Midterm Exam

CS 4104 (Summer 2024)

Submit a PDF file containing your solutions on Canvas by 11:59pm on Friday June 28, 2024.

Instructions:

- The Honor Code applies to this homework with the following exception:
  - You are allowed to discuss possible algorithms and bounce ideas with your classmates. Do not discuss proofs of correctness, final answer or running time in detail with your classmate. You must write down your solution individually and independently. Do not send a written solution to your classmates for any reason whatsoever.
  - You are not allowed to consult sources other than your textbook, the slides on the course web page, your own class notes, the TA, and the instructor. **In particular, do not use a search engine or large language models such as ChatGPT**.
- Do not forget to typeset your solutions. Every mathematical expression must be typeset as a mathematical expression, e.g., the square of n must appear as n^2 and not as `n2`. You can use the LaTeX version of the homework problems to start entering your solutions. At the end of each problem are three commented lines that look like the example below. You can uncomment these lines and type in your solution within the curly braces.

```
% \solution{
%
% }
```

- Do not make any assumptions not stated in the problem. If you do make any assumptions, state them clearly, and explain why the assumption does not decrease the generality of your solution
- You must also provide a clear proof that your solution is correct (or a counter-example, where applicable). Type out all the statements you need to complete your proof. You must convince us that you can write out the complete proof. You will lose points if you work out some details of the proof in your head but do not type them out in your solution.
- If you are proposing an algorithm as the solution to a problem, keep the following in mind:
  - Describe your algorithms as clearly as possible. The style used in the text book is fine, as long as your description is not ambiguous. Explain your algorithm in words. A step-wise description is fine. *However, if you submit detailed pseudo-code without an explanation, we will not grade your solutions.*Do not describe your algorithms only for a specific example you may have worked out.
  - Make sure to state and prove the running time of your algorithm. You will only get partial credit if your analysis is not tight, i.e., if the bound you prove for your algorithm is not the best upper bound possible.
  - You will get partial credit if your algorithm is not the most efficient one that is possible to develop for the problem.
- In general for a graph problem, you may assume that the graph is stored in an adjacency list and that the input size is m + n, where n is the number of nodes and m is the number of edges in the graph. Therefore, a linear time graph algorithm will run in $O(m + n)$time.

Problem 1 (5 Points) Prove that $7n^3 + 2n^2 + 4$ is $\Theta(n^3)$.

Problem 2 (5 Points) Prove that $5n \log n + 10n$ is $\Omega(n \log n)$.

Problem 3 (5 Points) Prove that $2^{n+1} + n^2$ is $O(2^n)$.

Problem 4 (10 Points) Prove that $n^{\log n} + 2^n$ is $O(2^n)$.

Problem 5 New team members are joining the Hokies, and there is a reassignment going on in the locker room. A team member is assigned a single locker. Each locker is numbered (both positive and negative), and when someone moves the contents of a locker to any other locker, the moving time is proportional to the distance between the lockers. If the contents of locker i are moved to locker j, it takes $|j - i|$ hours. For example, to move contents of locker 5 to locker 3, it takes 2 hours. To move the contents of locker 2 to 6 takes 4 hours.

As a CS4104 student, the facility manager has asked you to design an algorithm that, given current positions of members who need to be (re)assigned and a proposed locker position, can assign lockers with the aim of minimizing the time it takes for the last person to move. Below are example inputs and expected outputs:

- **Instance 1:**
  Current assignments (C): [4, -4, 2]
  Available lockers (A): [4, 0, 5]
  Output: 4
- **Instance 2:**
  Current assignments (C): [-10, -79, -79, 67, 93, -85, -28, -94]
  Available lockers (A): [-2, 9, 69, 25, -31, 23, 50, 78]
  Output: 102

Your task is to

1. (15 Points) Design an efficient algorithm and provide the pseudo-code. (Hint: The brute force approach is to check all permutations and find the one that has the lowest time for the last person)
2. (5 Points) Prove the correctness of the algorithm
3. (5) Calculate the time complexity

Problem 6 Imagine there are gardeners and flowers arranged in a single line. Each gardener can only water flowers within a certain range. You need to design an algorithm to maximize the number of flowers that get watered by the gardeners. Given an array that represents gardeners (denoted by 'G') and flowers (denoted by 'F') in a single line, design a greedy algorithm to maximize the number of flowers that get watered by the gardeners. Each gardener can water flowers within a certain range, say k positions to their left and right. The goal is to ensure that the maximum number of flowers are watered by the gardeners.

Problem Statement

Input: The input consists of:

- An array of characters where each character is either 'G' (gardener) or 'F' (flower).
- An integer k which denotes the range within which each gardener can water flowers.

Output: The output should be the maximum number of flowers that can be watered by the gardeners.

Example

Input: $arr = ['G',' F',' F',' G',' F',' F',' G'], k = 2$

Output: 5

Input: $arr = ['F',' F',' G',' F',' F',' G',' F',' G',' F'], k = 1$

Output: 6

1. (15 Points) Design an efficient greedy algorithm and provide the pseudo-code. Your algorithm should efficiently maximize the number of flowers that are watered by the gardeners.
2. (5 Points) Prove the correctness of the algorithm
3. (5 Points) Provide the time complexity of your solution.

Problem 7 You have just been assigned to run a network of warehouses and delivery system. Every warehouse in the network has at most one delivery route going into it and at most one delivery route going out of it. Central depots and delivery points are to be installed in a manner such that every warehouse with one outgoing route but no incoming route gets a central depot installed, and every warehouse with only an incoming route and no outgoing route gets a delivery point.

Given two integers n and p denoting the number of warehouses and the number of delivery routes, respectively, the connections of routes among the warehouses contain three input values: a_i, b_i, d_i denoting the route of distance d_i from warehouse a_i to warehouse b_i. Find an efficient solution for the network.

Problem Statement

Input: The input consists of:

- A line containing n and p
- n denotes the number of warehouses.
- p denotes the number of delivery routes.
- There will be p lines containing triples (a_i, b_i, d_i) where a_i is the starting warehouse, b_i is the ending warehouse, and d_i is the distance of the route.

Output: The output should contain:

- The number of pairs of central depots and delivery points t installed in the first line.
- The next t lines should contain three integers: the warehouse number of the central depot, the warehouse number of the delivery point, and the minimum distance of the route between them.

Example

Input:

```
5 4
1 2 10
2 3 5
3 4 7
4 5 6
```

Output:

```
1
1 5 5
```

1. (20 Points) Design and explain a greedy algorithm that solves this problem. Your algorithm should efficiently determine the optimal placement of central depots and delivery points. Provide the time complexity of your solution.
2. (5 Points) What would be the output of your algorithm for the following input
   Input:

```
6 5
1 2 8
2 3 6
4 5 12
5 6 15
3 4 10
```