PROBLEM SET 6
Due: Thursday, April 12

Reading: Red-Black Tree notes (Handout #22) Chapter 14; Sections 15.1 & 15.2; Chapter 19


Problem 1 [25]

a. Draw the sequence of 2-3 trees $T_0, T_1, ..., T_{12}$ that results from inserting the following letters one-by-one (from left to right) into an empty tree:

THE QUICKYAMS

b. Draw the sequence of 2-3 trees $T_{12}, T_{13}, ..., T_{24}$ that results from deleting the following letters one-by-one (from left to right) from the tree from part a.

THE QUICKYAMS

Problem 2 [25]

a. Draw the sequence of red-black trees $T_0, T_1, ..., T_{12}$ that results from inserting the following letters one-by-one (from left to right) into an empty tree:

THE QUICKYAMS

For eliminating red-red violations, use the simple algorithm presented in Handout #22, not the CLR version.

b. Draw the sequence of red-black trees $T_{12}, T_{13}, ..., T_{24}$ that results from deleting the following letters one-by-one (from left to right) from the tree from part a.

THE QUICKYAMS

Problem 3 [30] CLR Problem 14-2 (p. 278)

Problem 4 [20]

a. Describe how to augment each node of a red-black tree with extra fields so that calculating the predecessor and successor of any given node can be performed in constant time. (See Section 15.1 -- 15.2 on augmenting red-black trees.) You show that the new fields can be maintained efficiently during Insert and Delete. This requires showing two things: (1) maintaining the field after BST insertion or deletion takes time $O(\log(n))$ and (2) the fields can be updated in $O(1)$ for each rotation during the fixup phase.

b. Describe how to augment each node of a red-black tree with extra fields so that calculating the minimum and maximum of any given node can be performed in constant time.
Name:

Date & Time Submitted *(only if late)*:

Collaborators *(anyone you collaborated with in the process of doing the problem set)*:

*In the Time column, please estimate the time you spent on the parts of this problem set. Please try to be as accurate as possible; this information will help me to design future problem sets. I will fill out the Score column when grading your problem set.*

<table>
<thead>
<tr>
<th>Part</th>
<th>Time</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>General Reading</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Problem 1 [25]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Problem 2 [25]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Problem 3 [30]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Problem 4 [20]</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>