Note: Your TA probably will not cover all the problems. This is totally fine, the discussion worksheets are not designed to be finished in an hour. They are deliberately made long so they can serve as a resource you can use to practice, reinforce, and build upon concepts discussed in lecture, readings, and the homework.

Zero Sum Games: In this game, there are two players: a maximizer and a minimizer. We generally write the payoff matrix M in the perspective of the maximizer, so every row corresponds to an action that the maximizer can take, every column corresponds to an action that the minimizer can take, and a positive entry corresponds to the maximizer winning. M is a n by m matrix, where n is the number of choices the maximizer has, and m is the number of choices the minimizer has.

$$M = \begin{bmatrix} M_{1,1} & M_{1,2} & \cdots & M_{1,m} \\ M_{2,1} & M_{2,2} & \cdots & M_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ M_{n,1} & M_{n,2} & \cdots & M_{n,m} \end{bmatrix}$$

A linear program that represents fixing the maximizer's choices to a probabilistic distribution where the maximizer has n choices, and the probability that the maximizer chooses choice i is p_i is the following:

$$\begin{gathered} \max \ z \\ M_{1,1} \cdot p_1 + \cdots + M_{n,1} \cdot p_n \geq z \\ M_{1,2} \cdot p_1 + \cdots + M_{n,2} \cdot p_n \geq z \\ \vdots \\ M_{1,m} \cdot p_1 + \cdots + M_{n,m} \cdot p_n \geq z \\ p_1 + p_2 + \cdots + p_n = 1 \\ p_1, p_2, \ldots, p_n \geq 0 \end{gathered}$$

or in other words,

$$\max z \text{ s.t. } M^\top p \geq z\mathbf{1},\ \mathbf{1}^\top p = 1,\ p \geq 0$$

where $p = (p_1, \ldots, p_n)$.

The dual represents fixing the minimizer's choices to a probabilistic distribution. If we let the probability that the minimizer chooses choice j be q_j, then the dual is the following:

$$\begin{gathered} \min \ w \\ M_{1,1} \cdot q_1 + \cdots + M_{1,m} \cdot q_m \leq w \\ M_{2,1} \cdot q_1 + \cdots + M_{2,m} \cdot q_m \leq w \\ \vdots \\ M_{n,1} \cdot q_1 + \cdots + M_{n,m} \cdot q_m \leq w \\ q_1 + q_2 + \cdots + q_m = 1 \\ q_1, q_2, \ldots, q_m \geq 0 \end{gathered}$$

or in other words,

$$\min w \text{ s.t. } Mq \leq w\mathbf{1},\ \mathbf{1}^\top q = 1,\ q \geq 0$$

where $q = (q_1, \ldots, q_m)$.

By strong duality, the optimal value of the game is the same regardless of whether you fix the minimizer's distribution first or the maximizer's distribution first; i.e. $z^* = w^*$, or $z^* + (-w^*) = 0$.

1 Zero-Sum Games Short Answer

(a) Suppose a zero-sum game has the following property: The payoff matrix M satisfies $M = -M^\top$. What is the expected payoff of the row player?

Hint: try rewriting the minimizer's (i.e. column player's) LP as a maximization problem. Then, use the definition of a ZSG (i.e. when both players try to maximize their payoff, their optimal payoffs sum to 0).

(b) True or False: If every entry in the payoff matrix is either 1 or -1 and the maximum number of 1s in any row is k, then for any row with less than k 1s, the row player's optimal strategy chooses this row with probability 0. Justify your answer.

(c) True or False: Let M_i denote the ith row of the payoff matrix. If $M_1 = \frac{M_2 + M_3}{2}$, then there is an optimal strategy for the row player that chooses row 1 with probability 0. Justify your answer.

Solution:

(a) To get the column player's payoff matrix, we negate the payoff matrix and take its transpose. So we get that the row and column players' payoff matrices are the same matrix. In turn, they must have the same expected payoff, but also the sum of their expected payoffs must be 0, so both players must have expected payoff 0.

