Part3: Bug and progress report

Student Name: Yipeng Wang

Date: Mar/01/2016

For finding bugs of current library testing, I checked the insertion of new nodes in treap data structure. After insertion, I choose to check the priority of the leave nodes and the priority of the parent nodes to find whether it will return the expect result. Moreover, I checked whether the value of the left node is less than the value of the right node. In addition, I checked the merge function, which I got the expect result. Fortunately, until now, there are no any bugs I could found at this part.

For the progress to date, since I choose the treap library as the system which under my test. Treap is one kind of data structure, which has the attribute of tree and heap, that is why it is called “treap”. However, it also has somethings unique. In treap, each node will not only contain the value which seems like the “tree” data structure, but also contains the priority which is one of the unique attribute of “heap” data structure. So, to debug this data structure, we need to consider much more than simply test the tree or heap. As I said above, I constructed a new treap by using the TSTL, then I add nodes which has random value and random priority as the root of the treap. After that, I will keep insert new nodes in treap which follow the rules from the original python files. After I construct a complete treap, I started to check whether the priority of the leave nodes is less than the priority of his parent node. In addition, I check the value of the left.node is whether larger than the value of the right.node. For the merge part, I merged

new treap in TSTl to test it whether the priority of the left.treap is less than the priority of the right.treap, since each treap has no problem which I have already tested. Until now, after testing for many times, I got the fortunate result is that this program does not have any bugs at this part.

For the future works, I will continue to test the treap data structure. I have already planed for how to test my program. At first, I will test the delete function in treap. I would like to generate a treap with several nodes which has different values and priority. Then I will try to delete one of the node which has certain priority or value. Since if the value of the node which will be deleted is not leave, we need to rotate the treap to keep it balance. So, after delete, we could test whether the value of the certain node is larger than his right brother node. In addition, I could test the priority of the certain node is less than his partent node or even the root. If I have more time, I will test the rotation function step by step, which means, I will return the current value and priority after I try to delete one of the node. By using the result, I will try to compare his left or right brother or his parent to check whether it rotates as the rules or sometimes it will cause some wrong rotation.

For the system under the test, as far as I am concerned, TSTL could easily test this kind of data structure, like: tree, heap. Queue, or something like this, because, TSTL could generate several random values, by using this, we could easily test these data structure directly.

For the code coverage, I am not very familiar with that and I will keep learning from that.