Chennai Mathematical Institute

Design and Analysis of Algorithms CU2101

Mid-Term

Aug-Nov 2025, Time: 2:30 HRs Max mark: 50

Attempt all questions.

1. Prove or disprove: In a min-heap, the next larger element of any given element can be found in $O(\log n)$ time. (4)

2. Write the step-by-step execution (after each major steps) of the heap sort algorithm on the following input. (4)

$$[3, 5, 1, 9, 6, 8]$$

3. Give asymptotic upper and lower bounds for the following recurrence. Assume that $T(n)$ is constant for $n \leq 2$. Make your bounds as tight as possible, and justify your answers. (4)

$$T(n) = T(\sqrt{n}) + 5.$$

4. Describe an algorithm to determine in $O(n)$ time whether an arbitrary array $A[1 \ldots n]$ contains more than $n/4$ copies of any value. (4)

5. Suppose, we use the double hash function $h(k, i) = (h_1(k) + ih_2(k)) \bmod m$ to resolve collision via open addressing in a hash table T. Let $d \geq 1$ be the greatest common divisor of m and $h_2(k)$. Then an unsuccessful search for key k examines _____ many slots of the table T. Justify your answer. (4)

6. An *inversion* in an array $A[1 \ldots n]$ is a pair of indices (i, j) such that $i < j$ and $A[i] > A[j]$. The number of inversions in an n-element array is between 0 (if the array is sorted) and $\binom{n}{2}$ (if the array is sorted backward).

   Describe and analyze an algorithm to count the number of inversions in an n-element array in $O(n \log n)$ time. (5)

7. Suppose we have used Open addressing to insert n keys into a hash table of size m. Prove that under the simple uniform hashing assumption the cost of next insert operation is $\frac{1}{1-\alpha}$, where $\alpha = \frac{n}{m}$. (5)

8. Suppose we have n points scattered inside a two-dimensional box. A kd-tree recursively subdivides the points as follows. If the box contains no points in its interior, we are done. Otherwise, we split the box into two smaller boxes with a vertical line, through a median point inside the box (not on its boundary), partitioning the points as evenly as possible. Then we recursively build a kd-tree for the points in each of the two smaller boxes, after rotating them 90°. Thus, we alternate between splitting vertically and splitting horizontally at each level of recursion. The final empty boxes are called *cells*. (20)

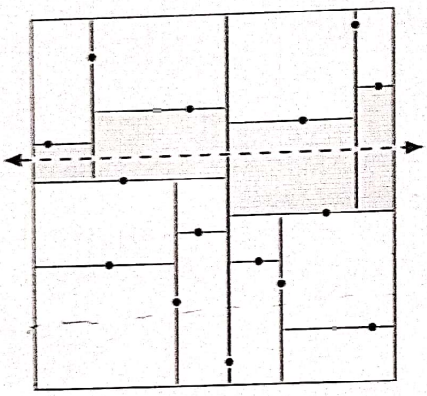

Figure 1: A kd-tree for 15 points. The dashed line crosses the four shaded cells.

(a) How many cells are there, as a function of n? Prove your answer is correct. (3)

(b) In the worst case, exactly how many cells can a horizontal line cross, as a function of n? Prove your answer is correct. Assume that $n = 2^k - 1$ for some integer k. (5)

(c) Suppose we are given n points stored in a kd-tree. Describe and analyze an algorithm that counts the number of points above a horizontal line (such as the dashed line in the figure) as quickly as possible. *[Hint: Use part (b).]* (6)

(d) Describe and analyze an efficient algorithm that counts, given a kd-tree containing n points, the number of points that lie inside a rectangle R with horizontal and vertical sides. *[Hint: Use part (c).]* (6)