CS 170 Homework 9 (Optional)

As this homework is optional, it will be worth no points. However, the topics covered in this homework will appear on midterm. So we highly recommend that you do it before checking the solution.

1 Baker

You are a baker who sells batches of brownies and cookies (unfortunately no brookies... for now). Each brownie batch takes 4 kilograms of chocolate and 2 eggs to make; each cookie batch takes 1 kilogram of chocolate and 3 eggs to make. You have 80 kilograms of chocolate and 90 eggs. You make a profit of 60 dollars per brownie batch you sell and 30 dollars per cookie batch you sell, and want to figure out how many batches of brownies and cookies to produce to maximize your profits.

(a) Formulate this problem as a linear programming problem. Draw the feasible region, and find the solution (state the cost function, linear constraints, and all vertices except for the origin).

(b) Suppose instead that the profit per brownie batch is C dollars and the profit per cookie batch remains at 30 dollars. For each vertex you listed in the previous part, give the range of C values for which that vertex is the optimal solution.

2 Dual of Maximum Independent Set

You are given a connected undirected graph $G = (V, E)$ where $|V| > 2$. Recall that a set of vertices $S \subseteq V$ is an *independent set* if there do not exist $u, v \in S$ such that there is an edge between u and v. In additon, an *edge cover* is a set of edges $C \subseteq E$ such that for each vertex v, there is some edge in C that it is incident to (so the edges in C 'cover' all the vertices).

(a) In the *maximum independent set* problem, you want to find an independent set of maximum size. Write the integer linear program (ILP) for this problem (that is, you may have the constraint $x \in \{0, 1\}$).

(b) Take your ILP, and replace the constraints of the form $x \in \{0, 1\}$ with $x \geq 0$ to get a linear program (LP). Then find the dual LP of this LP. What problem does the dual represent?

(c) True or false: For any connected graph, the optimum value for the (non-integer) primal-dual pair you constructed in part (b) are always equal. If true, prove. If false, give a counterexample.

(d) Take the ILP from part (a), and consider the ILP formed from the dual you found in part (b) by forcing all variables to be integers. True or false: The optimum values of these two ILPs are always equal.

3 Zero-Sum Games

Alice and Bob are playing a zero-sum game whose payoff matrix is shown below. The ij^{th} entry of the matrix shows the payoff that Alice receives if she plays strategy i and Bob plays strategy j. Alice is the row player and is trying to maximize her payoff, and Bob is the column player trying to minimize Alice's payoff.

Alice \Bob	1	2
1	4	1
2	2	5

Now we will write a linear program to find a strategy that maximizes Alice's payoff. Let the variables of the linear program be x_1, x_2 and p, where x_i is the probability that Alice plays row i and p denotes Alice's payoff.

(a) Write the linear program for choosing Alice's strategy to maximize her payoff.

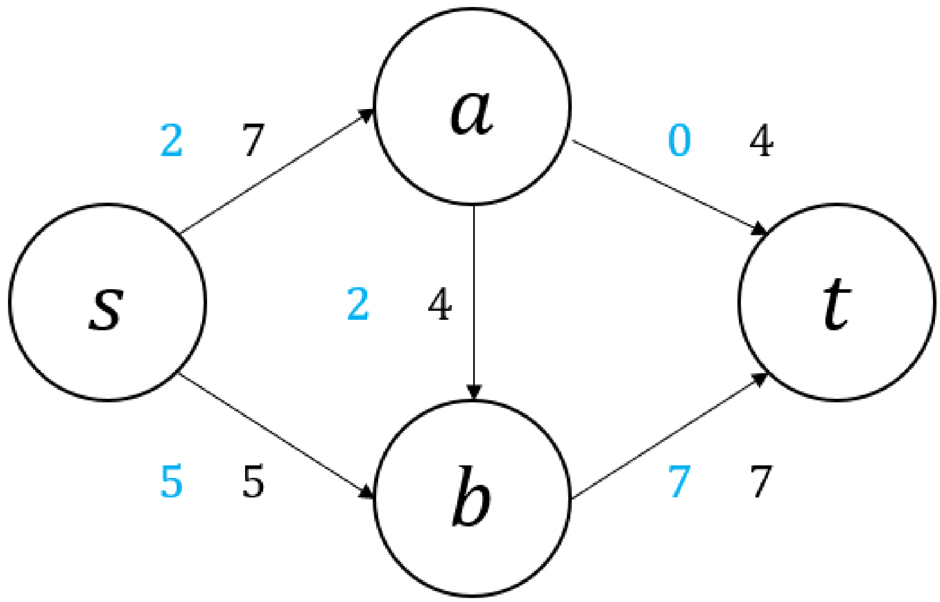
(b) Write a linear program from Bob's perspective trying to minimizing Alice's payoff. Let the variables of the linear program be y_1, y_2 and p, where y_i is the probability that Bob plays strategy i and p denotes Alice's payoff.

(c) As covered in lecture, Bob's linear program and Alice's are dual to each other. How can you see that this is the case for the LPs you have written here? Either take the dual mechanically or make a concise argument. (Hint: You may want to transform the linear programs into a form where it is easy to take the dual, if they are not already in such a form).

