International Journal of Computer Applications (0975 – 8887)
Volume 57– No.13, November 2012

A Review of Automatic Test Cases Generation

Ebrahim Shamsoddin-Motlagh
Computer Engineering Department, Faculty of Engineering, Science & Research Branch, Islamic Azad University, Tehran, Iran

ABSTRACT
Manual testing is hard and time consuming and it maybe impossible for large systems or tester mistake in the test. The software testing is the rising cost of activities in development software and the test case generation is important activity in software testing. Hence researches performed to automate testing such as automatic test case generation. This paper reports a survey of recent research to generate test case automatically. Those are presented from UML based, graph based, formal methods, web application, web service, and combined. Those needed future researches are presented.

General Terms
Testing, Test Case, Generate, Automatically, Dynamic Testing, Review

Keywords
Automatic Test Case Generation, test automate

1. INTRODUCTION
Software systems have been expanded in many sections of our life such as transportation, health, and media. Software reliability is very important and software testing is a way for verifying the right to work of a software system. Software testing is one of the most expensive activities and time consuming in development of software. A well tested software system will be validated by the customer before acceptance [1]. The effectiveness of this verification and validation process depends upon the number of errors found and rectified before releasing the system. This in turn depends upon the quality of test cases generated.
Test case generation is the most important part of the test. To cut down cost of manual testing and to increase reliability of the test, researchers have attempted to automate it and the test case generation automatically has been attempting of it [1]. Automatic test case generation is a good method for dynamic testing.
The paper is structured as follows. Section 2 a review related work of automatic test case generation. Finally, Section 3 outline suggests future research steps.

2. AUTOMATIC TEST CASE GENERATION

2.1 UML based testing
The number of researches performed system testing with using UML diagrams.

The research [2] is proposed a framework for automatic test case generation in black box testing for Java methods. In reference [3] is provided an approach to convert UML sequence diagram into a sequence diagram graph (SDG) and completed SDG nodes with different information necessary in combination of the test vectors, the SDG is used to generate test cases.

In a sample research generates test cases used to logic control and information of the state diagrams UML. In reference [5], is provided an approach to convert UML use case diagram into a use case diagram graph (UDG) and UML sequence diagram into a SDG, then combined UDG and SDG to generate a system testing graph (STG), finally traversed STG to generate test cases [4].

In the paper [6] is proposed the behavior models of the test sequences on statechart based and FSM based. The production environment is produced of several test methods, such as covering the switch, procedures DS and UIO for FSM and produced test coverage criteria in SCCT family for statechart based.

2.2 Graph based
Numbers of existing researches performed to generate test cases automatically with system graph.

In sample work is proposed an algorithm on random paths (WalkTest) to solve the structural problem of test case generation. WalkTest obtains high quality solutions in the situations of random function calls walk operator repeats. The first WalkTest turns solutions in gray code instead of the usual binary code, and turns test goals to minimize the objective function. Then selects and edits possible solutions on a continuous search space of in the gray code walk operator. Next step sorts the test objectives (objective) by analyzing the related control flow graph with reduction cost in run-time based. Finally WalkTest collects statistical data and test cases [7].

In the paper [8] is proposed used to rules to generate test cases automatically and test policies is expressed, the algorithm completes to search the tree for the reuse of test cases.

In a study to test automation, generates test case dynamically on genetic algorithms based. The system architecture consists of two subsystems [9]. Subsystems are program analyzer system (BPAS) and test case generator (ATCGS). BPAS subsystem analyzes the system under test. ATCGS subsystem searches in the input space and gets set of test cases optimized of test coverage criteria based on the edge / state.

2.3 Formal methods
Functional testing automatically is a potentially effective approach for software reliability, but it is a challenge due to the automation difficulties deriving adequate test cases and test oracle from informal or semi-formal specifications that often used to in practice [10]. Numbers of existing researches performed to generate test cases automatically with formal methods.

