International Journal of Computing and Corporate Research
ISSN (Online) : 2249-054X
Volume 5 Issue 3 May 2015
International Manuscript ID : 2249054XV5I3052015-06

THE PRAGMATIC REVIEW ON TEST CASE GENERATION FOR ANALYSIS AND EFFICIENCY OF WEB SERVER

Shweta Ahuja
M.Tech. Research Scholar
Computer Science and Engineering
Guru Nanak Institute of Technology
Mullana, Haryana, India

Shalini Kapoor
Assistant Professor
Computer Science and Engineering
Guru Nanak Institute of Technology
Mullana, Haryana, India

ABSTRACT

Test Case Generation is one of the important ask in software engineering that is having a set of conditions under which a test engineer determine whether an application, software system or one of its features is working as it was originally established for it to do. The mechanism for determining whether a software program or system has passed or failed such a test is known as a test oracle. In some settings, an oracle could be a requirement or use case, while in others it could be a heuristic. It may take many test cases to determine that a software program or system is considered sufficiently scrutinized to be released. Test cases are often referred to as test scripts, particularly when written - when they are usually collected into test suites. Monte Carlo methods or simply Monte Carlo experiments are a broad class of computational algorithms that rely on repeated random sampling to obtain numerical results. They are often used in physical and mathematical problems and are most useful when it is difficult or impossible to use other mathematical methods. Monte Carlo methods are mainly used in three distinct problem classes: optimization, numerical integration, and generating draws from a probability distribution. In this work, we have proposed to design and implement a sequence of test cases to evaluate the performance and integrity of web application so that the concurrent and high load applications can be executed without any overhead. In this work, a novel model is proposed

and shall be implemented as the future work. In the proposed model, the auto generated test cases shall be evaluated and processed by the web server and database engine and performance shall be analyzed on multiple parameters including cost, efficiency and execution time.

Keywords - Software Testing, Test Case Generation, Monte Carlo Based Test Case Generation

INTRODUCTION

Software testing is an investigation conducted to provide stakeholders with information about the quality of the product or service under test. Software testing can also provide an objective, independent view of the software to allow the business to appreciate and understand the risks of software implementation. Test techniques include the process of executing a program or application with the intent of finding software bugs (errors or other defects).

It involves the execution of a software component or system component to evaluate one or more properties of interest. In general, these properties indicate the extent to which the component or system under test:

- Meets the requirements that guided its design and development,
- Responds correctly to all kinds of inputs,
- Performs its functions within an acceptable time,
- Sufficiently usable,
- installed and run in its intended environments, and
- Achieves the general result its stakeholders desire.

REVIEW OF LITERATURE

To propose and defend the research work, a number of research papers are analyzed. Following are the excerpts from the different research work performed by number of academicians and researchers.

[1] With the growing complexity of web applications, identifying web interfaces that can be used for testing such applications has become increasingly challenging. Many techniques that work effectively when applied to simple web applications are insufficient when used on modern, dynamic web applications, and may ultimately result in inadequate testing of the applications' functionality. To address this issue, we present a technique for automatically discovering web application interfaces based on a novel static analysis algorithm. We also report the results of an empirical evaluation in which we compare our technique against a traditional approach. The results of the comparison show that our technique can (1) discover a higher number of interfaces and (2) help generate test inputs that achieve higher coverage.

[2] In this paper, the authors develop methods that use logged user data to build models of a web application. Logged user data captures dynamic behavior of an application that can be useful for addressing the challenging problems of testing web applications. Our approach automatically builds statistical models of user sessions and automatically derives test cases from these models.

We provide several alternative modeling approaches based on statistical machine learning methods. We investigate the effectiveness of the test suites generated from our methods by performing a preliminary study that evaluates the generated test cases. The results of this study demonstrate that our techniques are able to generate test cases that achieve high coverage and accurately model user behavior. This study provides insights into improving our methods and motivates a larger study with a more diverse set of applications and testing metrics.