(b) False: Consider the 2-by-3 payoff matrix:

$$\begin{bmatrix} 1 & -1 & 1 \\ -1 & 1 & -1 \end{bmatrix}$$

The row player's optimal strategy is to choose the two rows with equal probability - note that the column player doesn't care about choosing column 1 vs column 3, so this game is no different than the zero-sum game for the 2-by-2 payoff matrix:

$$\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}$$

(c) True: Consider the optimal strategy for the row player. If the row player chooses rows 1, 2, 3 with probabilities p_1, p_2, p_3, they can instead choose row 1 with probability 0, row 2 with probability $p_2 + p_1/2$, and row 3 with probability $p_3 + p_1/2$. The expected payoff of this strategy is the same, so this strategy is also optimal.

2 Domination

In this problem, we explore a concept called *dominated strategies*. Consider a zero-sum game with the following payoff matrix for the row player:

		Column: A	B	C
Row:	D	1	2	-3
	E	3	2	-2
	F	-1	-2	2

Note: All parts of this problem can be solved without using an LP solver or solving a system of linear equations.

(a) If the row player plays optimally, can you find the probability that they pick D without directly solving for the optimal strategy? Justify your answer.

Hint: How do the payoffs for the row player picking D compare to their payoffs for picking E?

(b) Given the answer to part a, if the both players play optimally, what is the probability that the column player picks A? Justify your answer.

(c) Given the answers to part a and b, what are both players' optimal strategies?

Solution:

(a) 0. Regardless of what option the column player chooses, the row player always gets a higher payoff picking E than D, so any strategy that involves a non-zero probability of picking D can be improved by instead picking E.

(b) 0. We know that the row player is never going to pick D, i.e. will always pick either E or F. But in this case, picking B is always better for the column player than picking A (A is only

better if the row player picks D). That is, conditioned on the row player playing optimally, B dominates A.

(c) Based on the previous two parts, we only have to consider the probabilities the row player picks E or F and the column player picks B or C. Looking at the 2-by-2 submatrix corresponding to these options, it follows that the optimal strategy for the row player is to pick E and F with probability $1/2$, and similarly the column player should pick B, C with probability $1/2$.

Flow. The *capacity* indicates how much flow can be allowed on an edge. Given a directed graph $G = (V, E)$ with edge capacites $c(u, v)$ (for all $(u, v) \in E$) and vertices $s, t \in V$, a flow is a mapping $f : E \to \mathbb{R}^+$ that satisfies

- Capacity constraint: $f(u, v) \leq c(u, v)$, the flow on an edge cannot exceed its capacity.
- Conservation of flows: $f^{\text{in}}(v) = f^{\text{out}}(v)$, flow in equals flow out for any $v \notin \{s, t\}$

Here, we define $f^{\text{in}}(v) = \sum_{u:(u,v)\in E} f(u, v)$ and $f^{\text{out}}(v) = \sum_{u:(v,u)\in E} f(u, v)$. We also define $f(v, u) = -f(u, v)$, and this is called *skew-symmetry*. Note that the total flow in the graph is $\sum_{v:(s,v)\in E} f(s, v) = \sum_{u:(u,t)\in E} f(u, t)$, where s is the source node of the graph and t is the target node.

Residual Graph. Given a flow network (G, s, t, c) and a flow f, the *residual capacity* (w.r.t. flow f) is denoted by $c_f(u, v) = c_{uv} - f_{uv}$. And the *residual network* $G_f = (V, E_f)$ where $E_f = \{(u, v) : c_f(u, v) > 0\}$.

Max Flow. Given a directed graph $G = (V, E)$ with edge capacities $c(u, v)$ for all $(u, v) \in E$ and vertices $s, t \in V$, the goal is to compute the maximum flow that can go from s to t. This can be solved with either Ford-Fulkerson (or its optimized variant Edmonds-Karp).

