CS 466/666 Fall 2018 due Tuesday October 16, 5 PM
A. Lubiw

ASSIGNMENT 3

DO NOT COPY. ACKNOWLEDGE YOUR SOURCES.

1. [10 marks] Recall the Union-Find data structure presented in class that uses tree linking and path compression to achieve $O(1)$ worst case time for a Union operation and $O(\alpha(m, n))$ amortized time for a Find operation, where n is the number of elements and m is the number of operations.

   Describe a data structure to perform the following operations on an array $A[1..n]$ where initially $A[i] = 0$ for all i.

   A. Given an index $i < n$ set $A[i]$ to 1.

   B. Given an index i, return $A[i]$.

   C. Given an index i, return the smallest index $j \geq i$ such that $A[j] = 0$. Note that such an index always exists since $A[n]$ is always 0.

   Operations A and B should run in $O(1)$ worst-case time, and operation C should run in amortized time $O(\alpha(m, n))$, where m is the number of operations.

2. [15 marks] This problem is about using randomness to maintain balance in a binary search tree, thus yielding a dictionary data structure with expected cost $O(\log n)$ for search, insert and delete. As usual, each element in the dictionary has a key attached to it. In addition, we will associate a *priority* with each element. The priority of an element will be chosen at random when the element is first inserted into the dictionary. Assume that the priorities are independent random variables. Assume for ease of analysis that keys are distinct and priorities are distinct. The search tree will be in binary search order with respect to the keys and *simultaneously* in heap order with respect to the priorities (minimum priority at the root).

   (a) [5 marks] Prove that given the keys and priorities there is a unique tree with these ordering requirements.

   (b) [5 marks] Prove that such a tree has expected height $O(\log n)$.

   (c) [5 marks] Describe how to insert in time $O(h)$ where h is the height of the tree. You may assume that rotation routines are given, i.e. you do not need to give details about implementing rotations. (Delete can also be done in $O(h)$ time, but you do not need to prove that for this assignment.)