Problem 62

Ryan Burmeister

March 6, 2016

1 Fastest Search

The fastest search time would be between the rBST and Treap as a Skip List would require the traversal over links with variable points in memory. The Treap would seem to have the fastest search time as it seems to be less biased towards the order of its inputs. A rBST would seem to have a greater chance to look closer like a degenerate tree if all of its inputs were in ascending or descening order. For that reason, a Treap would have the fastest searh time.

2 Fastest Insertion

Treaps would have the fastest insertion time as it simply requires the addition of an item to an array with some bubbling up as necessary. A Skip List requires the maniuplation of pointers and the creation of a node with a Tower of a particular height based on the number of flips. rBSTs require the restructuing of the tree based on where the new node is inserted.

3 Fastest Deletion

Treaps would have the fastest deletions. Skip Lists would require the changing of a number of pointers equivalent to the height. A rBST would again require the restructuring of the subtree in order to determine which nodes should be the roots of each subtree. Treaps would simply require moving elements in right diagonal of the subtree up the tree.

4 Simplest to Implement

A Skip List would be the easiest to implement as all that it would require is the creation of Towers (arrays of pointers) and the changing of pointers for insertion and deletion. Each of the tree methods require the user to correctly swap elements in an array with the insertion or deletion of nodes.

5 Smallest Memory Footprint

rBSTs would have the smallest footprint of the three. Both rBSTs and Treaps only store their data and a couple of pointers while Skip Lists store arrays based on the height of each tower. Furthermore, rBSTs just have to store the data while Treaps also store a value for the priority of each node.