Ford-Fulkerson. Keep pushing along $s-t$ paths in the residual graph and update the residual graph accordingly. The runtime of this algorithm is $O(mF)$, where $m = |E|$ and F is the value of the max flow.

Edmonds-Karp. We can implement Ford-Fulkerson using BFS to find the $s - t$ paths. This yields a faster runtime of $O(nm^2)$, where $n = |V|, m = |E|$.

Min-Cut. Given a directed graph $G = (V, E)$ with edge weights $w(u, v)$ for all $(u, v) \in E$ and vertices $s, t \in V$, the goal is to compute the lightest $s - t$ cut (i.e. the cut separating t from s with the smallest sum of edge weights crossing it). Note that the Min Cut problem is the dual of the Max Flow problem.

3 Max-Flow Min Cut Basics

For each of the following, state whether the statement is True or False. If true provide a short proof, if false give a counterexample.

(a) If all edge capacities are distinct, the max flow is unique.

(b) If all edge capacities are distinct, the min cut is unique.

(c) If all edge capacities are increased by an additive constant, the min cut remains unchanged.

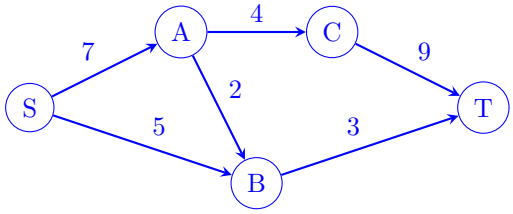
(d) If all edge capacities are multiplied by a positive integer, the min cut remains unchanged.

(e) In any max flow, there is no directed cycle on which every edge carries positive flow.

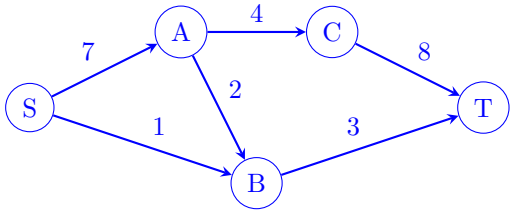
(f) There exists a max flow such that there is no directed cycle on which every edge carries positive flow.

Solution:

(a) False. Consider the following graph:



(b) False. Consider the graph below. The cut SA and the cut SAB are both size 7.

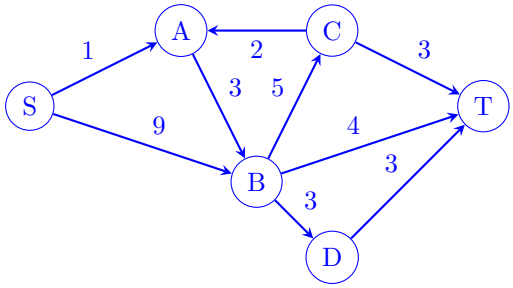


(c) False. Add one to all of the edge capacities of the graph in part (b). The cut SA and the SAB have different values now.

(d) True. Let the value of a cut be $\sum_e c_e$ and the value of the minimum cut be $\sum_{e'} c_{e'}$. The minimum cut must still be the minimum cut after multiplying the edges by a positive constant, due to the distributive property:

$$a \sum_{e'} c_{e'} - a \sum_{e} c_e = a(\sum_{e'} c_{e'} \sum_{e} c_e)$$

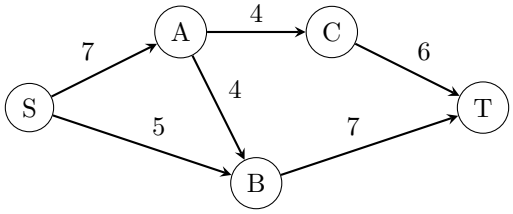
(e) False. Consider the graph below:



(f) True. See any graphs other than (e). Consider the graph in part (e) without the edge SA.