(d) What is the optimal solution and what is the value of the game?

4 Practice With Residual Graphs

(a) Consider the following network and flow on this network. An edge is labelled with its flow value (in blue) and capacity (in black). e.g. for the edge (s, a), we are currently pushing 2 units of flow on it, and it has capacity 7.



Draw the residual graph for this flow.

(b) We are given a network $G = (V, E)$ whose edges have integer capacities c_e, and a maximum flow f from node s to node t. Explicitly, f is given to us in the representation of integer flows along every edge e, (f_e).
However, we find out that one of the capacity values of G was wrong: for edge (u, v), we used c_{uv} whereas it really should have been $c_{uv} - 1$. This is unfortunate because the flow f uses that particular edge at full capacity: $f_{uv} = c_{uv}$. We could run Ford Fulkerson from scratch, but there's a faster way.

Describe an algorithm to fix the max-flow for this network in $O(|V| + |E|)$ time. Give a three-part solution.

5 Reductions Among Flows

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 directed graph G and the additional variant constraints, show how to construct a directed 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 $\forall v : \sum_{u:(u,v)\in E} f_{uv} \le c_v$.

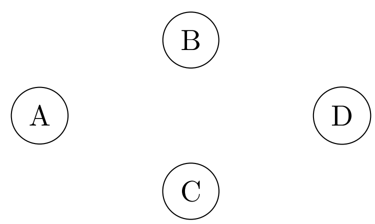
(b) **Max-Flow with Multiple Sources:** There are multiple source nodes $s_1, \ldots, s_k$, and the goal is to maximize the total flow coming out of all of these sources.

6 Flow Decomposition

Let $G(V, E)$ be a directed graph, and let f be any $s-t$ flow on this graph. Assume that there is no cycle in the graph where $f_e > 0$ for all edges in the cycle. Design an algorithm to decompose f into the sum of at most $|E|$ path flows. That is, your algorithm should find a set of $s-t$ paths $p_1 \dots p_k$ and corresponding flow values $F_1 \dots F_k$ such that:

- The number of paths k is at most $|E|$
- f is the sum of k path flows, where the ith flow sends F_i units of flow on path p_i. That is, for each edge e, if P_e is the set of paths in $p_1 \dots p_k$ that contain e, then $\sum_{p_i \in P_e} F_i = f_e$.

For example, in the below graph (where each edge e is labelled with f_e), one can decompose the flow into $p_1 = ((A, B), (B, C), (C, D)), p_2 = ((A, B), (B, D))$ where $F_1 = 1$, $F_2 = 2$.



Just the algorithm description and a brief explanation of why your algorithm finds at most $|E|$ paths are needed.

7 A Flowy Metric

Consider an undirected graph G with capacities $c_e \geq 0$ on all edges. G has the property that any cut in G has capacity at least 1. For example, a graph with a capacity of 1 on all edges is connected if and only if all cuts have capacity at least 1. However, c_e can be an arbitrary nonnegative number in general.

1. Show that for any two vertices $s, t \in G$, the max flow from s to t is at least 1.
2. Define the *length* of a flow f to be $\texttt{length}(f) = \sum_{e \in G} |f_e|$. Define the *flow distance* $d_{\texttt{flow}}(s, t)$ to be the minimum length of any $s-t$ flow f that sends one unit of flow from s to t and satisfies all capacities; i.e. $|f_e| \leq c_e$ for all edges e.

   Show that if $c_e = 1$ for all edges e in G, then $d_{\texttt{flow}}(s, t)$ is the length of the shortest path in G from s to t.

   (*Hint*: Let $d(s, t)$ be the length of the shortest path from s to t. A good place to start might be to first try to show $d_{\texttt{flow}}(s, t) \leq d(s, t)$. Then try to show $d_{\texttt{flow}}(s, t) \geq d(s, t)$)
3. **(Optional)** The shortest path satisfies the *triangle inequality*, that is for three vertices, s, t, and u in G, if $d(x, y)$ is the length of the shortest path from x to y, then $d(s, t) \leq d(s, u) + d(u, t)$. Show that the triangle inequality also holds for the flow distance. That is; show that for any three vertices $s, t, u \in G$

$$d_{\texttt{flow}}(s,t) \leq d_{\texttt{flow}}(s,u) + d_{\texttt{flow}}(u,t)$$

even when the capacities are arbitrary nonnegative numbers.

8 Running a Gym

You are running a gym and need to hire trainers for it. Here are the constraints:

- There are D days in total.
- There are n instructors for hire. The i^{th} instructor is available for hire from day a_i to day b_i. The numbers $\{a_i, b_i\}$ for $i = 1 \ldots n$ are given as input.
- Each instructor can be hired for their entire period of availability, i.e. if we hire the i^{th} instructor, then they would be at the gym every day from day a_i to day b_i.
- There should be **exactly** k instructors at the gym every day.

Your goal is to determine which instructors to hire so that the above constraints are met. Devise an efficient algorithm that solves the problem by constructing a directed graph G with a source s and sink t, and computing the maximum flow from s to t.