CS562 Final Paper

Wei Liu

ID: 932305582

Email: liuw3@oregonstate.edu

Prof: Alex

Date: 3/12/2016

Section 1: Introduction part

This is my final report for CS562 about TSTL testing. I used TSTL to test the library, which was searched on Github website about data structure. The library link will be shown below. This library is very huge so I chose single link list to be my test target. As, we know, single link list is the basic data structure and it is easy to figure out bugs or mistakes. In this python code, there are fourteen functions inside. It includes three basic functions, which are insert, delete and generate. For the insert function, it will add a new value on the head of the list. For test, I will compare the next value of head with the pre-head. If the next value of head is same as the pre-head and that means the new value has been inserted successfully. For the delete function, I will compare the next value of pre-head with new head, if they are same, that means the old head has been removed successfully. For the generate part, I will create a new list and compile insert function. If the head is the value that been inserted, that means the generate and insert functions are correct.

Section 2: Bugs report and fix (Did you find bugs?)

For my library, I have tested most of it but still can't find any bugs. This library that I chose was tested by the author and the test code is also shown on the website. I think the author has already fixed all bugs and made his python code excellent. That's why I can't find any bugs.

Section 3: Accomplish part (What did you accomplish?)

In my library, there are fourteen functions. Until now, I finished 11 functions, which are

Insert: This function is used to insert an element into the list, the position should be on the head. It is different from the append function.

Append: This function is used to append an element into the end of the list.

ReturnIndex: Return the first element that indexed that the process firstly found.

updataIndex: After insert a element, this function is used to update it with the position of index.

deleteIndex: similar as updateindex, this function is used to delete the element with the position of index.

insertBeforeIndex: Insert an element on the position, which is before the index.

InsertAfterIndex: Insert an element on the position, which is after the index.

Deletedata: This function is used to delete the data that was firstly meet with.

deleteAlldata: Obviously, this function will delete all data and makes the list empty.

Deletelist: This function is used to delete this list.

findMthTOlastNode: This function is used to search a specific data.

For test these functions easier, I also defined some structures to help me finish this test. I defined six of these and used them in actions part. These structures are length, Append, P_value, D_Index, D_datadele, Dele_V.

In the actions part, which is the testing part, I will introduce some of them and codes will be shown below:

Example 1:For test insert function:

As we known, after insert a value in the list, the length should plus 1.

(length(<LIST,1>.head) == PRE<(length(<LIST,1>.head))>+1)

I compared the current length with the pre length and if it is correct, that means the value has been inserted successfully.

Example 2: For test insertBeforeIndex:

I compared the date, which is collected at index position and compare it with the value that was inserted. If they are same, that means this function is correct. Also, the length of list should plus 1.

The function insert after index is similar as this function.

Section 4: How well does the tester work

I think the tester did an excellent job in his single linked list python code because I can't find any bugs inside and the author also tested by himself/herself by other language that I don't understand. Anyway, this library is a really perfect job that I have never seen before. For me, I chose this python code and try to understand every function inside to help me test it. This is important for next step. Define test syntax and parameters are the second part of my job. In the tstl file, I separated it into three parts, which are defining parameters part, pools and testing actions.

Section 5: What's wrong with TSTL?

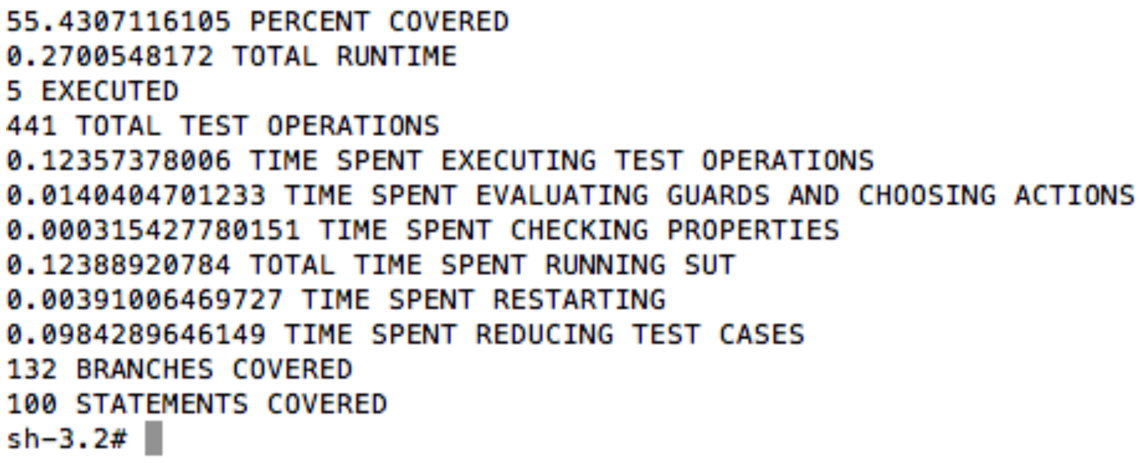
From this course begin, I first time know this language and try to use it and find something wrong or weakness. Until now, I feel that RandonTester is the only part that makes me uncomfortable, because it executes the SUT file randomly, that means I can't control the input string and that will cause a incorrect form.