A paper is proposed a decompositional approach to automatic test case generation of model-based formal specifications. The approach offers a functional scenario based test method, a set of test case generation criteria, a set of test case generation algorithms automatically and test oracle is well defined for test result analysis automatically [10].

In the study [11] is showed a static analysis feedback and unit test case generation framework (KUnit). Abilities shown in the framework include:

1) Full coverage extension methods to reach all the small stack structure
2) Competitive tool to improve the coverage, the size of test set and time is generated test set.

In the paper [12] is proposed an approach to generate test case on Object-Z specification based of a class, it is formalization based of the test case generation.

2.4 Web application

Numbers of existing researches performed to generate test cases automatically for web based applications.

In the research [13] is proposed the tow-phase approach to generate test cases automatically by analyzing web application structure. that is defined the dependence related to data dependence and control dependence in the web application, and detected their relationship with the source code, and improved to generate test cases by analyzing the results.

In reference [14], with respect to the test of multi-tier web have challenges and a defect in the activity of the intermediate layers, maybe spread to other layers, and web applications are often continuously, that is proposed an approach on the inter-connection dependence model to generate the web pages sequences are more prone to failure.

2.5 Web service

Web Services deploy among distributed applications. To ensure the quality of the services are published, bound, invoked and integrated at runtime, test cases are generated automatically and the test executed, monitored and analyzed at runtime [15]. For this reason Numbers of investigations have been active for test automation, in their attempted to automate process or processes of testing. Number of existing researches performed unit testing on WSDL file and generated test cases automatically [15, 16, and 17].

In the research [15] proposes an approach to generate test cases in the web services automatically. First that parses the WSDL file and transforms the DOM tree structure, then generates test cases from test data and test operation. In the research [16] proposes a formal approach to generate test case automatically for the web service single operation. In the research [17] focuses on Abstract Test Suite and Executable Test Suite to analyze WSDL and proposes to test WSDL based automatically.

Numbers of existing researches performed integration testing on BPEL file in the service oriented systems with operations graph.

In the research [18] proposes an approach to generate test cases on the process definition model a web service. First it creates a graph of processes in Web services. Then follow what the user is doing with the system, and finally provides the needed test cases.

In the research [19 and 20] proposes an approach to generate test cases for BPEL process Stream X-machine (SXM) based. Those define SXM [19]: "The SXM describes a system as a finite set of states, an internal memory and a number of transitions between the states." It used to SXM to convert BPEL to activity flow (activity graph).

You can automatically generate test cases from web service automata (WSA), WSA can be used for define the operational logic in BPEL [21].

The research [22] is provided an approach to design test cases based on functional properties of high-level business process model.

Researches [23 and 24] implemented an approach identifies the changes by performing control flow analysis and comparing the paths in a new version of composite service with those in the old use to extensible BPEL flow graph (XBFG).

TASSA is a framework for testing and validation in functional and non-functional behavior of service-based applications [25]. TASSA provides end-to-end testing of Service layer, Service Composition and coordination and business process of service-based applications. Another tool for automated testing is presented WSOTF [26]. WSOTF is an automatic conformance testing tool with timing constraints from a formal specification of web services composition that is implemented by an online testing algorithm.

In a study is expressed test approach described in BPEL web service composition [27]. The paper [28] is proposed an approach to generate a testbed for service-oriented systems. That takes a mobility model of nodes in the network which the accessed services are deployed.

The study [29] is expressed a framework is supporting tool for generating and executing web service requests and analyzing the subsequent request-response pairs automatically.

The study [30] is proposed an approach combines accessibility technologies for accessing and controlling graphical applications (GAPs) in a uniform way with a visualization mechanism that enables nonprogrammers to generate test cases for web services by performing drag-and-drop operations on graphical user interface (GUI) elements of GAPs.

Approaches described in Table 1 with different levels of test coverage service based system testing. In a service may be provided in the composition of services used to BPEL file, the reason test on the parts in this table is on the BPEL file also put some unit testing capabilities.

2.6 Combined

Numbers of existing researches propose to combine the number of test case generation methods and achieving better method for automatic test case generation.

