CS 240E: Data Structures and Data Management **Winter 2022**

Midterm Practice Problems

Reminder: Midterm is on Thursday, March 3, from 4:30-6:20 pm.

Note: This is a sample of problems designed to help prepare for the midterm exam. These problems do *not* encompass the entire coverage of the exam, and should not be used as a reference for its content.

1 Runtime Analysis

1.1 Expected run-time analysis

Consider the following pseudocode. What is its expected run-time?

```
int randomizedMystery(int n) {
   if (n<=1) return 0
   else {
      i = random(n)
      randomizedMystery(i)
   }
}
```

1.2 Amortized analysis

Consider the following method to simulate a queue with two stacks S_1, S_2.

- initialize: initialize both S_1 and S_2 as empty stacks
- enqueue(x): do $S_1.push(x)$
- dequeue(): If S_2 is empty, do $S_2.push(S_1.pop())$ until S_1 is empty. Then return $S_2.pop()$.

Using a suitable potential function, argue that it is a potential function and show that all operations take constant amortized time.

2 Sorting algorithms

2.1 Sorting an array

Given an array $A[0...n-1]$ of numbers, show that if $A[i] \geq A[j]$ for all j with $j \leq i - \log n$, the array can be sorted in $o(n \log n)$ time.

2.2 Multi-Way Merge

Given a set of k sorted arrays, where the combination of the k arrays has n elements in total, design a worst case $O(n \log k)$ time algorithm that produces a single sorted array containing all n elements.

3 Skip Lists

Suppose you have a skip list with only three levels. The top level contains only the sentinels. The lowest level has $n + 2$ keys: $-\infty, a_0, \cdots, a_{n-1}, \infty$, while the middle level contains $k + 2$ keys including the sentinels. Assume k divides n. Suppose that the k entries are evenly spread out and the first entry corresponds to a_0.

1. What is the worst case time for a query? Give a tight bound involving k and n.
2. Given n, how should you choose k to minimise the worst case, and what does the worst case become in that case?

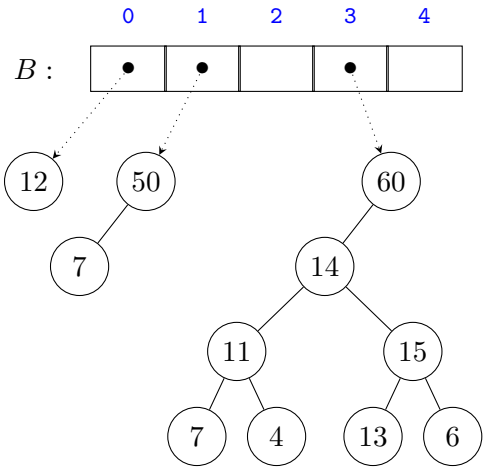
4 Lower Bound

Suppose you own n electrical devices. Each of them comes with a charger cable, which you tossed into a box when you got it. But now it is time to recharge the devices, and so you must find for each one the correct charger cable. For each device, exactly one charging cable is correct. The charging cables look similar enough that you cannot compare them amongst themselves. The only thing that you can do is plug a cable into a device, which will tell you whether the plug fits, or is too big, or is too small.
Argue that any algorithm to find cables for all devices must use $\Omega(n \log n)$ such operations in the worst case.

5 Misc. Computation

5.1 Binomial heap.

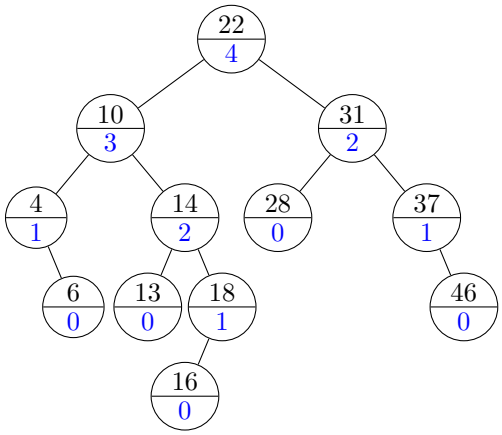
Consider the following binomial heap.



1. Show the resulting binomial heap when performing *insert*(40).
2. Show the resulting binomial heap when performing *deleteMax*.

5.2 AVL-trees

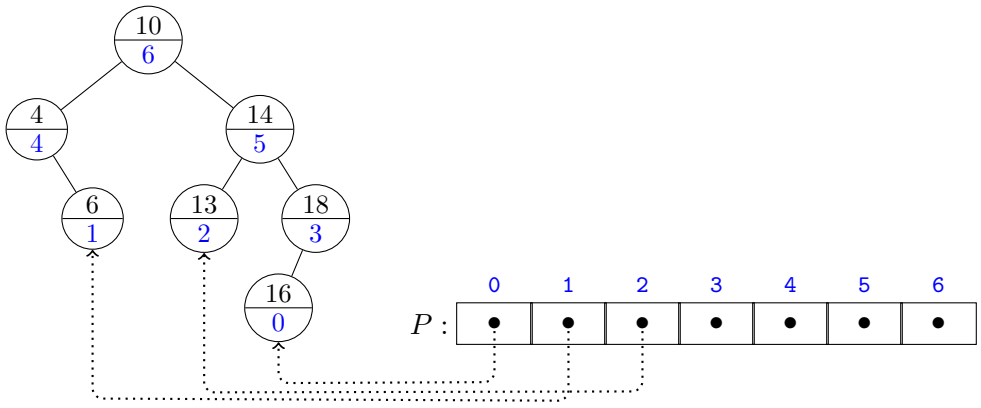
Consider the following AVL-tree.



1. Could there be an AVL-tree that (like this one) has height 4, but fewer nodes? Why not?
2. Show the result after performing *insert*(17).
3. Show the result after performing *delete*(4).
4. Pick a node z and a child y of z such that performing a single rotation at z to bring y up yields again an AVL-tree.
5. Pick a node z and a child y and grandchild x of z such that performing a double rotation at z to bring x up yields again an AVL-tree.

5.3 Treaps

Consider the following treap.



Show the result of doing *insert*(12) when the randomly chosen priority is 4.

6 Compressed Tries

Let T be a compressed trie of n strings from an alphabet Σ, which we assume does not contain the end-of-character symbol.

1. What is the maximum height of T?
2. What is the minimum height of T?

Give exact expressions (not order notation) in terms of n and $|\Sigma|$. If the answer depends on which of n and $|\Sigma|$ is greater, give answers for each scenario.

7 Hashing

Show the hash-tables that result if you insert the keys

$$103, 322, 841, 886, 296, 336, 571$$

(in this order) into an initially empty hash-table of size 10, using $h_0(k) = k \bmod 10$.

1. Use hashing with chaining, adding new items at the end of the bucket.
2. Use hashing with linear probing.
3. Use double hashing, with the second digit of key k as $h_1(k)$.
4. Use cuckoo hashing, with the first digit of key k as $h_1(k)$.