Feedback directed Random Test Generation

Project Proposal

Sahar Alizadeh

19/4/016

Introduction

Random testing selects inputs at random from the programmer's input space and checks the program behaves correctly on each space. In this testing, program was tested by generating random test by independent inputs and the outputs are compered against to verify that the test output is pass or fail. Theoretical works indicated that random testing is as effective as more semantic input generation techniques but some empirical studies suggest that systematic testing is more effective than random. Besides, random Test is useful in wildly used and well-tested libraries and performs semantic and undirected test generation. Random testing is also used to find errors in many applications include Unix utilities, Windows GUI applications, Haskell programs and Java programs [1].

Feedback directed test generation could perform systematic testing in term of coverage and error detection. It builds sequences incrementally, starting from an empty set of sequence and it take four inputs, sequences, filters and a time limit for generation process stops [1].

Proposal Idea

I would like to work on random test generation by incorporating feedback obtained from executing test inputs. In this technique, inputs are randomly selected from a method, which apply and find arguments from last constructed inputs. The result of execution will

distinguish that whether input is repetitive, useless or they are useful. So, that the goal is avoiding redundant and illegal inputs. The inputs are built incrementally which, means new test input extend previous ones. As soon as a test input created, they are executed and it uses execution results to guide generation towards sequences that create new object states. During generation, maintain a set of all objects created and a sequence is redundant if all the objects created during its execution are members of the above set.

In methodology part, a test consists of sequences of method, which are set up state and an result part which is final call which is based on these parts: method sequence, extending sequence, feedback directed generation, filtering and repetition [2], [3].

References

1- *Feedback directed random test generation, Carlos Pacheco, Shuvendu K. Lahiri, Micheal D. Ernst and Thomas Ball*

2- Model cheching programs, W. Visser, K. Havelund, G. Brat, S. Park and F. Lerda

3- Test input generation for Java containers using state matching. W. Visser and R. Pelanek