Project Proposal

Yi-Chiao, Yang
932526065

In the class CS562 Applied Software engineering, we created a tstl file to test a Python library through TSTL (Template Scripting Testing Language). In the tstl file, we usually need to design utility functions to help TSTL to generate and test Python library because not all of Python libraries have enough information for use to test it directly.

TSTL is kind of model based testing that people write a *harness code* for SUT to test. The feature of harness code is that it executes repeatedly with random values, and we can define these random values in harness codes [1]. The random testing can be applied for these testing filed, including model based testing, structural testing, Object-Oriented program testing, performance testing, etc. In addition, model based testing is usually seen as black-box testing. The black-box testing can be used for each level of testers. Testers just need to know the inputs and what the expected outcomes should be, and they do not need to how the test program executes or the principle of the test program. Unlike black-box testing, white-box testing needs testers to know the internal structure of test program, but it can achieve more percent of coverage.

The purpose of our project in this class is to design a test generator based on TSTL API. In the previous paragraph, we know that TSTL is based on random testing. In our past project, we define a range of value for pools to use. I am not sure what kind of method of random generation TSTL use, but we define a sequence of values for pools to use. In my idea of test generation, I wonder what the outcome is after I define non-sequence of value for pools to use, and how can I improve the efficiency of TSTL. Fast Antirandom Test generation (FAT) [2] has some ideas that are related to my thoughts. FAT is based on the Hamming distance and Cartesian distance [3]. The concept of Hamming distance is between two binary strings of equal long and their number of digits in corresponding positions is different. For the FAT, we assume that the process generates some values after the first testing. In second testing, we apply the idea of FAT to get the total maximum distance form the previous test or all previous tests, and get the corresponding values of total maximum distance and repeat these procedures until we find a error or the resources of the process is gone.

For my future works, the original idea of Antrandom Testing is proposed by Yashwant K. Malaiya in 1995. It is an old paper, but many people research and develop this idea in recent years. I need to read more papers that are related to the idea of Antirandom testing. Also, I will attempt to integrate my algorithm for implementation.

[1] Alex Groce and Jervis Pinto. "A Little Language for Testing." In *NASA Formal Methods Symposium*, pages 204-218, Pasadena, California, April 2015
[2] Anneliese von Mayrhauser, Andre Bai, Tom Chen, Charles Anderson, Amjad Hajjar: Fast Antirandom (FAR) Test Generation. HASE 1998: 262-269
[3] Y.K.Malaiya,"AntirandomTesting:GettingtheMostoutof Black-Box Testing", *Procs. ISSRE '95*, Toulouse, Oct. 1995, p. 86-95.
http://www.cs.colostate.edu/~malaiya/p/antirandom95.pdf