CS 584/684 Algorithm Design and Analysis

Homework 7

Portland State U, Winter 2021 *03/02/21*
Lecturer: Fang Song *Due: 03/11/21*

Instructions. This problem set contains 8 pages (including this cover page) and 4 questions. A random subset of problems will be graded.

- Your solutions will be graded on *correctness* and *clarity*. You should only submit work that you believe to be correct, and you will get significantly more partial credit if you clearly identify the gap(s) in your solution. It is good practice to start any long solution with an informal (but accurate) summary that describes the main idea. You may opt for the "I take 15%" option.
- You need to submit a PDF file before the deadline. Either a clear scan of you handwriting or a typeset document is accepted. You will get 5 bonus points for typing in LaTeX (Download and use the accompany TeX file).
- You may collaborate with others on this problem set. However, you must **write up your own solutions** and **list your collaborators and any external sources** for each problem. Be ready to explain your solutions orally to a course staff if asked.
- For problems that require you to provide an algorithm, you must give a precise description of the algorithm, together with a proof of correctness and an analysis of its running time. You may use algorithms from class as subroutines. You may also use any facts that we proved in class or from the book.
- **If you describe a Greedy algorithm, you will get no credit without a formal proof of correctness, even if your algorithm is correct.**
- A proof of NP-completeness should include two parts: 1) the problem is in NP; and 2) the problem is NP-hard.

Exercises. Do not turn in.

1. A clique in an undirected graph $G = (V, E)$ is a subset $V' \subseteq V$ of vertices, each pair of which is connected by an edge in E. (In other words, a clique is a *complete* subgraph of G. The size of a clique is the number of vertices it contains. The Clique problems asks to decide whether a clique of a given size k exists in the graph. Show that Clique is NP-complete.

Problems to turn in.

1. (Demand) Suppose instead of capacities, we consider networks where each edge $u \to v$ has a non-negative *demand* $d(u \to v)$. Now an (s,t)-flow f is *feasible* if and only if $f(u \to v) \geq d(u \to v)$ for every edge $u \to v$. (Feasible flow values can now be arbitrarily large.) A natural problem in this setting is to find a feasible (s,t)-flow of *minimum* value.

   (a) (10 points) Describe an efficient algorithm to compute a feasible (s,t)-flow, given the graph, the demand function, and vertices s and t as input. (Hint: find a flow that is non-zero everywhere, and then scale it up to make it feasible.)

   (b) (10 points) Suppose you have access to a subroutine MaxFlow that computes *maximum* flows in networks with edge capacities. Describe an efficient algorithm to compute a *minimum* flow in a given network with edge demands; your algorithm should call MaxFlow exactly once.

   (c) (10 points) State and prove an analogue of the max-flow min-cut theorem for this setting. (Do minimum flows correspond to maximum cuts?)

2. (Integer linear programming) An *integer linear-programming* problem is a linear-programming problem with the additional constraint that the variables x must take on *integral* values. Specifically, given an integer $m \times n$ matrix A, an integer vector b of dimension m and integer vector c of dimension n, we want to maximize $c^T \cdot x$ under the constraints $Ax \leq b$ and $x \geq 0$.

   (a) (7 points) Show that *weak duality* (CLRS Lemma 29.8) holds for an integer linear program.

   (b) (8 points) Show that *strong duality* (CLRS Theorem 29.10) does not always hold for an integer linear program.

(c) (10 points (bonus)) Given a primal linear program in standard form, let us define P to be the optimal objective value for the primal linear program, D to be the optimal objective value for its dual, IP to be the optimal objective value for the integer version of the primal (that is, the primal with the added constraint that the variables take on integer values), and ID to be the optimal objective value for the integer version of the dual. Assuming that both the primal integer program and the dual integer program are feasible and bounded, show that

$$IP \leq P = D \leq ID\,.$$

(d) (10 points) Consider further *0-1 integer programming*, where one needs to decide if there exists an integer vector x of dimension n with elements in the set $\{0,1\}$such that $Ax \leq b$. Prove that 0-1 integer programming is NP-complete.

3. (Randomized and approximate algorithms)

   (a) (10 points) (Hat-check) Each of n customers gives a hat to a hat-check person at a restaurant. The hat-check person gives the hats back to the customers in a uniformly random order. What is the expected number of customers who get back their own hat?

(b) (10 points) (3-Coloring) Suppose you are given a graph $G = (V, E)$, and we want to color each node with one of three colors, even if we aren't necessarily able to give different colors to every pair of adjacent nodes. We say an edge (u, v) is satisfied if the colors assigned to u and v are different.

Consider a coloring scheme that *maximizes* the number of satisfied edges, and let c^* denote this number. Give a poly-time algorithm that produces a coloring that satisfies at least $\frac{2}{3}c^*$ edges. If you want to use an randomized algorithm, the *expected* number of edges it satisfies should be at least $\frac{2}{3}c^*$.

(c) (10 points) You have received a present containing n pieces of candy with weights $W[1, \ldots, n]$ (in ounces). You want to load the candy into as many boxes as possible, so that each box contains *at least* L ounces of candy. Describe an efficient 2-approximation algorithm for this problem. (Hint: First consider the case where every piece of candy weighs less than L ounces.)