Homework 7: Approximation and Final Review

Version 2: We added some hints to 2.2 and corrected a typo that said "vertex" instead of "variable", and clarified in 2.1 for the recurrence we only need the value.

Due Date: This assignment is due at 11:59 PM Friday March 12 (Seattle time, i.e. GMT-8).

You will submit the written problems as a PDF to gradescope. Please put each numbered problem on its own page of the pdf (this will make selecting pages easier when you submit).

Remember to assign pages in your submission!

Collaboration: Please read the full collaboration policy. If you work with others (and you should!), you must still write up your solution independently and name all of your collaborators somewhere on your assignment.

Style and Assumptions: You should check the Style Guide for general principles on how to format your answers. You may use any algorithm described in this list without further explanation. If you wish to use an algorithm you learned in 373 (or 142/143) that isn't on the list, please ask on Ed so we can add it to the list for everyone.

1. Practice Exam [20 points]

Here is practice material that was given last quarter (it was originally a final for a related course). Problems 1-5 of the exam are a mix of short answer and mechanical questions. Problems 6-10 are more traditional "design an algorithm problems."

Set yourself a timer of 2 hours, to attempt this exam under "test conditions" (e.g. writing out your answers on paper, try to work straight through without taking breaks, etc.).[1] You will **not** be turning in these problems for grading, instead you'll use it to frame any additional studying you want to do.

Specifically, do **not** try to do all the problems (this exam is A LOT for the 2 hour time frame it was originally given in). Instead,

- Start your timer. Read through every problem on the exam.
- Of problems 1-5, choose 2 that you intend to do.
- Of problems 6-10, choose 3 that you intend to do.
- In your remaining time, do your best to write up solutions to 2 problems from the first half and 3 from the second half.
- When your timer goes off STOP.

After the two hours ends, answer the following questions.

(a) In the first half, which problems did you decide to do? Did you feel like you got through all of them?

(b) In the second half, which problems did you decide to do? Did you feel like you got through all of them?

(c) Look at the solutions for just the problems you attempted. For each question, did you get it right? If not, is there a takeaway for you for something you should study/do in the real exam?

(d) Look at the problems you chose to do **and** the problems you skipped, is there a topic (or multiple topics) you think you are most comfortable with? Least comfortable with? Do you want to prioritize any of those topics in your studying?

[1] Even though you won't need to be under "test conditions" for your exam, it's still helpful for studying.

2. Choose Your Own Adventure

Do **at least** one of these two problems. If you do both, we'll give you the higher score in the gradebook and convert the lower score to extra credit.

2.1. DP!

We said in class that finding a longest path is NP-hard.

(a) This result is very conter-intuitive. It seems like we should be able to just take the DPs we wrote for Floyd-Warshall or Bellman-Ford and flip the mins to maxes! This idea does not work because of cycles (we want a path that does not repeat vertices, but if we can repeat vertices we can always go along a positive weight cycle repeatedly, and the recurrences don't explicitly prevent that). Think about why the recurrences fail when there are cycles (you might want to do an example or two). You do not have to write anything for this part [0 points]

(b) Write a formula for a recurrence for finding the **length of** the longest path in a weighted directed acyclic graph. (There's no need to totally reinvent the wheel! You might want to start with the recurrence on slide 7 here) [10 points]

(c) Describe what the recurrence calculates in 1-2 sentences of English. Be sure to describe what any parameters represent. [3 points]

(d) What would a return statement look like in a method to calculate the longest path in the whole graph (e.g. what input(s) to the recurrence would you look up? If you look up more than one value, what is your overall answer?) [2 points]

(e) Describe an evaluation order to evaluate your recurrence (you do not have to write the actual code). I.e. tell us which values to calculate first and which to calculate last. (Our explanation is 1 sentence). [3 points]

(f) Wait! Finding longest paths is NP-hard, have you shown $P = NP$? Describe why or why not in 1-3 sentences [2 points]

2.2. Approximation

A **matching** is a set of edges so that no two edges share an endpoint. You write the following LP for the maximum matching problem. You have a variable x_e for every edge e.

max $\sum_e x_e$

Subject to:

$\sum_{e:u \text{is an endpoint of } e} x_e \leq 1$ for every vertex u

$0 \leq x_e \leq 1$ for every edge e

Your output might be fractional, but you can assume that every variable is set to $0, 1,$ or $1/2$.[2]. We'll round to an integer matching, while ensuring that if the LP finds a (fractional) matching with k "edges", you find a (true) matching of at least $2k/3$ edges.

(a) Describe how to produce an integer matching (your algorithm should run in polynomial time, but do not worry about getting the best possible efficiency; we won't deduct for slow polynomial time algorithms) [8 points]
Hint: Think about the degree of vertices if you only include edges that have their variables set to more than 0. What can you say about what the graph looks like?

[2]For this particular type of LP, there are some LP solvers which can do that automatically. You don't need to worry about why (look up "basic feasible solution" if you're really curious); for designing the algorithm you just need to know that this is possible

(b) Argue that your algorithm is polynomial time (our argument is 2 sentences) [2 points]

(c) Argue that your algorithm prodcues a "real" matching (i.e. that you no longer have any fractional variables, and that you really have a matching, not just some set of edges). Our argument is 3 sentences. [4 points]

(d) Argue that your true matching has at least $2k/3$ edges, where $k = \sum_e x_e$ for the x_e found by the LP. Our argument is 2-3 sentences (and a formula) [4 points]

(e) Give an example of a graph where a maximum value of the LP really is more than the value of any matching (hint: your graph must not be bipartite for this to be possible) [2 points]