[3] Web services applications are built by the integration of many loosely coupled and reusable services using open standards. Testing Web service is important in detecting faults and assessing quality attributes. A difficulty in testing Web services applications is the unavailability of the source code for both the application builder and the broker. This paper propose a solution to this problem by providing a formal, specification-based approach for automatically generating test cases for web services based on the WSDL input messages parts' XML Schema datatypes. Examples of using this approach are then given in order to give evidence of its usefulness. The role of the application builders and the brokers in using this approach to test Web services is also described.

[4] Modified condition/decision coverage is a structural coverage criterion requiring that each condition within a decision is shown by execution to independently and correctly affect the outcome of the decision. This criterion was developed to help meet the need for extensive testing of complex Boolean expressions in safety-critical applications. The paper describes the modified condition/decision coverage criterion, its properties and areas for further work.

[5] Pairwise testing is a specification-based testing criterion, which requires that for each pair of input parameters of a system, every combination of valid values of these two parameters be covered by at least one test case. In this paper, we propose a new test generation strategy for pairwise testing.

[6] Test case prioritization techniques schedule test cases for execution in an order that attempts to maximize some objective function. A variety of objective functions are applicable; one such function involves rate of fault detection-a measure of how quickly faults are detected within the testing process. An improved rate of fault detection during regression testing can provide faster feedback on a system under regression test and let debuggers begin their work earlier than might otherwise be possible. In this paper we describe several techniques for prioritizing test cases and report our empirical results measuring the effectiveness of these techniques for improving rate of fault detection. The results provide insights into the tradeoffs among various techniques for test case prioritization

[7] Pairwise testing has become an indispensable tool in a software tester’s toolbox.This article pays special attention to usability of the pairwise-testing technique. Most tools, however, lack practical features that are necessary for them to beused in the industry.This article pays special attention to usability of the pairwise-testing technique. Inparticular, it does not describe any radically new

method of efficient generation of pairwisetest suites—a topic that has already been researched extensively—neither does it refer toany specific case studies or results that have been obtained through this method oftest-case generation. It does focus on ways in which the pure pairwise-testing approach

[8] Metaheuristic search techniques have been extensively used to automate the process of generating test cases, and thus providing solutions for a more cost-effective testing process. This approach to test automation, often coined "Search-based Software Testing" (SBST), has been used for a wide variety of test case generation purposes. Since SBST techniques are heuristic by nature, they must be empirically investigated in terms of how costly and effective they are at reaching their test objectives and whether they scale up to realistic development artifacts. However, approaches to empirically study SBST techniques have shown wide variation in the literature. This paper presents the results of a systematic, comprehensive review that aims at characterizing how empirical studies have been designed to investigate SBST cost-effectiveness and what empirical evidence is available in the literature regarding SBST cost-effectiveness and scalability. We also provide a framework that drives the data collection process of this systematic review and can be the starting point of guidelines on how SBST techniques can be empirically assessed. The intent is to aid future researchers doing empirical studies in SBST by providing an unbiased view of the body of empirical evidence and by guiding them in performing well-designed and executed empirical studies.

[9] One of the major costs in a software project is the construction of test-data. This paper outlines a generalised test-case data generation framework based on optimisation techniques. The framework can incorporate a number of testing criteria, for both functional and non-functional properties. Application of the optimisation framework to testing specification failures and exception conditions is illustrated. The results of a number of small case studies are presented and show the efficiency and effectiveness of this dynamic optimisation-base approach to generating test-data.

[10] Software Product Lines (SPL) are difficult to validate due to combinatorics induced by variability across their features. This leads to combinatorial explosion of the number of derivable products. Exhaustive testing in such a large space of products is infeasible. One possible option is to test SPLs by generating test cases that cover all possible T feature interactions (T-wise). T-wise dramatically reduces the number of test products while ensuring reasonable SPL coverage. However, automatic generation of test cases satisfying T-wise using SAT solvers raises two issues. The encoding of SPL models and T-wise criteria into a set of formulas acceptable by the solver and their satisfaction which fails when processed "all-at-once"'. We propose a scalable toolset using Alloy to automatically generate test cases satisfying T-wise from SPL models. We define strategies to split T-wise combinations into solvable subsets. We design and compute metrics to evaluate strategies on Aspect OPTIMA, a concrete transactional SPL.

