Chennai Mathematical Institute

Design and Analysis of Algorithms

Assignment-1

Aug'25-Nov'25 Max Marks: 60

Note: Unless stated otherwise $\log n$ denotes $\log_2(n)$.

1. Show that for any positive integer n:
   (a) Prove or Disprove: $n! \stackrel{?}{\in} \Theta(n^n)$
   (b) Prove: $\log(n!) = \Theta(n \log n)$
   (c) Prove: $H_n = \sum_{i=1}^{n} \frac{1}{i} = \Theta(\log n)$ (hint: try plotting the curve $y = \frac{1}{x}$)

   (5)

2. (a) Suppose we are given two sorted arrays $A[1 \cdots n]$ and $B[1 \cdots n]$. Describe an algorithm to find the median of the union of A and B in $O(\log n)$ time. Assume the arrays contain no duplicate elements.
   (b) Now suppose we are given three sorted arrays $A[1 \cdots n]$, $B[1 \cdots n]$, and $C[1 \cdots n]$. Describe an algorithm to find the median element of $A \cup B \cup C$ in $O(\log n)$ time.
   (c) You are given two sorted lists of size m and n. Give an $O(\log m + \log n)$ time algorithm for computing the kth smallest element in the union of the two lists.

   (15)

3. You are given an array of N integers in which the first n of them are positive and the rest are negative, but you do not know n. Give an
   (a) $O(\log N)$ algorithm to find n.
   (b) $O(\log n)$ algorithm to find n.

   (10)

4. Given a list of n elements, give an efficient algorithm to determine whether they are all distinct. (5)

5. Solve the following recurrence relations and give a $\Theta(\cdot)$ bound for each of them. Assume that the functions evaluate to a constant for $n = 1$.
   (a) $T(n) = 2T(n/3) + 1$
   (b) $T(n) = 5T(n/4) + n$
   (c) $T(n) = T(n-1) + 2$
   (d) $T(n) = 2T(n-1) + 1$

(e) $T(n) = T(\sqrt{n}) + 1$ (Hint: assume $n = 2^k$ and rewrite the recurrence in terms of k.)

(10)

6. Suppose you are choosing between the following three algorithms:
   (a) Algorithm A solves problems by dividing them into five subproblems of half the size, recursively solving each subproblem, and then combining the solutions in linear time.
   (b) Algorithm B solves problems of size n by recursively solving two subproblems of size $n - 1$ and then combining the solutions in constant time.
   (c) Algorithm C solves problems of size n by dividing them into 9 subproblems of size $n/3$, recursively solving each subproblem, and then combining the solutions in $O(n^2)$ time.

   What are the running times of each of these algorithms (in Big-O notation), and which would you choose? (5)

7. Given two sets S_1 and S_2, and a real number x, find whether there exists an element from S_1 and an element from S_2 whose sum is exactly x. The algorithm should run in time $O(n \log n)$, where n is the total number of elements in both sets. (10)