CS 170 Homework 3

Due **2/14/2022, at 10:00 pm (grace period until 11:59pm)**

1 Study Group

List the names and SIDs of the members in your study group. If you have no collaborators, you must explicitly write "none".

2 Preorder, Postorder

Suppose we just ran DFS on a directed (not necessarily strongly connected) graph G starting from vertex r, and have the pre-visit and post-visit numbers $pre(v), post(v)$ for every vertex. We now delete vertex r and all edges adjacent to it to get a new graph G'. Given *just* the arrays $pre(v), post(v)$, describe how to modify them to arrive at new arrays $pre'(v), post'(v)$ such that $pre'(v), post'(v)$ are a valid pre-visit and post-visit ordering for some DFS of G'.

3 Where's the Graph?

Each of the following problems can be solved with techniques taught in lecture. Construct a simple directed graph and write an algorithm for each problem by black-boxing algorithms taught in lecture and in the textbook.

(a) Sarah wants to do an extra credit problem for her math class. She is given three numbers: 1, x, and y. Starting from x, she needs to find the shortest sequence of additions, subtractions, and divisions (only possible when the number is divisble by y) using 1 and y to get to 2021. If there are multiple sequences with the shortest length, return any one of them. She can use 1 and y multiple times. Give an algorithm that Sarah can query to get this sequence of arithmetic operations.

(b) There are n different species of Gem Berry, all descended from the original Stone Berry. For any species of Gem Berry, Emily knows all of the species *directly* descended from it. Emily wants to write a program. There would be two inputs to her program: a and b, which represent two different species of Gem Berries. Her program will then output one of three options in constant time (the time complexity cannot rely on n):

(1) a is descended from b.

(2) b is descended from a.

(3) a and b share a common ancestor, but neither are descended from each other.

Unfortunately, Emily is very limited in space and cannot store all of the descendants of a given species for all of the species. Give an algorithm that Emily's program could use to solve the problem above given the constraints. Emily can run some algorithm on her data and store the outputs of the algorithm for her program.

(c) Bob has n different boxes. He wants to send the famous "Blue Roses' Unicorn" figurine from his glass menagerie to his crush. To protect it, he will put it in a sequence of boxes. Each box has a weight w and size s; with advances in technology, some boxes have negative weight. A box a inside a box b cannot be more than 15% smaller than the size of box a; otherwise, the box will move, and the precious figurine will shatter. The figurine needs to be placed in the smallest box x of Bob's box collection.

Bob (and Bob's computer) can ask his digital home assistant Falexa to give him a list of all boxes less than 15% smaller (but not necessarily lighter) than a given box c. Bob will need to pay postage for each unit of weight. Find an algorithm that will find the lightest sequence of boxes that can fit in each other in linear time.

4 The Greatest Roads in America

Arguably, one of the best things to do in America is to take a great American road trip. And in America there are some amazing roads to drive on (think Pacific Crest Highway, Route 66 etc). An intrepid traveler has chosen to set course across America in search of some amazing driving. What is the length of the shortest path that hits at least k of these amazing roads?

Assume that the roads in America can be expressed as a directed weighted graph $G = (V, E, d)$, and that our traveler wishes to drive across at least k roads from the subset $R \subseteq E$ of "amazing" roads. Furthermore, assume that the traveler starts and ends at her home $h \in V$. You may also assume that the traveler is fine with repeating roads from R, i.e. the k roads chosen from R need not be unique.

Design an efficient algorithm to solve this problem. Provide a 3-part solution with runtime in terms of $n = |V|$, $m = |E|$, k.

Hint: Create a new graph G' based on G such that for some s', t' in G', each path from s' to t' in G' corresponds to a path of the same length from h to itself in G containing at least k roads in R. It may be easier to start by trying to solve the problem for $k = 1$.

5 Pattern Matching

Consider the following string matching problem:
Input:

- A string g of length n made of 0s and 1s. Let us call g, the "pattern".
- A string s of length m made of 0s and 1s. Let us call s the "sequence".
- Integer k

Goal: Find the (starting) location of all length n-substrings of s which match g in at least $n - k$ positions.
Example: Using 0-indexing, if $g = 0111$, $s = 01010110111$, and $k = 1$ your algorithm should output 0,2,4 and 7.

(a) Give a $O(nm)$ time algorithm for this problem.

We will now design an $O(m \log m)$ time algorithm for the problem using FFT. *Pause a moment here to contemplate how strange this is. What does matching strings have to do with roots of unity and complex numbers?*

(b) Devise an FFT based algorithm for the problem that runs in time $O(m \log m)$. Write down the algorithm, prove its correctness and show a runtime bound.

Hint: On the example strings g and s, the first step of the algorithm is to construct the following polynomials

$$0111 \to 1 + x + x^2 - x^3$$
$$01010110111 \to -1 + x - x^2 + x^3 - x^4 + x^5 + x^6 - x^7 + x^8 + x^9 + x^{10}$$

(c) (Extra Credit) Often times in biology, we would like to locate the existence of a gene in a species' DNA. Of course, due to genetic mutations, there can be many similar but not identical genes that serve the same function, and genes often appear multiple times in one DNA sequence. So a more practical problem is to find all genes in a DNA sequence that are similar to a known gene.

This problem is very similar to the one we solved earlier, the string s is complete sequence and the pattern g is a specific gene. We would like to find all locations in the complete sequence s, where the gene g appears, but for k modifications.

Except in genetics, the strings g and s consist of one of four alphabets $\{A, C, T, G\}$ (not 0s and 1s). Can you devise an $O(m \log m)$ time algorithm for this modified problem?