Section 6: What is good about TSTL?

Before I know TSTL, I usually feel confuse and hardness to find bugs in python code. For a new learner, I may make many mistakes or incomplete thoughts. With the help of TSTL, I think I can improve my python written and design and help me find my bugs really easy. During I use TSTL, I found that tstl can produce huge of random date, these data will be send to test functions. This part makes the user comfortable and easy. More than this, TSTL has good perform on testing boundaries with the help of pointers.

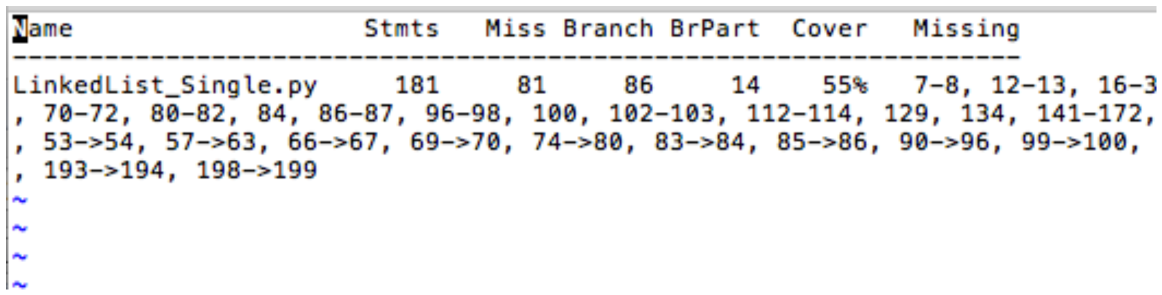
Section 7: Coverage summaries, any other important data on the test effort

As the image shown at the last, my total test coverage is about 55% of my library. Some of functions I can't test because I don't understand those python code. I tried to test every function but failed. I think the most import thing is time, if I have enough time later, I will try to test all of them.



```
55.4307116105 PERCENT COVERED
0.2700548172 TOTAL RUNTIME
5 EXECUTED
441 TOTAL TEST OPERATIONS
0.12357378006 TIME SPENT EXECUTING TEST OPERATIONS
0.0140404701233 TIME SPENT EVALUATING GUARDS AND CHOOSING ACTIONS
0.000315427780151 TIME SPENT CHECKING PROPERTIES
0.12388920784 TOTAL TIME SPENT RUNNING SUT
0.00391006469727 TIME SPENT RESTARTING
0.0984289646149 TIME SPENT REDUCING TEST CASES
132 BRANCHES COVERED
100 STATEMENTS COVERED
sh-3.2#
```

Also, here is the coverage.out file.



```
Name                       Stmts   Miss Branch BrPart   Cover   Missing
-------------------------------------------------------------------
LinkedList_Single.py      181     81     86     14     55%   7-8, 12-13, 16-3
, 70-72, 80-82, 84, 86-87, 96-98, 100, 102-103, 112-114, 129, 134, 141-172,
, 53->54, 57->63, 66->67, 69->70, 74->80, 83->84, 85->86, 90->96, 99->100,
, 193->194, 198->199
~
~
~
~
```

This photo shows the coverage and the missing part. Some of the missing parts are the initial part or claim part, which are unable to be test. For my library, I think it is impossible to test one hundred percent.