Adaptive Random Test Generation in TSTL

Thang Hoang

April 19, 2016

1 Project Description

It is well-known that random testing has been intensively adopted to detect potential failures in software due to its simplicity, efficiency and ease of implementation [5] compared with other complicated testing techniques [4]. Despite its surprisingly effective production of error-revealing test cases, random testing sometimes might generate a numerous of test cases which are unreachable, illegal or duplicated with others, and therefore limit its effectiveness. As observed by Chen et al [3], the probability of finding failures in the software does not only depend on its failure rate, but also depends on the geometric pattern of the inputs. From their empirical observation, a lot of programs bugs are found in contiguous areas of input domain which is denoted as *failure patterns*. Specifically, if test case t does not find any bugs in the software under test (SUT), then the other test cases having a distance "far" from t is more likely to find bugs easier than t's neighbors [3]. Therefore, the authors proposed Adaptive Random Testing (ART), a technique that can improve random testing by considering such geometric input patterns. In other words, ART enhances the diversity of the test cases by randomly selecting test cases as in RT but in such a way that they can spread over the input domain. The main intuition of ART can be descried as follows: First, the authors employ two sets of test cases including a non-failure executed set which is initialized as empty and a candidate set which is initialized by randomly selecting a test case from input domain. For each step, one test case from the candidate set that has the largest distance (e.g., Euclidean distance) with all test cases in the executed set will be selected for executing. This process is repeated until all allocated resources are spent (e.g., timeout expired)or a bug is found. Finally, the effectiveness of ART is evaluated by using F-measure (i.e., the expected number of test cases to find the first failure in SUT). This approach makes ART as simple as RT and yet achieves better result in finding bugs. Moreover, as has been shown in [1], ART achieves code coverage significantly higher than RT. Based on such properties, ART can be considered as an ideal alternative for RT to validate the software reliability.

Therefore, according to superior features of ART in terms of efficiency versus failure finding capability compared with other complicated techniques, I would like to investigate on Adaptive Random Testing algorithm in this project. This includes implementing ART in TSTL to improve the efficiency of random testing in finding bugs in Python programs. The most papers that I would rely my work on is from Chen et al [2, 1], who are the authors of ART. Moreover, I would like to investigation on how to adapt the coverage of ART as shown in [1] in TSTL somehow. Finally, I expect that by using ART over TSTL to evaluate the confidence of programs, it will take less time to detect bugs in Python program, cover more branches and therefore, cost less resource than other techniques.

1.1 Plan

I organize the plan for this project as follows:

- 04/20 – 04/27: Survey papers on Adaptive Random Testing. I would like to focus mostly on two papers from Chen et al. [1, 2].
- 04/28 – 05/03: Preliminarily implement ART in TSTL. This includes investigating APIs that TSTL supports and attempting to implement ART using such APIs. The objective is to have preliminary test generation results for milestone 1.
- 05/03 – 05/18: Improve the previous implementation to make it more efficient in terms of bug finding and resource usage to achieve milestone 2. This include optimizing some parameters and branch coverage implementation.
- 05/19 – 06/06: Finalize the project by trying to optimize the implementation regarding to some resource parameters.

References

[1] Tsong Yueh Chen, Fei-Ching Kuo, Huai Liu, and W Eric Wong. Code coverage of adaptive random testing. *Reliability, IEEE Transactions on*, 62(1):226–237, 2013.

[2] Tsong Yueh Chen, Fei-Ching Kuo, Robert G Merkel, and TH Tse. Adaptive random testing: The art of test case diversity. *Journal of Systems and Software*, 83(1):60–66, 2010.

[3] Tsong Yueh Chen, Hing Leung, and IK Mak. Adaptive random testing. In *Advances in Computer Science-ASIAN 2004. Higher-Level Decision Making*, pages 320–329. Springer, 2004.

[4] Alex Groce, Gerard Holzmann, and Rajeev Joshi. Randomized differential testing as a prelude to formal verification. In *Software Engineering, 2007. ICSE 2007. 29th International Conference on*, pages 621–631. IEEE, 2007.

[5] Richard Hamlet. Random testing. *Encyclopedia of software Engineering*, 1994.