A research is proposed an approach combines techniques the random test case generation and the invariant extraction and achieved test case generation and selection automatically. The result of this test program is smaller than random test case generation [31].

A study is proposed an approach focuses on the theory of Markov chains and a combination of functions to get test cases for unexpected failure. It compares set of test cases generated by the two approaches and combines them occurs a more efficient of model-based testing system environment [32].

Table 1. Web service based systems testing approaches at levels testing

Approaches in web service testing	Level Testing			
	Unit Testing	Integration Testing	Regression Testing	Non-Functional Testing
[15, 16, 17, 30]	To generate test cases by WSDL	-	-	-
SAT Solver [18]	-	To generate test cases by processes and user activity	Use to save test cases	-
[27, 29]	-	To generate test cases for processes business	-	-
SXM [19, 20], WSA [21], Tabu [22]	BPEL testing	To generate test cases for BPEL service	-	-
TASSA [25]	Layer service testing	to generate test cases of orchestration and BPEL service	-	Layers testing, Coordination and service composition
WSOTF [26]	Analysis WSDL	To generate test cases of specification system	-	-
XBFG [23,24]	-	-	To select test cases	-
[28]	-	-	-	To generate specification in mobile system model

2.7 Summarized

The paper [1] is presented a survey on automatic test case generation approaches that are found literature in the before 2005 year. Several approaches are proposed to generate test cases include mainly random, path-oriented, goal-oriented and intelligent approaches and expressed researches on them. These approaches can be classified to static and dynamic. Static approaches are often symbolic execution based, whereas dynamic approaches obtain the necessary data by run of the program under test.

Table 2 shows relation any type of automatic test case generation and levels of software testing. UML Based testing can be used to all levels. Web service testing in the unit testing level can be used to in the integration testing level, and in the integration testing can be used to system testing and vice versa because system can be created by BPEL process and BPEL process used to integration of system.

Table 2. Type of automatic test case generation and levels of software testing

Automatic test case generation	Level testing		
	Unit testing	Integration testing	System testing
UML Based	[2, 3, 4, 5, 6]	[2, 3, 4, 5, 6]	[2, 3, 4, 5, 6]
Graph Based	[7, 9]	[8]	[8]
Formal method	[11, 12]	[10]	[10]
Web application	-	[13, 14]	[13, 14]
Web service	[15, 16, 17]	[15, 16, 17] [18…30]	[18…30]

3. Conclusions

The part of 2 this paper was expressed automatic test case generation at six parts include: UML based, graph based, formal methods, web application, web service, and combined. Those proposed approaches to generate test cases. In the UML based approaches need UML file, and analyze that file, then create a graph and generate test cases for cover graph. Graph based use to white box testing and need source file of program. Web application and web service approaches use to specific software and generate test cases. In the combined approaches achieve better method in combination of the number of test case generation methods.

Future works will propose specific approaches for specific software in the software logic or improve existing approaches for specific software. Another create test cases generation framework for general software or specific software. Future work can be integration of available tools.

4. ACKNOWLEDGMENTS

The author would like to thank specially Dr. Seyed Hasan Mirian Hossienabadi who has extended his support for successful completion of this paper.

5. REFERENCES

[1] Prasanna, M., Sivanandam, S.N., Venkatesan, R., Sundarrajan, R. (2005). A SURVEY ON AUTOMATIC TEST CASE GENERATION. Academic Open Internet Journal, Volume 15, http://www.acadjournal.com/.

