CS 182
Advanced Topics in Algorithms
Spring 2003
Problem Set 3b
Due Thursday, February 11

In addition to Problem Set 3a (which was deferred last time), please do the following two problems:

1. **[25 Points] Analyzing Insertion into Splay Trees.** In class we analyzed the amortized cost of performing a FIND in a splay tree. We showed that while the actual cost of the FIND is d (where d is the depth of the last node visited during the FIND), the change in potential due to the subsequent splaying step was less than or equal to $(3\log_2 n) - d + 2$, giving us an amortized cost of $(3\log_2 n) + 2 \in O(\log_2 n)$.[1]

   Now let's find the amortized cost of performing an INSERT in a splay tree. The actual cost of the insertion and subsequent splaying is d, the depth of the leaf at which the insertion is performed. However, the amortized cost will be less because of the potential change. The potential changes in two ways. First the potential changes because a node is inserted at the bottom of the tree, increasing the rank, $r(v)$, of each node on the path from the root to this leaf. Next, the potential changes due to the splay. We already know that the splay operation contributes a change of potential which is less than or equal to $(3\log_2 n) - d + 2$. So, we only need to determine the change in potential due to the increase in ranks along the insertion path.

   Let $n(v)$ and $r(v)$ denote the number of nodes in the tree rooted at v and the rank of v, respectively, before the insertion. Let $n'(v)$ and $r'(v)$ denote these values immediately after the insertion (but before the splay). Show that the change in the tree's potential immediately after the insertion but before the splay is $O(\log_2 n)$. (Note: To do this, write down the simple relationships that you can find between the various n, r, n' and r' variables. Then formulate a summation for the change in potential. Finally, find an upper-bound on this summation.) Now put everything together to conclude that the amortized cost of an INSERT in a splay tree is $O(\log_2 n)$.

2. **[20 Points] ACN Programming Competition!** You are competing in the International ACN (Association of Computer Nerds) Programming Competition. In one of the programming problems, you need to implement a queue. Unfortunately, you are pressed for time. Fortunately, you already have good working code for a stack. You may assume that the stack is implemented efficiently and supports operations PUSH, POP, and SIZE (which returns the number of items on the stack) in time $O(1)$. Show how 2 stacks can be instantiated to implement a queue. Moreover, show that the

[1] You might argue that the actual cost of a FIND is $2d$ since we first pay d to go down the tree and then pay d again to do the splay step. In reality, the cost is some constant K times d and we just use 1 for this constant to keep life simple. If we want to keep the constant K around (because living simply is boring), we can just scale the potential function up by that same constant so that the Kd real cost is now offset by a Kd decrease in the potential. Of course, now the $3\log_2 n$ term now becomes $3K\log_2 n$.

amortized running time for ENQUEUE and DEQUEUE are $O(1)$ using this 2-stack implementation.