Project Proposal

Zheng Zhou
`Student ID: 932463162`

April 19, 2016

1 Introduction

Random tester has already been proven as a not only simple but also effective approach to perform testing tasks for software. In common random testing, test cases usually are chosen randomly under uniform distribution. The performance of random testing can be quite poor due to duplicate and meaningless test cases, and also, it does not make use of any information to generate test cases. Thus random tester might not be a powerful testing method. Under some circumstances we want to perform better test harness on SUT to reduce overhead. Apparently random tester can not guarantee to find the failures with low overhead. Thus what I want to complete this term is to implement a test harness based on random tester but more likely to detect failures using fewer test cases then ordinary random tester. I will refer to the paper *Adaptive Random Testing*[2] to implement adaptive random testing. The idea behind adaptive random testing is to distribute test cases more evenly within the input space of SUT. Namely, adaptive random testing is a smarter algorithm that can choose relatively better test cases internally to reduce unnecessary testing input (e.g. semantically similar input that tests same branches or statements or same inputs with different order). Generally speaking, more evenly spread test cases have a better chance to hit the failure pattern with less test cases. Commonly in random testing studies, the rate of failure-causing inputs is used in the measurement of effectiveness. As the paper described, the effectiveness of adaptive random testing can be significantly improved without increasing overhead of test harness, which is huge improvement, and it is worth implementing with TSTL.
ART considered the distribution of test cases in input space, and try to generate test cases more evenly. There are a lot of different criteria to describe evenly. In this paper, authors mainly considered to use Euclidean distance between the elements in executed set and candidate set. we pick up next test case such that the minimum distance should be the largest minimum distance. Based on this criteria we are actually choosing a more suitable test case for next iteration. After taking the information of test cases into account, the tester will perform more reasonable and evidently better in experiment. Furthermore, according to the paper *Code Coverage of Adaptive Random Testing*[1], ART can achieve higher code coverage than random testing with the same number of test cases, which is also a interesting fact. This paper also showed ART delivers higher effectiveness and higher confidence in the reliability of the SUT even there is no failure detected. Hence it will be exciting if we can implement ART in TSTL. With a lot of flexible APIs, TSTL is definitely feasible to implement ART algorithm.

2 Project Plan

For implementing adaptive random testing using TSTL in next few weeks, I will first read the paper carefully to ensure I understand the exact algorithm of adaptive random testing. In the meantime I will also look up the source code of random tester that is already existed in current TSTL version, and to understand how random tester works in TSTL with SUT API. After I get some progress on implementation, I will use avlblock.tstl buggy version to test my tester to see if everything running in right direction. this will be the first rough version of ART for project milestone 1. If everything goes well, Branch coverage will be implemented during next phase, and algorithm will also be improved. Finally, the measurement of adaptive random testing will be based on the expected number of test cases required to detect the first failure as paper introduced (F-measure). I will do comparison for existed random tester and the adaptive random tester I implement. Expectedly, adaptive random tester should have lower F-measure. All of those measurement will be reported in the final report.

References

[1] H. L. Tsong Yueh Chen, Fei-Ching Kuo. Code coverage of adaptive random testing, 2013.

[2] I. M. T.Y. Chen, H. Leung. Adaptive random testing, 2004.