[2] Hu, Y.T., & Lin, N.W. (2010). Automatic Black-Box Method-Level Test Case Generation Based on Constraint Logic Programming. Computer Symposium (ICS), 977-982. Doi: 10.1109/COMPSYM.2010.5685369

[3] Sarma, M., Kundu, D., Mall. R. (2007). Automatic Test Case Generation from UML Sequence Diagrams. 15th International Conference on Advanced Computing and Communications, 60-65. Doi: 10.1109/ADCOM.2007.68

[4] Samuel, P., Mall, R., Bothra, A.K. (2008). Automatic test case generation using unified modeling language (UML) state diagrams. The Institution of Engineering and Technology, IET Softw, Vol. 2, No. 2, pp. 79–93. Doi: 10.1049/iet-sen:20060061

[5] Sarma, M., & Mall, R. (2007). Automatic Test Case Generation from UML Models. 10th International Conference on Information Technology, 196-201. Doi: 10.1109/ICIT.2007.26

[6] Santiago, V., Vijaykumar, N. L., Guimaraes, D. (2008). An Environment for Automated Test Case Generation from Statechart-based and Finite State Machine-based Behavioral Models. IEEE International Conference on Software Testing Verification and Validation Workshop (ICSTW'08).

[7] Xuan, J., Jiang, H., Ren, Z., Hu, Y., Luo, Z. (2009). A Random Walk Based Algorithm for Structural Test Case Generation. 2nd International Conference on Software Engineering and Data Mining (SEDM), 583-588. Retrieved from ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5542856

[8] Liu, Z., Gu, N., Yang, G. (2005). An Automate Test Case Generation Approach Using Match Technique. The Fifth International Conference on Computer and Information Technology (CIT'05), 922-926. Doi: 10.1109/CIT.2005.64

[9] Sofokleous, A. A., Andreou, A. S. (2008). Automatic, evolutionary test data generation for dynamic software testing. The Journal of Systems and Software, 81, 1883–1898. Doi:10.1016/j.jss.2007.12.809

[10] Liu, S., & Nakajima, S. (2010). A Decompositional Approach to Automatic Test Case Generation Based on Formal Specifications. 2010 Fourth International Conference on Secure Software Integration and Reliability Improvement, 147-155. Doi: 10.1109/SSIRI.2010.11

[11] Deng, X., Robby, Hatcliff, J. (2007). Kiasan/KUnit: Automatic Test Case Generation and Analysis Feedback for Open Object-oriented Systems. Testing: Academic and Industrial Conference - Practice And Research Techniques, 3-12. Doi: 10.1109/TAIC.PART.2007.32

[12] Ashraf, A., & Nadeem, A. (2006). Automating the Generation of Test Cases from Object-Z Specifications. Proceedings of the 30th Annual International Computer Software and Applications Conference (COMPSAC'06), 101-104. Doi: 10.1109/COMPSAC.2006.120

[13] Tung, Y.H., Tseng, S.S., Lee, T.J., Weng, J.F. (2010). A Novel Approach to Automatic Test Case Generation for Web Applications. 10th International Conference on Quality Software, 399-404. Doi 10.1109/QSIC.2010.33

[14] Dai, Z., & Chen, M.H. (2007). Automatic Test Case Generation for Multi-tier Web Applications. 9th IEEE International Workshop on Web Site Evolution (WSE), 39-43. Doi: 10.1109/WSE.2007.4380242

[15] Bai, X., Dong, W., Tsai, W. T., Chen, Y. (2005). WSDL-Based Automatic Test Case Generation for Web Services Testing. 2005 IEEE International Workshop on Service-Oriented System Engineering (SOSE'05), 1-6. Doi: 10.1109/SOSE.2005.43

[16] Ma, C., Du, C., Zhang, T., Hu, F., Cai, X. (2008). WSDL-Based Automated Test Data Generation for Web Service. International Conference on Computer Science and Software Engineering, 731-737. Doi: 10.1109/CSSE.2008.790

[17] Dong, W. (2009). Testing WSDL_based Web Service Automatically. World Congress on Software Engineering, 521-525. Doi: 10.1109/WCSE.2009.133

[18] Radhakrishnan, K., & Podorozhny, R. (2009, February 16). Automatic test case generation for web service processes using a SAT solver (Report Number TXSTATE-CS-TR-2009-13). https://digital.library.txstate.edu/bitstream/handle/10877/2581/fulltext.pdf

