of a class on dynamic analysis and second is systematic testing. JWalk help programmer to generate unit tests that sufficiently cover the test java class's state space. Using JWalk compares against expert manual testing (using Junit). This is because the tool discovers the test class systematically, and suggests test-cases that a programmer usually forgets, The JWalk tool [17] generates test sets for one CUT at a time by using specification-based test generation algorithms that verify the complete algebraic structure, or transitions and high-level states of the CUT [18].

G. *Jcute:*

Concolic Unit Testing Engine (CUTE) tool is developed at the University of Illinois by Koushik Sen which handles multi-threaded programs and pointer operations CUTE which is targeting C. The jCUTE [19] tool is a unit testing engine for Java. The Java Concolic Unit Testing Engine (jCUTE) automatically produces unit tests for Java programs. Concolic execution integrates randomized concrete execution done with automatic constraint and symbolic execution. jCUTE uses symbolic execution which allows discern input that lead to different execution paths and randomized concern execution helpful to overcome constraint solver limitations, like inability to analyze system calls or solve general systems of non-linear integer equations by this combination, jCUTE is generate test cases that execute differentexecution paths in Java programs. jCUTE explore race condition and deadlock by systematic schedule.

H. *CATG:*

CATG is concolic unit testing tool for Java programs. Concolic testing maintains a symbolic state and concrete state and dynamically performs symbolic execution, while the program is executed on some input values. Where all variables map at the concrete state to their concrete values and the symbolic state only maps variables that have non-concrete values. CATG sometime integrate with TesMa, which is model-based testing tool which automatically generate test cases from formal design documents which are provided in the form of database table definitions, diagrams of process flow and screen definitions. TesMa able to creates Java programs by using these design documents. CATG does concolic testing to generate suitable database for these Java Programs and test application by taking required test input [20].

I. *EVOSUITE:*

EvoSuite is result of research project by Dr.Gordon Fraser and Dr. Andrea Arcuri was originally created in 2010. EvoSuite is automatic test generator tool that automatically generates test cases for java programs that achieve high code coverage [21]. EvoSuite generates and optimizes whole test suites to satisfy a coverage criterion. EvoSuite do branch coverage and mutation testing as test objectives.

J. *GRT:*

Guided Random Testing i.e., GRT is an automatic test generator tool for Java programs. GRT guide run time test generation process by static and dynamic program analysis. It uses feedback-directed random testing. Static analysis extracts knowledge of domain from the software under test which is input for run-time test generation for improving systematic test coverage in generation phase done by dynamic analysis. GRT itself can be improved in a number of ways. It is tempting to incorporate symbolic execution techniques to achieve higher code coverage, especially in the face of complicated branches. Simple specialized treatments, such as handling less visible code, may be surprisingly effective. We also plan to enhance the test oracle of GRT. Currently, GRT focuses on leveraging program analysis to obtain high code coverage, using simple oracles, such as software crashes and exceptions [22].

K. *JTExpert :*

JTExpert can automatically generate a whole test suite for satisfying the branch coverage criterion on a given Java class. Test data generation for object oriented programming is challenging because of features like abstraction, encapsulation and visibility which does not give direct access to some part of source code ,so to overcome the problem of generating test cases for whole these parts JTExpert tool is developed which is search based tool used to achieve high code coverage [23]. JTExpert is available as an executable jar file. It takes as inputs a Java file or a Java project directory and for every java class under test automatically produces a test-data suite in Junit format.

L. *Symbolic Path Finder:*
Symbolic path finder (SPF) tool do symbolic execution of Java byte code. SPF is freely available open source automatic test generation tool. SPF usually used for systematic generation of test cases that achieve high testing code coverage and for checking safety violations, like assert and concurrency errors, in programs with user specified inputs. SPF handles inputs and operations on Booleans, integers, real's, strings, and complex data structures as well as solves complex mathematical constraints. SPF tool done symbolic execution with constraint solving and model checking for automated test case generation it also detect errors in java byte code [24].

M. *T3:*
T3 automated unit testing tool for java classes.it can randomly generates sequence of method calls for the java class which are given as input.T3 detect unexpected exception but if class has assertion written by you, then violations to those will also be caught. T3 is fast and able to generate thousands of test sequences in few second.it can run form command line [25].

N. *Jcrasher:*
Jcrasher tool is done automatic testing for java code. Jcrasher use instance generator approach i.e., it can create instance of different types to test the behavior of public method under random data by examining the type of information of java classes and constructing code fragment. Jcrasher also able to detect bugs.Jcrasher analyze methods and determine the space and time allotted for tested method's parameter.it is completely automatic and provide analysis of method, find out bug, produce test files for Junit. Jcrasher combined with Eclipse IDE [26].

