CS 561, HW3

Prof. Jared Saia, University of New Mexico

Due: October 21st

1. A certain computer virus spreads as follows. In a given minute, the number of machines infected at the last minute will each infect one new machine. However, a number of machines are disinfected that is equal to the number of machines in an infected state 2 minutes ago. In this problem, you will first write and solve a recurrence relation for the number of machines infected after n minutes. Let $f(n)$ be the number of machines that are infected at minute n. Write a recurrence relation for this value. Now find the general form solution to this recurrence using annihilators. Now assume $f(0) = 1$ and $f(1) = 2$. What is the exact solution for $f(n)$?

   Now consider the following second computer virus. In a given minute, only *half* of the number of machines infected at the last minute each infect one new machine. However, no machines are ever disinfected. What is the general form of the solution for this second virus? Which virus spreads more quickly, the first or the second?

2. Problem 4-5 (VLSI chip testing) - This is a really good divide and conquer problem I left out of the last hw

3. Exercise 6.4-2: Argue the correctness of heapsort

4. Exercise 12.2-4 (Prof. Bunyan's property)

5. Problem 12-3 (Average Node Depth in Randomly Built Binary Search Tree)

6. Exercise 13.3-1 ("In line 16 of RB-Insert ...")

7. Problem 13-3 (AVL Trees)

8. Problem 13-4 (Treaps)

9. (h-trees) A h-tree is a rooted binary tree that is a useful data structure for designing self-healing networks. Let ℓ be a positive integer. For ℓ a power of 2, the complete tree with ℓ leaf nodes is the unique h-tree with ℓ leaf nodes. For ℓ not a power of 2, a tree with ℓ leaf nodes is a h-tree if and only if (1) the root node, r, has two children; (2) the left subtree of r is the root of a complete binary containing $2^{\lfloor \log \ell \rfloor}$ leaf nodes; and (3) the right subtree of r is a h-tree. (Recall that a *complete* binary tree is one where every internal node has two children and every leaf node has the same depth)

   Show the following by induction:

   - For all positive ℓ, there is a unique h-tree with ℓ leaf nodes.
   - Call the h-tree with ℓ leaf nodes $h - tree(\ell)$. Then, the height of $h - tree(\ell)$ is $\lceil \log \ell \rceil$

10. In the self-healing application of h-trees, the leaf nodes are associated with actual machines in a network, and the internal nodes represent additional "router nodes" (a scarce resource). To merge a list of h-trees, $h_1, h_2, \ldots, h_x$ we want to create a single new h-tree, h, which contains as leaf nodes all the leaf nodes in $h_1, h_2, \ldots, h_x$, and adds the smallest number of new internal nodes as possible.

    - Show how you can quickly merge a collection of x h-trees, each of size no more than n, into a single big h-tree by adding no more than $O(x \log n)$ additional internal nodes. What is the runtime of your algorithm?

    Hint: Think about how to set up a correspondence between binary numbers and h-trees, and binary addition and h-tree merging.