CS 569 Spring 2016
Qi Wang
wangq5@oregonstate.edu
ID: 932-439-151

Project Proposal

Introduction

Before I start to work on this project, I have read many relevant papers about test generation method. The test generation is the process of creating many sets of test cases that will be used to test the software applications. So how to make the test generation method accurate and efficient is a very important part for the software testing. In the last term, we use the random test method to test the tstl program. The random testing method is a black-box software testing technique, so the program uses the randomly generated inputs. It actually has many advantages. For example, in some complex systems, the random testing could produce some inputs that are would not think to try for users, For the most programs, this testing approach is useful, but there are still some programs are not always found bugs when using the random testing, because this method could not make sure every possible inputs would be tested.

Proposal

So in this project, my main idea to implement the adaptive random test generation is trying to use fewer test cases to detect the software faults in TSTL. According to the article "Adaptive Random Testing", the authors state that the adaptive random testing method is based on the intuition that can distribute test cases evenly for the software under test (SUT), and it can use fewer test cases to detect the fault or failure than the random testing method. The authors use an empirical study on adaptive random testing, and the study shows that the adaptive random testing can have smaller F-measure than random testing. The result also shows that the adaptive random testing has a good chance at outperforming random testing. So it may be efficient to implement the adaptive random testing generation method in TSTL.

In the article "Adaptive Random Testing: the ART of Test Case Diversity", the

authors state that the adaptive random testing method achieve close to the theoretical maximum test case effectiveness by any possible testing method using the same information. What is more, the recent research shows that the adaptive random testing method can be applied to large range of software.

Implement Plan

I will implement the adaptive random testing generation in TSTL. In the following weeks, first of all, I will need to read more relevant papers to ensure to totally understand the adaptive random testing algorithm. I will also think what kind of programs need to be tested, and also study the source code of random testing in the TSTL. Secondly, when getting the initial version of the program, I may use the avl.tstl and linklist.tstl programs to test to make sure if the tester works well. So I think this would be the first version of the project. Then I will implement the branch coverage and revise the code in the following weeks. I will also keep improve the program to make sure the final version of the program can be meet the number of test cases to detect the faults. At last, I may compare the ordinary random testing and the adaptive random testing in TSTL, and I expect that my program can be more efficient and use fewer test cases than the random testing.

References:

[1] T.Y. Chen, F.C. Kuo, H. Liu, and W.E. Wong. Code coverage of adaptive random testing. *IEEE Transactions on Reliability*, 62(1):226–237, March 2013. http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=6449335&url=http%3A%2F%2Fieeexplore.ieee.org%2Fiel5%2F24%2F6471782%2F06449335.pdf%3Farnumber%3D6449335

[2] T.Y. Chen, H. Leung, and I.K. Mak. Adaptive random testing. http://www.utdallas.edu/~ewong/SYSM-6310/03-Lecture/02-ART-paper-01.pdf

[2] T.Y. Chen, F. Kuo, R. Merkel, and T.H.Tse. Adaptive random testing: The art of test case diversity. *Journal of Systems and Software,* 83(1):60–66, 2010.

https://pdfs.semanticscholar.org/11c1/87d3cd8394f4d13523b97d0a40cbdced1691.pdf