O. *PET:*
Partial Evaluation-based Test Case Generator for Byte code (PET) converts Java byte code to corresponding CLP code and generates test cases from it.PET is automatically generating test cases from Java byte code which is depending on the technique of partial evaluation. The system take input as a byte code program and a set of optional parameters, including a description of a coverage criterion and produces as output a set of test cases which provide assurance that the selected coverage criterion is achieved.PET performs two partial evaluations (PE). The first PE decompiles the Java byte code program into an equivalent CLP (Constraint Logic Programming) counterpart The advantage of decompiling to CLP is that it freely handles most of the required constraints . The second PE generates a test-case from the CLP program. PET tool is freely available on its web site. PET is also applicable to larger programs [27].

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Vol. 4, Issue 1, January 2016

IV. CATEGORIZATION OF AUTOMATIC TEST GENERATION TOOL FOR OBJECT ORIENTED CODE

This section describes the categorization of automation supported unit-testing tools for object oriented code.

Table 1. Categorization of automatic test generation tool for object oriented code

SR. NO	TOOL	INSTITUTION	LAST MODIFICATION ON	TYPE	METHOD	INPUT CODE TYPE	REQUIRED INPUT	OUTPUT	TESTING TECHNIQUE	DOMAIN	LICENSE
1.	Jtest	Parasoft	2015	Commercial	Static analysis	JAVA	Source code	Junit test cases	White box testing technique	Desktop and server edition	Commercial tool with 14 days validity
2.	C++ Test	Parasoft	2010	Commercial	Random	C++	Source and binary code	Unit tests	Static analysis	Desktop	Commercial 14 days validity
3.	AgitarOne	Agitar Technologies	2015	Commercial	Random	Java	Source code	Junit tests	Observation derived testing	Desktop	Commercial 30 days validity
4.	CodePro Analytix	Google Inc	2010	Commercial	Random	Java	Source code	Junit test cases	White box Testing technique	Desktop	Apache license 2.0
5.	Randoop	MIT CSAIL	2015	Open source software	Random	JAVA	Source code	Unit test suite	Feedback directed random testing	Desktop	MIT license
6.	Jwalk	University of *Sheffield*	2013	Academic research	Random	JAVA	Source code	Test report	lazy systematic testing	Desktop	JWalk License
7.	JCUTE	University of *Illinois* By kaushik sen	2006	Academic research	Symbolic execution	JAVA	Source code	Unit tests	Search based concolic testing	Desktop	free
8.	CATG	University of *Illinois* By kaushik sen	2015	Open source software	Symbolic execution	JAVA	Source code	Test cases	Random testing	desktop	Open source BSD license
9.	EvoSuite	Research project by dr.gordon Fraser and dr.andrea arcuri	2015	Academic research and open source software	Search Based	JAVA	Source code	Junit tests	Search based approach	desktop	LGPL License
10.	GRT	Lei ma,cheng,hiroyuki,Johannes,rudolf	2015	Academic research	Random	JAVA	Source or byte code	Junit Test cases	Static and runtime analysis	desktop	Free
11.	JTExpert	Abedelilah sakti,gilles,yann-gael	2015	Academic research	Search Based	JAVA	Source code	Whole test suite	Search Based	Desktop	Free
12.	Symbolic PathFinder	Microsoft	2015	Open source software	Symbolic Execution	JAVA	Software	Test cases	Model checking	Web application	Used at NASA Free for research
13.	T3	Wishnu prasetya	2015	Open source software	Random	JAVA	Source code	Test suite	Dynamic testing	desktop	GPL (general public license)version 3
14.	Jcrasher	Christoph and yannis	2007	Academic research	Random	JAVA	Source code	Test File	Random or robust testing	desktop	Apache Licence
15.	PET	Elvira and miguel	2011	Academic Research	Symbolic Execution	JAVA	Byte code	Test cases	White box testing	desktop	Free

V. CONCLUSION AND FUTURE WORK

In this paper we represented different techniques and tools for test data generation tool for object oriented data and also brief look on it. We can also provide categorization of these tools with respect to some parameters only because contains some commercial tools, due to accessibility problem to these tools we could not provide the detailed

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Vol. 4, Issue 1, January 2016

information about them.This survey evaluates efficiency and effectiveness of each tool, which is helpful to develop new tool with same properties as several other with some extra properties.

The idea to develop new efficient and effective tool by merging properties of some tools of similar kind that can find more range of errors and improve the code coverage for object oriented code by considering features of object oriented languages.

