CSCI 411 - Advanced Algorithms and Complexity
Exam 1 - Written and Coding Questions

March 19, 2023

Solutions to the written portion of the exam should be submitted via PDF to Canvas. Make sure to justify your answers. C++ code should be submitted both on Canvas and on turnin. The exam will close on **April 3rd at 11:59 pm**. Note that there are a total of 65 points on this portion of the exam. 5 of these are extra credit.

You must complete solutions to these problems **individually**. You may ask and answer questions on Discord or Canvas meant to clarify the problems but you may not discuss or post solutions. You are allowed to use your notes, material posted on Canvas, and the textbook (*Introduction to Algorithms*, Thomas Cormen, et al.) during the exam. However, you **are not permitted** to use any other resources. This includes Google, Stack Overflow, and all other websites and documents available online.

Good luck!

Written Problems

1. Given a string S and access to a function `makeSuffixArray(S)` which creates and returns the suffix array of S in linear time, determine the number of times that a string R occurs as a substring in S as efficiently as possible.

   (a) (3 pts) Describe an intuitive approach for solving this problem. Be as clear and precise as possible.

   (b) (5 pts) Write pseudocode for a function `numOccurrencesA(S, R)` which returns the number of times R appears in S as a substring. You may use the function `makeSuffixArray(S)`.

   (c) (2 pts) Analyze the asymptotic run time of your algorithm in terms of n, the length of S, and m, the length of R.

2. Given a string S and access to a function `makeSuffixTree(S)` which creates and returns the suffix tree of S in linear time, determine the number of times that a string R occurs as a substring in S as efficiently as possible.

   (a) (3 pts) Describe an intuitive approach for solving this problem. Be as clear and precise as possible.

   (b) (5 pts) Write pseudocode for a function `numOccurrencesT(S, R)` which returns the number of times R appears in S as a substring. You may use the function `makeSuffixTree(S)`.

   (c) (2 pts) Analyze the asymptotic run time of your algorithm in terms of n, the length of S, and m, the length of R.

3. Given an undirected, unweighted graph $G = (V, E)$, determine the number of unique shortest paths from $s \in V$ to $t \in V$.

   (a) (3 pts) Describe an intuitive approach for solving this problem. Be as clear and precise as possible.

   (b) (5 pts) Write pseudocode for a function `countShortestPaths(G, s, t)` which returns the number of unique shortest paths from s to t in G.

   (c) (2 pts) Analyze the asymptotic run time of your algorithm.

4. Given a set of coin denominations C, determine the total number of ways to produce an amount of money m where order matters. For example, if $C = \{1, 3\}$ and $m = 6$, there are six ways to reach m:

$$[1, 1, 1, 1, 1, 1]$$
$$[1, 1, 1, 3]$$
$$[1, 1, 3, 1]$$
$$[1, 3, 1, 1]$$
$$[3, 1, 1, 1]$$
$$[3, 3]$$

   (a) (5 pts) Describe the optimal substructure of this problem. In particular, define the solution for an amount of money m in terms of solutions for amounts $\mu < m$. Justify your answer.

   (b) (4 pts) Write pseudocode for a function `numWays(m, C)` which returns the number of ways in which the coins of C can be arranged to reach m.

   (c) (1 pts) Analyze the asymptotic run time of your algorithm in terms of both m, the target amount of money, and $|C|$, the number of available coin denominations.

5. Given a multiset of positive integers S and a target value m, determine whether or not there is a subset $R \subseteq S$ such that $\sum_{r \in R} r = m$ using dynamic programming. Consider similarities between this problem and the 0-1 knapwsack problem.

   (a) (5 pts) Describe the optimal substructure of this problem. In particular, define the solution for the full set S and full target m in terms of solutions to subsets of S and targets $\tau < m$. Justify your answer.

   (b) (4 pts) Write pseudocode for a function `subsetSum(S, m)` which returns `true` if the target can be achieved using the elements of S and `false` otherwise.

   (c) (1 pts) Analyze the asymptotic run time of your algorithm in terms of both n, the size of S, and m, the target.

Coding Problem

(10 pts) Write a C++ implementation of the pseudocode you developed for problem (5b) and submit to Canvas and to turnin as exam_1.cpp. You may find the skeleton code in exam_1_skeleton.cpp on Canvas helpful.

- Input will come from cin
  - The first line will contain two integers, n and m, separated by a space.
    - n is the size of the multiset S.
    - m is the target.
  - The second line contains n space separated positive integers.
- Print output to cout
  - Print `true` if m can be achieved using the elements of S and `false` otherwise.

Examples

In the following examples, red values represent one valid subset achieving the target.

Example 1:

Input:
3 10
5 3 1

Expected output:
false

Example 2:

Input:
3 10
2 5 3

Expected output:
true

Example 3:

Input:
5 17
16 3 9 2 55

Expected output:
false

Example 4:

Input:
5 19

16 3 9 2 55 17

Expected output:
true