International Journal of Computing and Corporate Research
ISSN (Online) : 2249-054X
Volume 5 Issue 3 May 2015
International Manuscript ID : 2249054XV5I3052015-06

PROBLEM FORMULATION

As the domain of Software Testing and Test Case Generation is much diversified, there is lots of scope of research for the scholars and practitioners.

PROPOSED WORK

- Generation of Test Cases based on the attributes of Form in the Web Application
- Simulation and Implementation of the Fuzzy Random Values generated from the values generator
- Analysis and deep investigation of the performance of database engine and apache web server based on different file and attribute sizes / length.

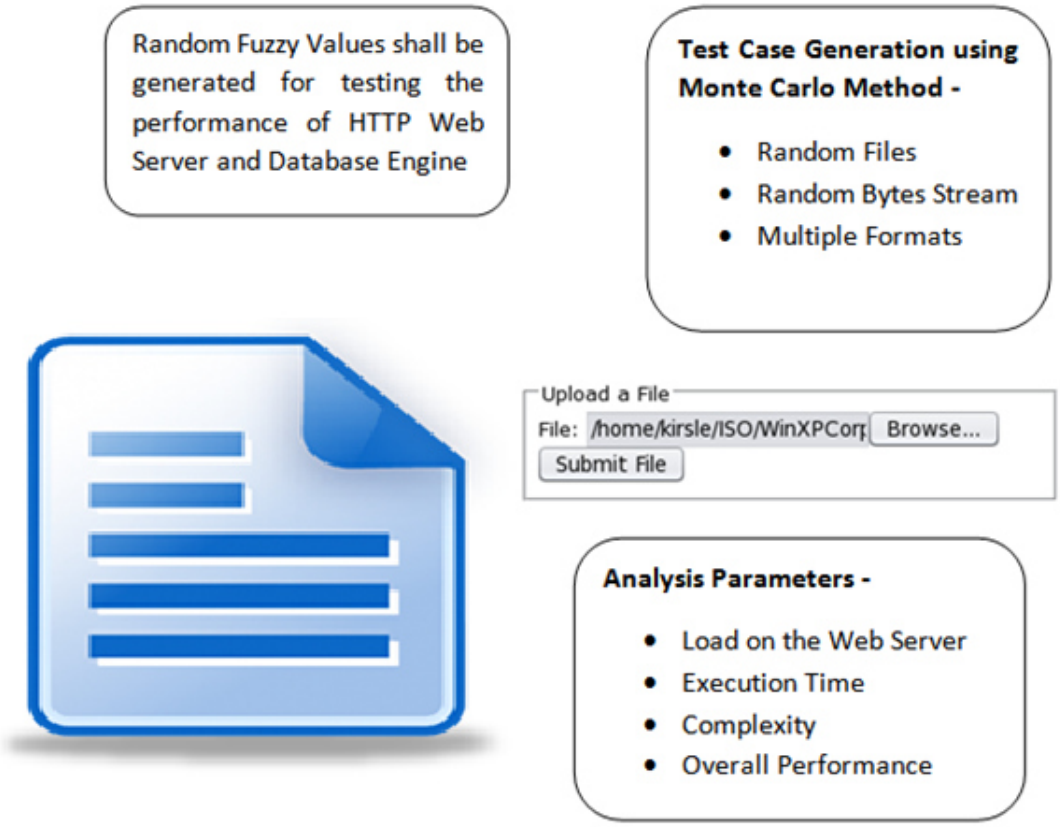

Figure 1 - Proposed Model and Aspects

The main objectives of this research are

1. To implement the existing algorithm [11]
2. In the proposed work, we will develop and deploy a simulator in PHP to generate the sequence of test cases.

