Homework 3

CS6033 Design and Analysis of Algorithms I
Fall 2024
(Sec. B, Prof. Yi-Jen Chiang)

Due: Wed. 10/9 by 1pm
(submit online on NYU Brightspace; one submission per group)
Maximum Score: 105 points

Note: This assignment has 3 pages.

1. (20 points)

(a) Textbook Exercise 11.2-2 (page 281). Just write down your final result. **(5 points)**

(b) Consider inserting the keys 10, 22, 31, 4, 15, 28, 17, 88, 59 into a hash table of length $m = 11$ using open addressing with the auxiliary hash function $h'(k) = k$. Illustrate the result of inserting these keys using linear probing, using quadratic probing with $c_1 = 1$ and $c_2 = 3$, and using double hashing with $h_1(k) = k$ and $h_2(k) = 1 + (k \bmod (m - 1))$.
For each method, just write down your final result. **(15 points — 5 points for each method)**

2. (13 points)

Consider the AVL-tree T as shown in Fig. 1 below, where the numbers are the keys stored.

(a) Copy the tree in your write-up, and for each node v label the height of the subtree rooted at v. (We define the height of a null node to be 0 and the height of a leaf to be 1). Verify that this is indeed a valid AVL-tree. **(3 points)**

(b) Suppose now we insert a key 7 to the tree T, and then insert another key 26 to T. For each insertion, show the resulting tree right after the insertion (i.e., before any re-balancing), and the tree after each structural change during re-balancing, as well as the final tree. **($3 + 3 = 6$ points)**

(c) Ignoring the insertions in **part (b)**, suppose we delete the key 15 from the tree T as shown in Fig. 1. We use the policy that when an internal node with no null child is deleted, we replace the deleted key with its **successor** key in the tree whenever the successor key is available. Show the tree right after the deletion (i.e., before any re-balancing), and the tree after each structural change during re-balancing, as well as the final tree. **(4 points)**

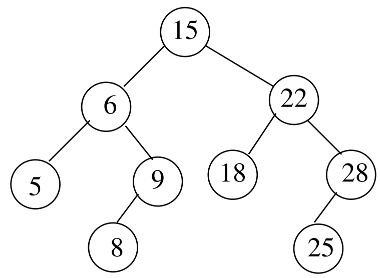

Figure 1: The AVL-tree for Question 2.

3. (20 points)
Show the results of inserting the keys

8, 21, 19, 13, 5, 14, 10, 22, 24, 25, 15, 20, 16, 18, 3, 4, 26, 27, 6

into an initially empty (2,4)-tree using the 2-pass insertion algorithm (as discussed in class and in the hand-out slides of (2,4)-trees), assuming that when a split operation occurs, the node in question is always split into a **left node** with **2 keys** and a **right node** with **1 key**. Draw only the configurations of the tree **right after some node has been split** (if inserting a key causes more than one split operation, draw the tree configuration right after **each** such split), and also draw the final configuration.
(Note: There are 8 such split operations in the sequence; scores of them are: 1, 2, 2, and 3 points for each of the remaining 5 splits.) $(\mathbf{1+2+2+3*5=20}$ **points**)

4. (16 points)
Show the results of deleting keys 20, 17, 10, 15 in that order, from the (2,4)-tree shown in Fig. 2 below (where the numbers shown are the keys stored), using the **2-pass deletion algorithm** (as discussed in class and in the hand-out slides of (2,4)-trees). Note that when deleting a key k from an internal node, we use the policy of replacing k with its **predecessor** key in the tree whenever the predecessor key is available. Also, for both the *transfer* and the *fusion* operations, if the immediate left and right siblings are both available for the operation, we use the policy of always using the **left** sibling. For each deletion, show the tree after each structural change and the final tree, and state the name of the re-structuring operation (transfer or fusion) if such operation(s) occur.

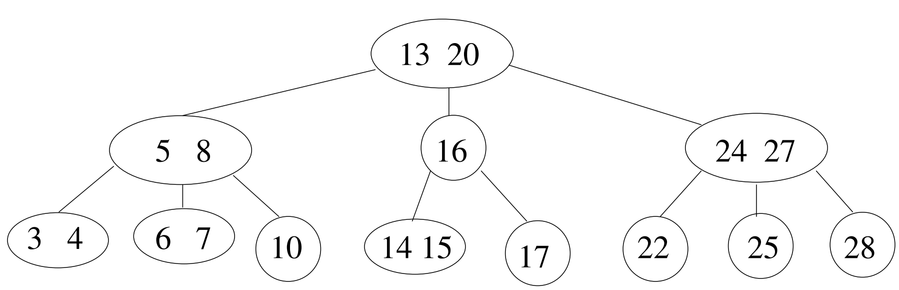

Figure 2: The (2,4)-tree for Question 4.

5. (24 points) Textbook Problem 11-3 (page 308, **slot-size bound for chaining**).

Do every part but **skip part (d)**. However, assume the results of **part (d)** to do **part (e)**, i.e., assume that some constant $c > 1$ exists such that $Q_{k_0} < 1/n^3$ for $k_0 = c \lg n / \lg \lg n$ and that $P_k < 1/n^2$ for all $k \geq k_0$ (and of course $k \leq n$).

Hints:

1. You may find the following **Boole's inequality** (also called the **union bound**, as discussed and proved in class) useful: Let A_i be an event for $i = 1, 2, \cdots, n$. Then $\Pr\{A_1 \cup A_2 \cup \cdots \cup A_n\} \leq \sum_{i=1}^{n} \Pr\{A_i\}$.

2. For **part (c)**, first show that $Q_k < \frac{1}{k!}$ (by arguing that $(1 - 1/n)^{n-k} < 1$ and so on), and then apply Stirling's approximation on $k!$. You do **not** need to prove Stirling's approximation.

3. For **part (e)**, observe that

$$E[M] = \sum_{x=0}^{n} x \Pr\{M = x\} = \sum_{x \leq t} x \Pr\{M = x\} + \sum_{x > t}^{n} x \Pr\{M = x\},$$

where $\sum_{x \leq t} x \Pr\{M = x\}$ is related to $\Pr\{M \leq t\}$ and $\sum_{x>t}^{n} x \Pr\{M = x\}$ is related to $\Pr\{M > t\}$, for any $t \in (0, n)$. Compare with the $E[M]$ formula to be proved and choose a suitable value for t. Finally, use this $E[M]$ formula and the results of **part (d)** as stated above to derive the $O()$ bound for $E[M]$.

(**Notes: 4 points for (a), 4 points for (b), 6 points for (c), and 10 points for (e).**)

6. (12 points)

Given n distinct, unsorted integers where their value range is $>> n$, design and analyze an algorithm to report all pairs of integers (a, b) among them such that $|a - b| = 1$. Your algorithm should run in $O(n)$ **expected** time and $O(n)$ worst-case space.