CS 170
Fall 2017

Efficient Algorithms and Intractable Problems
Prasad Raghavendra and Sanjam Garg

Homework 8

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 Wednesday, November 8, at 4:59pm

0. Who did you work with?

List all your collaborators on this homework. If you have no collaborators, please list "none".

1. (★★★ level) iPhone X

Apple has just gotten hacked and their instruction manual for producing their new iPhone X has been released to the public. A couple factories that specialize in producing knockoff popular consumer products take advantage of this and start producing their own cheaper versions of the iPhone X. These factories do not have the means for distribution, but fortunately, a few companies that specialize in distribution of knockoff products are interested in this promising business opportunity. There are a total of m factories, each capable of producing a_i knockoff iPhones. There are also n distributors, each demanding b_i knockoff iPhones.

Note: Solve parts $(b), (c)$ independently of each other.

(a) Each factory i can supply to any distributor they choose. Find an efficient algorithm to determine whether it is feasible for all demand to be met.

(b) Each factory i is willing to deliver to distributors at most c_i distance away. Each factory i has a distance $d_{i,j}$ from distributor j. Solve part (a) with this additional constraint.

(c) Each factory and distributor now belongs to one of p different countries. Each country has a maximum limit on the amount of iPhones that can be imported (e_k) or exported (f_k). Deliveries within the same country don't contribute towards this limit. Solve part (a) with this additional constraint.

2. (★★★ level) What happens in Vegas...

The Venetian at Las Vegas has just introduced a brand new game, Rambis. The rules are simple. You and the dealer both have three cards - Jack, Queen, and King - and you both place one on the table face down. Then the dealer will flip both the cards and pay you a certain amount based on the table below.

Note that a negative value means you owe the casino that amount. For example, if you chose Jack and the dealer also chose Jack, you would owe the casino $10.

		Dealer: Jack	Queen	King
You:	Jack	-10	3	3
	Queen	4	-1	-3
	King	6	-9	2

Feel free to use an online LP solver to solve your LPs in this problem.
Here is an example of a Python LP Solver and its Tutorial.

(a) Write an LP to find your optimal strategy. What is the optimal strategy and expected payoff?

(b) Now do the same for the dealer. What is the optimal strategy and expected payoff?

3. (★★★★ level) Minimum Spanning Trees

Consider the spanning tree problem, where we are given an undirected and connected graph $G : (V, E)$ with edge weights $w_{u,v}$ for every pair of vertices u, v.

An integer linear program that solves the minimum spanning tree problem is as follows:

$$\begin{aligned}
&\text{Minimize} \sum_{(u,v)\in E} w_{u,v} x_{u,v} \\
&\text{subject to} \sum_{\{u,v\}\in E : u\in L, v\in R} x_{u,v} \geq 1 \quad \text{for all partitions of } V \text{ into disjoint nonempty sets } L, R \\
&\qquad x_{u,v} \in \{0,1\}, \quad \forall (u,v) \in E
\end{aligned}$$

(a) Give an interpretation of the purpose of the objective function, decision variables, and constraints.

(b) Is creating the formulation a polynomial time algorithm with respect to the size of the input graph?

(c) Suppose that we relaxed the binary constraint on the decision variables $x_{u,v}$ with the non-negativity constraint:

$$x_{u,v} \geq 0, \quad \forall (u,v) \in E$$

How does the new linear program solution's objective value compare to the integer linear program's? Provide an example (decision variables and objective value) where the above LP relaxation achieves a better objective value than the ILP formulation.

4. (★★★ level) Decision vs. Search vs. Optimization

The following are three formulations of the VERTEX COVER problem:

- As a *decision problem*: Given a graph G, return TRUE if it has a vertex cover of size at most b, and FALSE otherwise.
- As a *search problem*: Given a graph G, find a vertex cover of size at most b (that is, return the actual vertices), or report that none exists.
- As an *optimization problem*: Given a graph G, find a minimum vertex cover.

At first glance, it may seem that search should be harder than decision, and that optimization should be even harder. We will show that if any one can be solved in polynomial time, so can the others:

Describe your algorithms precisely; justify correctness and running time. No pseudocode.

Hint for both parts: Call the black box more than once.

(a) Suppose you are handed a black box that solves VERTEX COVER (DECISION) in polynomial time. Give an algorithm that solves VERTEX COVER (SEARCH) in polynomial time.

(b) Similarly, suppose we know how to solve VERTEX COVER (SEARCH) in polynomial time. Give an algorithm that solves VERTEX COVER (OPTIMIZATION) in polynomial time.

5. (★★★★ level) Max-Flow Variants

Show how to reduce the following variants of Max-Flow to the regular Max-Flow problem, i.e. do the following steps for each variant: Given a graph G and the additional variant constraints, show how to construct a graph G' such that

(1) If F is a flow in G satisfying the additional constraints, there is a flow F' in G' of the same size,

(2) If F' is a flow in G', then there is a flow F in G satisfying the additional constraints with the same size.

Prove that properties (1) and (2) hold for your graph G'.

(a) **Max-Flow with Vertex Capacities:** In addition to edge capacities, every vertex $v \in G$ has a capacity c_v, and the flow must satisfy $\forall v : \sum_{u:(u,v)\in E} f_{uv} \leq c_v$.

(b) **Max-Flow with Multiple Sources:** There are multiple source nodes $s_1, \ldots, s_k$, and the goal is to maximize the total flow coming out of all of these sources.

(c) **Feasibility with Capacity Lower Bounds:** In addition to edge capacities, every edge (u, v) has a demand d_{uv}, and the flow along that edge must be at least d_{uv}. Instead of proving (1) and (2), design a graph G' and a number D such that if the maximum flow in G' is at least D, then there exists a flow in G satisfying $\forall (u, v) : d_{uv} \leq f_{uv} \leq c_{uv}$.

6. (??? **level)** **(Optional) Redemption for Homework 7**

Submit your *redemption file* for Homework 7 on Gradescope. If you looked at the solutions and took notes on what you learned or what you got wrong, include them in the redemption file.