3. These test cases shall have different priorities based on fuzzy random values. Each test case or file generated randomly by the simulation shall be executed on the Apache

   Web Server. Depending upon the initial attributes (Size and Name), the random priorities shall be allocated. Once the process of allocation of weights / values is done, then the actual implementation and testing of web server shall be initiated.
4. The effect and overall performance of the web based software application shall be investigated based on the priority and fuzzy random values.
5. The test cases shall be generated using monte carlo simulation that is one of the prominent method to generate the inputs and expected outcome.
6. In base paper, there is no measurement of execution time and overhead issues.
7. To measure the execution time and complexity of the existing algorithm in the base paper
8. To propose and implement the novel algorithm that will be better than the classical approach in terms of enhanced test case generation.

CONCLUSION

In this work, a novel algorithmic approach and model is proposed that will generate the test cases for web application based form. In this approach, the form values shall be inserted and processed by the script and overall performance of the system including web server and database engine shall be evaluated based on multiple parameters. Using this approach, the tolerance level and threshold of database engine as well as server can be set with effective evaluation.

REFERENCES

[1] Rayadurgam, S., & Heimdahl, M. P. E. (2001). Coverage based test-case generation using model checkers. In Engineering of Computer Based Systems, 2001. ECBS 2001.Proceedings. Eighth Annual IEEE International Conference and Workshop on the (pp. 83-91). IEEE.

[2] Ali, S., Briand, L. C., Hemmati, H., &Panesar-Walawege, R. K. (2010). A systematic review of the application and empirical investigation of search-based test case generation.Software Engineering, IEEE Transactions on, 36(6), 742-762.

[3] Tracey, N., Clark, J. A., &Mander, K. (1998). The way forward for unifying dynamic test-case generation: The optimisation-based approach. In Proceedings of the IFIP International Workshop on Dependable Computing and Its Applications (DCIA).

[4] Perrouin, G., Sen, S., Klein, J., Baudry, B., & Le Traon, Y. (2010, April). Automated and scalable t-wise test case generation strategies for software product lines. In Software Testing, Verification and Validation (ICST), 2010 Third International Conference on (pp. 459-468). IEEE.

[5] Sant, J., Souter, A., & Greenwald, L. (2005, May). An exploration of statistical models for automated test case generation.In ACM SIGSOFT Software Engineering Notes (Vol. 30, No. 4, pp. 1-7).ACM.

[6] Hanna, S., & Munro, M. (2007, May). An approach for specification-based test case generation for Web services.In Computer Systems

and Applications, 2007.AICCSA'07. IEEE/ACS International Conference on (pp. 16-23). IEEE.

[7] Tai, K. C., & Lie, Y. (2002). A test generation strategy for pairwise testing.IEEE Transactions on Software Engineering, 28(1), 109-111.

[8] Rothermel, G., Untch, R. H., Chu, C., &Harrold, M. J. (1999). Test case prioritization: An empirical study. In Software Maintenance, 1999.(ICSM'99) Proceedings. IEEE International Conference on (pp. 179-188). IEEE.

[9] Czerwonka, J. (2006, October). Pairwise testing in the real world: Practical extensions to test-case scenarios.In Proceedings of 24th Pacific Northwest Software Quality Conference, Citeseer (pp. 419-430).

[10] Bertolino, A. (2007, May). Software testing research: Achievements, challenges, dreams. In 2007 Future of Software Engineering (pp. 85-103).IEEE Computer Society.

[11] Volume 5, Issue 1, January 2015 ISSN: 2277 128X International Journal of Advanced Research in Computer Science and Software Engineering Research Paper Available online at: www.ijarcsse.com Test Case Selection and Prioritization Using Multiple Criteria M. Suppriya, A. K. Ilavarasi, Department of Computer Science Sona College of Technology, Tamil Nadu, India