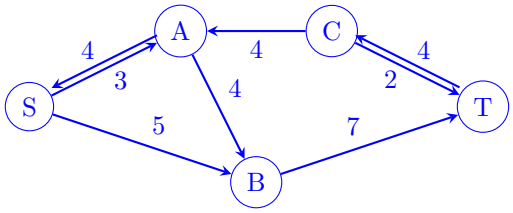
4 Residual in graphs

Consider the following graph with edge capacities as shown:



(a) Consider pushing 4 units of flow through $S \to A \to C \to T$. Draw the residual graph after this push.

Solution:

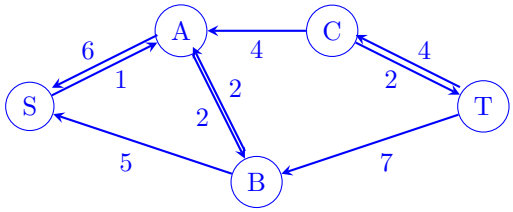


(b) Compute a maximum flow of the above graph. Find a minimum cut. Draw the residual graph of the maximum flow.

Solution: A maximum flow of value 11 results from pushing:

- 4 units of flow through $S \to A \to C \to T$;
- 5 units of flow through $S \to B \to T$; and
- 2 units of flow through $S \to A \to B \to T$.

(There are other maximum flows of the same value, can you find them?) The resulting residual graph (with respect to the maximum flow above) is:

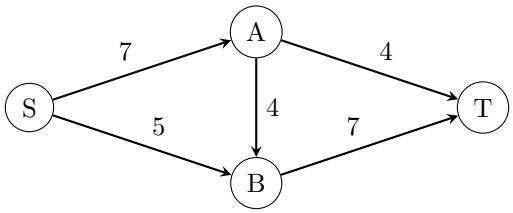


A minimum cut of value 11 is between $\{S, A, B\}$ and $\{C, T\}$ (with cross edges $A \to C$ and $B \to T$).

5 Max Flow, Min Cut, and Duality

In this exercise, we will demonstrate that LP duality can be used to show the max-flow min-cut theorem.

Consider this graph instance of max flow:



Let f_1 be the flow pushed on the path $\{S, A, T\}$, f_2 be the flow pushed on the path $\{S, A, B, T\}$, and f_3 be the flow pushed on the path $\{S, B, T\}$. The following is an LP for max flow in terms of the variables f_1, f_2, f_3:

$$
\begin{aligned}
\max \quad & f_1 + f_2 + f_3 && \\
& f_1 + f_2 \leq 7 && \text{(Constraint for } (S, A)) \\
& f_3 \leq 5 && \text{(Constraint for } (S, B)) \\
& f_1 \leq 4 && \text{(Constraint for } (A, T)) \\
& f_2 \leq 4 && \text{(Constraint for } (A, B)) \\
& f_2 + f_3 \leq 7 && \text{(Constraint for } (B, T)) \\
& f_1, f_2, f_3 \geq 0 &&
\end{aligned}
$$

The objective is to maximize the flow being pushed, with the constraint that for every edge, we can't push more flow through that edge than its capacity allows.

(a) Find the dual of this linear program, where the variables in the dual are x_e for every edge e in the graph.

(b) Consider any cut in the graph. Show that setting $x_e = 1$ for every edge crossing this cut and $x_e = 0$ for every edge not crossing this cut gives a feasible solution to the dual program.

(c) Based on your answer to the previous part, what problem is being modelled by the dual program? By LP duality, what can you argue about this problem and the max flow problem?