REFERENCES

1. Myers G.J., “The art of software testing”, 1979,John Wiley and Sons.
2. http://www.code-based test generation overview and tools.html.
3. S. H. Aljahdali, A. S. Ghiduk, and M. El-Telbany, “The limitations of genetic algorithms in software testing”, 2010,IEEE ACS International Conference on Computer Systems and Applications (AICCSA), pp. 1–7.
4. Edvardso,”A Survey on Automatic Test data generation‟, 1999,second conference on computer science and engineering Vol.2, No.1, pp.343- 351.
5. J. C. King, “Symbolic execution and program testing“,1976,ACM,vol. 19, no. 7, pp. 385–394. [cited at p.22, 24].
6. Howden, W.E.,”Symbolic testing and the DISSECT symbolic evaluation system‟,1977, IEEE Transactions on Software Engineering vol.3, no. 4, pp. 266-278.
7. John Clarke, Mark Harman, Bryan Jones ,“The Application of Metaheuristic Search techniques to Problems in Software Engineering‟,2000, IEEE Computer Society Press Vol.42, N’o.1, pp.247-254.
8. L. A. Clarke and D. J. Richardson, Applications of symbolic evaluation”, 2000, Journal of Systems and Software , 5(1):15–35, 1985.
9. Tao Feng, KasturiBidarkar,”A Survey of Software Testing Methodology”, 2008,vol.25, no-3, pp.216-226.
10. Voas, J.M, Morell, J. and Miller, K.W,”Predicting where faults can hide from testing‟1991, IEEE vol: 8, pp, 41-48.
11. McMinn,”Search Based Software Test Data Generation:A survey”, 2004,Journal on Software Testing, Verification, and Reliability vol.14, no.2, pp.105-156.
12. Parasoft,”Jtest:Java unit testing and code compliance-Parasoft”,2007. http://www.parasoft.com/jsp/products/home.jsp?product=Jtest. [cited at p.35].
13. Paraosft,” Parasoft C++test User's Guide”, 2010.
14. Google Inc.,” Codeproanalytix user guide”,http://developers.google.com/java-dev-tools/codepro/doc/.].
15. Carlos Pacheco and Michael D. Ernst,” Randoop: Feedback-Directed Random Testing for Java” 2007, In OOP-SLA 2007: Conference on Object Oriented Programming Systems Languages and Applications, ACM.
16. Carlos Pacheco, Shuvendu K. Lahiri, Michael D. Ernst,and Thomas Ball,” Feedback-directed Random Test Generation”, 2007, In Proceedings of the 29th International Conference on Software Engineering, Minneapolis, MN, USA, 2007. IEEE.
17. Simons, A. J. H.,” JWalk: a tool for lazy systematic testing of Java classes by design introspection and user interaction”, 2007, Software. Eng, 14 (4), Springer USA (2007), 369-418.
18. N. Smeets and A. J. H. Simons, “Automated Unit Testing with Randoop, JWalk and muJava versus Manual JUnit Testing”,2009, in Department of Mathematics and Computer Science in University of AntWerp.
19. K. Sen, “Cute :A concolic unit testing engine for c and java” ,Web, 2007.http://osl.cs.uiuc.edu/˜ksen/cute/. [cited at p.39, 44].
20. Tanno, X Zhang, T Hoshino, K Sen ,”TesMa and CATG: automated test generation tools for models of enterprise applications”, Proceedings of the 37th International Conference on Software Engineering.
21. Fraser, G., Arcuri, “ EvoSuite: automatic test suite generation for object-oriented software” In Proceedings of the 19th ACM SIGSOFT symposium and the 13th European conference on Foundations of software engineering. pp. 416–419. ACM 2011.
22. Lei Ma, CyrilleArtho, Cheng Zhang, Hiroyuki Sato, Johannes Gmeiner and Rudolf Ramler,” GRT: Program-Analysis-Guided Random Testing”, to appear in Int'l Conf. Automated Software Engineering 2015, Nov. (Distinguished Paper Award), 212--223.
23. Sakti A., Pesant G., Gueheneuc Y.G,” Instance generator and problem representation to improve object oriented code coverage”, IEEE Trans. on Software Engineering 41(3) 2015.
24. C. S. Pasareanu et al, "Symbolic PathFinder: integrating symbolic execution with model checking for Java bytecode analysis," 2013, Automated Software Engineering 20:3, pp 391-425. DOI: 10.1007/s10515-013-0122-2.
25. W.B. Prasetya,”T3i: A Tool for Generating and Querying Test Suites for Java”, 10th Joint Meeting of the European Software Engineering Conference (ESEC) and the ACM SIGSOFT Symposium on the Foundations of Software Engineering (FSE), ACM, 2015.
26. Christoph Csallner and Yannis Smaragdakis,”JCrasher: an automatic robustness Tester for Java”, 2000, John Wiley & Sons, Ltd.
27. E. Albert, M. Gomez-Zamalloa, and G. Puebla, "PET: A Partial Evaluation-based Test Case Generation Tool for Java Byte code”, 2010 , In Proc. PEPM, ACM, pp. 25-28. DOI: 10.1145/1706356.1706363.