Homework 5

CS 4104 (Summer 2024)

Assigned on July 31, 2024.
Submit a PDF file containing your solutions on Canvas by 11:59pm on Tuesday August 06, 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 `n^2`. 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.

Acknowledgement: Some of the questions in this Homework are inspired by/adopted from previous CS 4104 offerings at VT including by T. M. Murali and others.

Problem 1 (10 Points) Use the substitution method to guess and prove that the running time of $T(n) = T(\frac{n}{3}) + \Theta(\sqrt{n})$.

Problem 2 (25 Points) **Design** and **analyze** a divide and conquer algorithm to find the outer contour of a set of buildings. Each building is represented as a triplet (L_i, R_i, H_i), where L_i is the left x-coordinate, R_i is the right x-coordinate, and H_i is the height.
Example 1:

- Input: $[(1, 3, 4), (2, 4, 5), (5, 6, 3)]$
- Expected Output: $[(1, 4), (2, 5), (4, 0), (5, 3), (6, 0)]$

Example 2:

- Input: $[(2, 9, 10), (3, 7, 15), (5, 12, 12), (15, 20, 10), (19, 24, 8)]$
- Expected Output: $[(2, 10), (3, 15), (7, 12), (12, 0), (15, 10), (20, 8), (24, 0)]$

Problem 3 (25 Points) **Design** and **analyze** a divide and conquer algorithm to find the length of the longest bitonic subsequence in an array. A bitonic subsequence is a sequence that first increases and then decreases.
Example 1:

- Input: $[18, 2, 3, 4, 5, 3, 2, 18, 8]$
- Expected Output: 6 (Subsequence: $[2, 3, 4, 5, 3, 2]$)

Example 2:

- Input: $[12, 11, 40, 5, 3, 41]$
- Expected Output: 5 (Subsequence: $[12, 11, 40, 5, 3]$)

Problem 4 (20 Points) Given a set of points in a 2D plane, design a greedy algorithm to find the minimum cost to connect all the points such that the total cost of the connections is minimized. The cost of connecting two points $(x1, y1)$ and $(x2, y2)$ is the Manhattan distance $|x1 - x2| + |y1 - y2|$.
Example 1:

- Input: $[(0, 0), (2, 2), (3, 10), (5, 2), (7, 0)]$
- Expected Output: 20

Example 2:

- Input: $[(3, 12), (5, 10), (7, 9), (2, 6), (5, 5), (8, 4)]$
- Expected Output: 18

Problem 5 (20 Points) Given a string s, design a dynamic programming algorithm to find the length of the longest palindromic subsequence in s. A subsequence is a sequence that can be derived from another sequence by deleting some elements without changing the order of the remaining elements.
Example 1:

- Input: $s = "bbbab"$
- Expected Output: 4 (Longest palindromic subsequence: "bbbb")

Example 2:

- Input: $s = "cbbd"$
- Expected Output: 2 (Longest palindromic subsequence: "bb")

Example 3:

- Input: $s = "agbdba"$

- Expected Output: 5 (Longest palindromic subsequence: "abdba")

Example 4:

- Input: $s = "abcdefg"$
- Expected Output: 1 (Longest palindromic subsequence: any single character, e.g., "a")

Example 5:

- Input: $s = "aabca"$
- Expected Output: 3 (Longest palindromic subsequence: "aaa" or "aca")

Problem 6 (20 Points) Given a string s and a dictionary of strings $wordDict$, design a dynamic programming algorithm to determine if s can be segmented into a space-separated sequence of one or more dictionary words.

Example 1:

- Input: $s = "codeworld"$, $wordDict = ["code", "world"]$
- Expected Output: true (Segmentation: "code world")

Example 2:

- Input: $s = "applepenapple"$, $wordDict = ["apple", "pen"]$
- Expected Output: true (Segmentation: "apple pen apple")