Solution:

(a) The dual is:

$$\begin{aligned} \min \quad & 7x_{SA} + 5x_{SB} + 4x_{AT} + 4x_{AB} + 7x_{BT} \\ x_{SA} + x_{AT} &\geq 1 \qquad \text{(Constraint for } f_1\text{)} \\ x_{SA} + x_{AB} + x_{BT} &\geq 1 \qquad \text{(Constraint for } f_2\text{)} \\ x_{SB} + x_{BT} &\geq 1 \qquad \text{(Constraint for } f_3\text{)} \\ x_e &\geq 0 \quad \forall e \in E \end{aligned}$$

(b) Notice that each constraint contains all variables x_e for every edge e in the corresponding path. For any $s-t$ cut, every $s-t$ path contains an edge crossing this cut. So for any cut, the suggested solution will set at least one x_e to 1 on each path, giving that each constraint is satisfied.

(c) The dual LP is an LP for the min-cut problem. By the previous answer, we know the constraints describe solutions corresponding to cuts. The objective then just says to find the cut of the smallest size. By LP duality, the dual and primal optima are equal, i.e. the max flow and min cut values are equal.

6 Flow vs LP

You play a middleman in a market of m suppliers and n purchasers. The i-th supplier can supply up to $s[i]$ products, and the j-th purchaser would like to buy up to $b[j]$ products.

However, due to legislation, supplier i can only sell to a purchaser j if they are situated at most 1000 miles apart. Assume that you're given a list L of all the pairs (i, j) such that supplier i is within 1000 miles of purchaser j. Given m, n, $s[1..m]$, $b[1..n]$, and L as input, your job is to compute the maximum number of products that can be sold. The runtime of your algorithm must be polynomial in m and n.

Show how to solve this problem using a network flow algorithm as a subroutine. Describe the graph and explain why the output from running the network flow algorithm on your graph gives a valid solution to this problem.

Solution:

(a) *Algorithm:* We create a bipartite graph with $m + n + 2$ nodes. Label two of the nodes as a "source" and a "sink." Label m nodes as suppliers, and n nodes as purchasers. Now, we will create the following edges:

- Create an edge from the source to supplier i with capacity $s[i]$.

- For each pair (i, j) in L, create an edge from supplier i to purchaser j with infinite capacity.
- Create an edge from purchaser j to the sink with capacity $b[j]$. We then plug this graph into our network flow solver, and take the size of the max flow as the number of dollars we can make.

Proof of correctness: We claim that the value of the max flow is precisely the maximum amount transactions we can make. To show this, we can show that a strategy of selling products corresponds exactly to a flow in this graph and vice versa.

For any flow, let $x_{i,j}$ be the amount of flow that goes from the node of supplier i to the node of purchaser j. Then, we claim a product selling strategy will sell exactly $x_{i,j}$ products from supplier i to purchaser j. This is a feasible strategy, since the flow going out of the node of supplier i is already bounded by $s[i]$ by the capacity of the edge from the source to that node, and similarly for the purchasers. Similarly, for the other direction, one can observe that any feasible selling strategy leads to a feasible flow of the same value.

(b) We define a variable $x_{i,j}$, denoting the amount of products we take from supplier i and sell to purchaser j. Then, we have the following linear program

$$\begin{aligned}
\max \sum_{i=1}^{m} \sum_{j=1}^{n} x_{i,j} \\
\sum_{i=1}^{m} x_{i,j'} \leq b[j'], \text{ for all } j' \in [1, n] \\
\sum_{j=1}^{n} x_{i',j} \leq s[i'], \text{ for all } i' \in [1, m] \\
x_{i,j} = 0, \text{ for all } (i, j) \notin L \\
x_{i,j} \geq 0, \text{ for all } (i, j)
\end{aligned}$$

The linear program has mn variables and $O(mn)$ linear inequalities, so it can be solved in time polynomial in m and n.

(c) Network flow is better. Linear programming is not guaranteed to find an integer solution (not even if one exists), so the approach in part (b) might yield a solution that would involve selling fractional products. In contrast, since all the edge capacities in our graph in part (a) are integers, the Ford-Fulkerson algorithm for max flow will find an integer solution. Thus, max flow is the better choice, because there are algorithms for that formulation that will let us find an integer solution.

