CS 170 HW 10

Due on -

1 (Dynamic Programming) Greedy Cards

Ning and Evan are playing a game, where there are n cards in a line. The cards are all face-up (so they can both see all cards in the line) and numbered 2–9. Ning and Evan take turns. Whoever's turn it is can take one card from either the right end or the left end of the line. The goal for each player is to maximize the sum of the cards they've collected.

(a) Ning decides to use a greedy strategy: "on my turn, I will take the larger of the two cards available to me". Show a small counterexample ($n \leq 5$) where Ning will lose if he plays this greedy strategy, assuming Ning goes first and Evan plays optimally, but he could have won if he had played optimally.

(b) Evan decides to use dynamic programming to find an algorithm to maximize his score, assuming he is playing against Ning and Ning is using the greedy strategy from part (a). Help Evan develop the dynamic programming solution by providing an algorithm with its runtime and space complexity.

2 (Greedy Algorithms) Doctors and Patients

A doctor's office has n customers, labeled $1, 2, \ldots, n$, waiting to be seen. They are all present right now and will wait until the doctor can see them. The doctor can see one customer at a time, and we can predict exactly how much time each customer will need with the doctor: customer i will take $t(i)$ minutes.

(a) We want to minimize the average waiting time (the average of the amount of time each customer waits before they are seen, not counting the time they spend with the doctor). What order should we use? You do not need to justify your answer for this part. (Hint: sort the customers by ____)

(b) Let $x_1, x_2, \ldots, x_n$ denote an ordering of the customers (so we see customer x_1 first, then customer x_2, and so on). Prove that the following modification, if applied to any order, will never increase the average waiting time:

- If $i < j$ and $t(x_i) \geq t(x_j)$, swap customer i with customer j.

(For example, if the order of customers is $3, 1, 4, 2$ and $t(3) \geq t(4)$, then applying this rule with $i = 1$ and $j = 3$ gives us the new order $4, 1, 3, 2$.)

(c) Let u be the ordering of customers you selected in part (a), and x be any other ordering. Prove that the average waiting time of u is no larger than the average waiting time of x—and therefore your answer in part (a) is optimal.

Hint: Let i be the smallest index such that $u_i \neq x_i$. Use what you learned in part (b). Then, use proof by induction (maybe backwards, in the order $i = n, n-1, n-2, \ldots, 1$, or in some other way).

3 (Greedy) Tree Perfect Matching

A *perfect matching* in an undirected graph $G = (V, E)$ is a set of edges $E' \subseteq E$ such that for every vertex $v \in V$, there is exactly one edge in E' which is incident to v.

Give an algorithm which finds a perfect matching *in a tree*, or reports that no such matching exists. Describe your algorithm, prove that it is correct and analyse its running time.

4 (Linear Programming) Minimum Spanning Trees

Consider the minimum spanning tree problem, where we are given an undirected graph G 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:

$$\begin{aligned}
\text{Minimize} \quad & \sum_{(u,v)\in E} w_{u,v} x_{u,v} \\
\text{subject to} \quad & \sum_{\{u,v\}\in E: u\in S, v\in V\setminus S} x_{u,v} \geq 1 \quad \text{for all } S \subseteq V \text{ with } 0 < |S| < |V| \\
& \sum_{\{u,v\}\in E} x_{u,v} \leq |V| - 1 \\
& x_{u,v} \in \{0,1\}, \quad \forall (u,v) \in E
\end{aligned}$$

(a) Show how to obtain a minimum spanning tree T of G from an optimum solution of the ILP, and prove that T is indeed an MST. Why do we need the constraint $x_{u,v} \in \{0, 1\}$?

(b) How many constraints does the program have?

(c) Suppose that we *replaced* the binary constraint on each of the decision variables $x_{u,v}$ with the pair of constraints:

$$0 \leq x_{u,v} \leq 1, \quad \forall (u,v) \in E$$

How does this affect the optimum value of the program? Give an example of a graph where the optimum value of the relaxed linear program differs from the optimum value of the integer linear program.

5 (Max-Flow LP) Min Cost Flow

In the max flow problem, we just wanted to see how much flow we could send between a source and a sink. But in general, we would like to model the fact that shipping flow takes money. More precisely, we are given a directed graph G with source s, sink t, costs l_e, capacities c_e, and a flow value F. We want to find a nonnegative flow f with minimum cost, that is $\sum_e l_e f_e$, that respects the capacities and ships F units of flow from s to t.

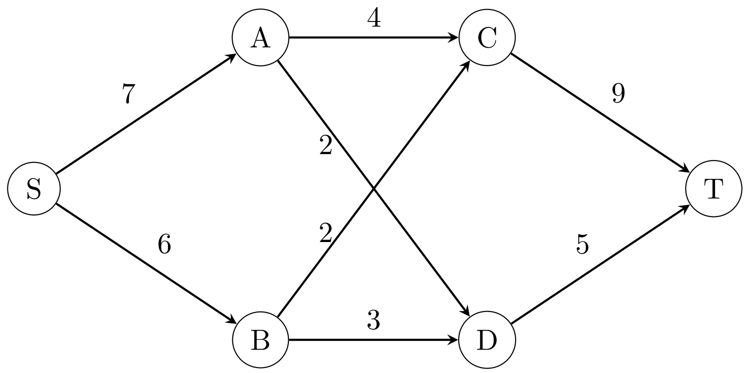
(a) Show that the minimum cost flow problem can be solved in polynomial time.

(b) Show that the shortest path problem can be solved using the minimum cost flow problem

(c) Show that the maximum flow problem can be solved using the minimum cost flow problem.

6 (Max Flows) Bottleneck Edges

Consider the following network (the numbers are edge capacities):



(a) Find the following:

- A maximum flow f, specified as a list of $s-t$ paths and the amount of flow being pushed through each.
- A minimum cut, specified as a list of edges that are part of the cut.

(b) Draw the residual graph G_f (along with its edge capacities). In this residual network, mark the vertices reachable from S and the vertices from which T is reachable.

(c) An edge of a network is called a *bottleneck edge* if increasing its capacity results in an increase in the maximum flow. List all bottleneck edges in the above network.

(d) Give a very simple example (containing at most four nodes) of a network which has no bottleneck edges.

(e) Give an efficient algorithm to identify all bottleneck edges in a network. (Hint: Start by running the usual network flow algorithm, and then examine the residual graph.)