Design and Analysis of Algorithms

Instructor : Dr. Sasanka Roy

September 26, 2023

1 Marks Distribution

• Assignment/Coding/Presentation	20 Marks
• Mid-Semester Examination	30 Marks
• End-Semester Examination	50 Marks
Total	**100 Marks**

2 Topics for Presentation

- λ-calculus Model of Computation

3 Topics for Coding

1. All Canonical Circles
2. Sorting by Permutations
3. Smallest Disk Center of Axis
4. Brute Force Linear Programming

4 Programming Problems

4.1 All Distinct Canonical disks

Input is a finite set of two-dimensional points denoted as $P = \{p_1, p_2, \ldots, p_n\}$, where each point p_i is represented as a pair of real numbers (x_i, y_i), denoting its Cartesian coordinates

A canonical disk set, $ds(c, r)$, of a disk d with center at a $c \in R^2$ and radius r, is defined as follows: $ds(c, r)= \{p_i \in P | dist(c, p_i) <= r\}$.

Problem: Find all possible distinct $ds(c, r)$.

4.2 Sorting by Permutations

The task is to create a program that will sort a given array of n distinct integers by generating and evaluating all possible permutations of the given n-sized input array. The goal is to find and output the permutation that represents the earliest non-decreasing arrangement, thereby sorting the array in ascending order. You must implement a brute-force approach to solve this problem, such that the program looks through all permutations and verify if the sequence is sorted.

4.3 Smallest Disk Center on x-axis

Write a program that efficiently computes the smallest radius disk that contain a given set of two-dimensional points P in the Cartesian plane and center of the disk must lie on one the x-axis

4.4 Brute Force Linear Programming

Develop a brute-force program for solving 2D linear programming problems. Input of the your program would be the coefficients of the objective function and the set of constraints in the form of linear inequalities. Your program should generate (i) all possible intersection points of the constraints (ii) check the feasibility of all the intersection points and obtain the optimum solution from the set of feasible intersection points.

5 Additional Problems (Optional)

5.1 Implementation of AVL Trees

Implement an AVL (Adelson-Velsky and Landis) tree data structure with various operations to maintain a self balanced binary search tree. The task is to implement the AVL tree and a set of operations to support insert, delete, search, and traverse (in-order traversal) the tree efficiently. Demonstrate various cases of rotations and verify if height(h) and number of nodes at height n denoted by (N_n) grow in accordance with the known relations of $h \leq \frac{3}{2} n \log_2(N)$ and $N_n \leq N_{n-1} + N_{n-2} + 1$