School of Computer Science
University of Guelph

CIS*3490 The Analysis and Design of Algorithms

Winter 2022
Instructor: Fangju Wang

Assignment 2 (100%)

In the following questions, express the algorithms in the pseudocode we are using (in the textbook and lecture slides), and implement your algorithms in the C programming language.

1. Counting Inversions (40%)

$A[0..n-1]$ is an array of n distinct numbers. A pair of array elements $(A[i], A[j])$ is called an inversion if $A[i] > A[j]$ for $i < j$.

1.1 Design a brute force algorithm to count the number of inversions in an array, analyze the number of executions of its basic operation, and determine the efficiency class.

1.2 Design a recursive divide-and-conquer algorithm of $\Theta(n \log n)$ to count the number of inversions in an array, set up a recurrence to analyze the number of executions of its basic operation of the best case, and determine the efficiency class. Use the Master Theorem to verify the efficiency class in your analysis result.

1.3 Implement the two algorithms, and test them by using *data_A2_Q1.txt*, which includes 50,000 integers. Your programs are required to display the numbers of inversions and execution time. Compare the two algorithms in execution time and theoretical analysis.

You can hard-code the name of data file in your programs. The file will be used to grade your programs.

2. Finding Shortest Path Around (60%)

Let S be a set of points in a 2-dimensional plane. The convex hull of set S is the smallest convex set containing S. (Please find more about the convex hull problem, especially the definition of *extreme point*, on pages 109-113 in the textbook.) It is assumed that not all the points in S are on a straight line.

The points in the convex hull can be viewed as fence posts that support a fence surrounding all the points in S. Let s_1 and s_2 be two points in the hull. A path from s_1 to s_2 is a sequence of points in the hull. By viewing points in the hull as fence posts, we may consider a path as a sequence of posts holding a fence segment. The problem of *finding shortest path around* is to find the shortest path from s_1 to s_2 that cannot cross inside the fenced area, but it goes along the fence. From s_1 to s_2, there are two paths along the fence, going in the two directions. The problem is to find the shorter one.

2.1 Design a brute force algorithm to solve the shortest path around problem and analyze its efficiency.

2.2 Design a recursive divide-and-conquer algorithm of $\Theta(n \log n)$ to solve the shortest path around problem, set up a recurrence to analyze the number of executions of the basic operation to compute the hull (for the best case), and determine the efficiency class. Use the Master Theorem to verify the efficiency class in your analysis result.

2.3 Implement the two algorithms and test them using *data_A2_Q2.txt*. The file contains 30,000 points (pairs of x-y coordinates). Your programs will be graded by using this file and two points s_1 and s_2 that are in the hull of the points in the file. Your programs are required to display the number of points on the path, the length of the path, and x-y coordinates of the points (in the order from s_1 to s_2 inclusive). Your programs are also required to display execution time for *hull computing*. Compare the two algorithms in execution time and theoretical analysis.

You can hard-code the name of data file in your programs. The file will be used to grade your programs. Your program should prompt the user to enter s_1 and s_2.

Note: Write your own code for this assignment. NO code from any source is allowed.

Due time: 08:00am, Monday, February 14, 2022. Submit your work as a tar file to Moodle.