CS 170 Spring 2017 — Efficient Algorithms and Intractable Problems — Prasad Raghavendra and Sanjam Garg

Homework 4

Instructions: You are welcome to form small groups (up to 4 people total) to work through the homework, but you **must** write up all solutions by yourself. List your study partners for homework on the first page, or "none" if you had no partners.

If using LaTeX (which we recommend), you may use the homework template linked on this Piazza post to get started.

Begin each problem on a new page. Clearly label where each problem and subproblem begin. The problems must be submitted in order (all of P1 must be before P2, etc). For questions asking you to give an algorithm, respond in what we will refer to as the *four-part format* for algorithms: main idea, pseudocode, proof of correctness, and running time analysis.

Read the Homework FAQ Piazza post on Piazza before doing the homework for more explanation on the four-part format and other clarifications for our homework expectations.

No late homeworks will be accepted. No exceptions. This is not out of a desire to be harsh, but rather out of fairness to all students in this large course. Out of a total of approximately 12 homework assignments, the lowest two scores will be dropped.

Special Questions:

- *Shortcut questions*: Short questions are usually easy questions that give you opportunities to practice basic materials. However, we understand that some of you are very familiar with the topics and do not want to spend too much time on easy questions. Therefore, we design shortcut questions for this purpose. A shortcut question usually has multiple parts that build upon each other and are ordered by their difficulty level. You can work on those in order or start from wherever you like. However you only need to submit the last part you are able to solve. For example, if a question has 5 parts (a, b, c, d, e), you are confident about part e, you should submit part e without any of the previous four parts. If you are confident about d but not sure about e, you should submit d for grading purposes. Please clearly indicate in your submission which part you are submitting.
- *Redemption questions*: It is important that you carefully read the posted solutions, even for problems you got right. To encourage this, you have the option of submitting a redemption file, a few paragraphs in which you explain, for each problem you choose to cover, what you did wrong and what the right idea was in your own words (not cutting and pasting from the solution!), and appending it to your homework. For example, suppose that as you review your solutions to HW1, you realize you had misunderstood question 3 and answered it incorrectly. You would write down what you just learned, and then submit it in your HW2 assignment the following week. Because these are mainly for your benefit, feel free to format them however is most useful for you.
- *Extra credit questions*: We might have some extra credit questions in the homework for people who really enjoy the materials. However, please note that you should do the extra credit problems only if you really enjoy working on these problems and want an extra challenge. It is likely not the most efficient manner in which to maximize your score.

Due Tuesday, February 28, at 11:59am

1. (★★ level) Short Questions

1. Let G be a connected undirected graph with positive lengths on all the edges. Let s be a fixed vertex. Let $d(s,v)$ denote the distance from vertex s to vertex v, i.e., the length of the shortest path from s to v. If we choose the vertex v that makes $d(s,v)$ as small as possible, subject to the requirement that $v \neq s$, then does every edge on the path from s to v have to be part of every minimum spanning tree of G?
2. The same question as above, except now no two edges can have the same length.

2. (★★ level) Huffman Encoding

We use Huffman's algorithm to obtain an encoding of alphabet $\{a,b,c\}$ with frequencies f_a, f_b, f_c. In each of the following cases, either give an example of the frequencies (f_a, f_b, f_c) that would yield the specified code, or explain why the code cannot possibly be obtained (no matter what the frequencies are).

(a) Code: $\{0,10,11\}$

(b) Code: $\{0,1,00\}$

(c) Code: $\{10,01,00\}$

3. (★★★★ level) Preventing Conflict

A group of n guests shows up to a house for a party, and any two guests are either friends or enemies. There are two rooms in the house, and the host wants to distribute guests among the rooms, breaking up as many pairs of enemies as possible. The guests are all waiting outside the house and are impatient to get in, so the host needs to assign them to the two rooms quickly, even if this means that it's not the best possible solution. Come up with an efficient algorithm that breaks up at least half the number of pairs of enemies as the best possible solution, and prove your answer.

Hint: Try assigning guests one at a time. Consider how many pairs of enemies are broken up with each iteration.

4. (★★ level) Graph Subsets

Let $G = (V,E)$ be a connected, undirected graph, with edge weights $w(e)$ on each edge e. Some edge weights *might be negative*. We want to find a subset of edges $E' \subseteq E$ such that $G' = (V,E')$ is connected, and the sum of the edge weights in E' is as small as possible.

1. Is it guaranteed that the optimal solution E to this problem will always form a tree?
2. Does Kruskal's algorithm solve this problem? If yes, explain why in a sentence or two; if no, give a small counterexample.
3. Briefly describe an efficient algorithm for this problem. Just the main idea is enough (1-3 sentences). No need for a 4-part solution.

5. (★★★ level) Arbitrage Shortest-path algorithms can also be applied to currency trading. Suppose we have n currencies $C = \{c_1, c_2, \dots, c_n\}$: e.g., dollars, Euros, bitcoins, dogecoins, etc. For any pair i,j of currencies, there is an exchange rate $r_{i,j}$: you can buy $r_{i,j}$ units of currency c_j at the price of one unit of currency c_i. Assume that $r_{i,i} = 1$ and $r_{i,j} \geq 0$ for all i,j.

(a) The Foreign Exchange Market Organization (FEMO) has hired Oski, a CS170 alumnus, to make sure that it is not possible to generate a profit through a cycle of exchanges; that is, for any currency $i \in C$, it is not possible to start with one unit of currency i, perform a series of exchanges, and end with more than one unit of currency i. (That is called *arbitrage*.) Give an efficient algorithm for the following problem: given a set of exchange rates $r_{i,j}$ and two specific currencies s,t, find the most advantageous sequence of currency exchanges for converting currency s into currency t. We recommend that you represent the currencies and rates by a graph whose edge lengths are real numbers.

(b) In the economic downturn of 2016, the FEMO had to downsize and let Oski go, and the currencies are changing rapidly, unfettered and unregulated. As a responsible citizen and in light of what you saw in lecture this week, this makes you very concerned: it may now be possible to find currencies $c_{i_1}, \dots, c_{i_k}$ such that $r_{i_1,i_2} \times r_{i_2,i_3} \times \cdots \times r_{i_{k-1},i_k} \times r_{i_k,i_1} > 1$. This means that by starting with one unit of currency c_{i_1} and then successively converting it to currencies $c_{i_2}, c_{i_3}, \dots, c_{i_k}$ and finally back to c_{i_1}, you would end up with more than one unit of currency c_{i_1}. Such anomalies last only a fraction of a minute on the currency exchange, but they provide an opportunity for profit.

You decide to step up to help out the World Bank. Given an efficient algorithm for detecting the presence of such an anomaly. You may use the same graph representation as for part (a).