[19] Ma, C., Wu, J., Zhang, T., Zhang, Y., Cai, X. (2008) Automatic Test Case Generation for BPEL Using Stream X-Machine. International Journal of u- and e- Service, Science and Technology, 27-36. Retrieved from http://www.sersc.org/journals/IJUNESST/vol1_no1/papers/04.pdf

[20] Ma, C., Wu, J., Zhang, T., Zhang, Y., Cai, X. (2008). Testing BPEL with Stream X-machine. International Symposium on Information Science and Engieering, 578-582. Doi: 10.1109/ISISE.2008.201

[21] Zheng, Y., Zhou, J., Krause, P. (2007, September). An Automatic Test Case Generation Framework for Web Services. JOURNAL OF SOFTWARE, VOL. 2, NO. 3, 64-77. Retrieved from http://epubs.surrey.ac.uk/1975/1/fulltext.pdf

[22] Bakota, T., Beszédes, Á., Gergely, T., Gyalai, M. I., Gyimóthy, T., Füleki, D. (2008). Semi-Automatic Test Case Generation from Business Process Models. This research was supported in part by the Hungarian national

grants RET-07/2005, OTKA K-73688 and TECH_08-A2/2-2008-0089 SZOMIN08. Retrieved from http://www.inf.u-szeged.hu/~beszedes/research/bakota09_semiautomatic.pdf

[23] Li, B., Qiu, D., Ji, S., Wang, D. (2010). Automatic Test Case Selection and Generation for Regression Testing of Composite Service Based on Extensible BPEL Flow Graph. 26th IEEE International Conference on Software Maintenance in TimiSoara, Romania, 1-10. Doi: 10.1109/ICSM.2010.5609541

[24] Li, B., Qiu, D., Leung, H., Wang, D. (2012). Automatic test case selection for regression testing of composite service based on extensible BPEL flow graph. The Journal of Systems and Software Volume 85, Issue 6, 1300–1324. Doi:10.1016/j.jss.2012.01.036

[25] Ilieva, S., Pavlov, V., Manova, I. (2010). A Composable Framework for Test Automation of Service-Based Applications. 2010 Seventh International Conference on the Quality of Information and Communications Technology, 286-291. Doi: 10.1109/QUATIC.2010.54

[26] Cao, T.D., Felix, P., Castanet, R. (2010). WSOTF An Automatic Testing Tool for Web Services Composition. Fifth International Conference on Internet and Web Applications and Services, 7-12. Doi: 10.1109/ICIW.2010.9

[27] Lallali, M., Zaidi, F., Cavalli, A., Hwang, I. (2008). Automatic Timed Test Case Generation for Web Services Composition. Sixth European Conference on Web Services, 53-62. Doi: 10.1109/ECOWS.2008.14

[28] Bertolino, A., Angelis, G.D., Lonetti, F., Sabetta, A. (2008). Automated Testbed Generation for Service-oriented Mobile Applications. 34th Euromicro Conference Software Engineering and Advanced Applications, 321-328. Doi: 10.1109/SEAA.2008.33

[29] Martin, E., Basu, S., Xie, T. (2007). Automated Testing and Response Analysis of Web Services. IEEE International Conference on Web Services (ICWS), 647-654. Doi: 10.1109/ICWS.2007.49

[30] Conroy, K. M., Grechanik, M., Hellige, M., Liongosari, E. S., Xie, Q. (2007). Automatic Test Generation From GUI Applications For Testing Web Services. Software Maintenance, IEEE International Conference on ICSM, 345-354. Doi: 10.1109/ICSM.2007.4362647

[31] Zeng, F., Cao, Q., Mao, L., Chen, Z. (2009). Test Case Generation based on Invariant Extraction. 5th International Conference Wireless Communications, Networking and Mobile Computing. WiCom '09, 1-4. Doi: 10.1109/WICOM.2009.5302578

[32] Nakao, H., & Eschbach, R. (2008). Strategic usage of test case generation by combining two test case generation approaches. The Second International Conference on Secure System Integration and Reliability Improvement, 213-214. Doi: 10.1109/SSIRI.2008.17