CS2040S: Data Structures and Algorithms

Tutorial Problems for Week 7 Monday: Heaps and Priority Queues

For: 26 September 2022, Tutorial 5

Problem 1. True or False?

For each of the following, determine if the statement is True or False, justifying your answer with appropriate explanation.

a) The smallest element in a min heap is always the root. true

b) The second largest element in a max heap with more than two elements (all elements are unique) is always one of the children of the root. true

false. only need o(1) time cause
the parent is at floor (i/2) index using 1 indexed arr

c) When a heap is stored in an array, finding the parent of a node takes at least $O(\log n)$ time.

d) Every node in the heap except the leaves has exactly 2 children. false

e) We can obtain a sorted sequence of the heap elements in $O(n)$ time.

false- heap sort doesnt sort the array + it takrs O(n logn) time

Problem 2. Greater than x

theres 3 kinds of traversal for this do pre-order traversal(check root, left then right)

Give an algorithm to find all vertices bigger than some value x in a **max heap** that runs in $O(k)$ time where k is the number of vertices in the output.
(This is different from most algorithms you have encountered which are dependent on the size of the input, instead of the size of the output. We sometimes call this output sensitive algorithms).

for each node check, if its bigger than x. if yes output the node and continue traversing
but if its < x just terminate traversal on the subtree and go to parent and continue

Problem 3. Updating a heap

Give an algorithm for the `update(int old_key, int new_key)` operation, which updates the value of `old_key` in a binary heap (max or min) with `new_key` in the binary heap in $O(\log n)$ time, which does not change the time complexity of the other operations. You are required to modify the other operations in the binary heap if needed, including any additional data structures used. You may assume all values in the heap will be unique. (Additional: What if the keys are not unique?)

if get index of element in binary heap, can do shift down/shift up easily

Problem 4. Sorted? Almost. (Adapted from CS2040 AY19/20 Sem1 Final Exam)

A array A_k of n unique floating point values of no fixed precision is partially sorted when each value differs from its correct position in the sorted array by no more than k positions, where k is a positive integer. For example, an array A_2 could be $[1, 4, 3, 2, 6, 5, 7]$, where the sorted output should be $[1, 2, 3, 4, 5, 6, 7]$.

Text

Problem 4.a. Give an algorithm to sort a partially sorted array in $O(n)$ time, where $k = 1$.

Problem 4.b. Give an algorithm to sort a partially sorted array in $O(n \log k)$ time, where k can be any positive integer smaller than n.

Problem 5. Which number to pick?

The Great Overlord of Arithmetic has a challenge for you! He gives you a stack of n integers, and allows you to repeatedly perform the following operation:

- From any of the top k integers in the stack, you may remove one of them and add it to the Fundamental Pool of Arithmetic.

The Great Overlord of Arithmetic wants you to calculate the maximum total value of numbers that can possibly be placed in the Fundamental Pool of Arithmetic (starting from 0). Give an efficient algorithm to do this, and state the time complexity.

For example, for $k = 2$, where integers in the stack are $[2, -10, 2, -6, 5]$, the output of your algorithm should be 4. Another example for $k = 5$, where integers in the stack are $[-1, -1, -1, -1, -1, 10]$, the output of your algorithm should be 9.