15-451 Algorithms, Spring 2018
Recitation #7 Worksheet

Rock-Paper-Scissors with a Twist Suppose we have a non-standard game of rock-paper-scissors, which is still zero-sum, but with the following payoffs for the row player (Alice):

Alice plays \ Bob plays	r	p	s
r	0	-1	2
p	1	0.5	-1
s	-1	2	-1

1. If Alice decides to play $\mathbf{p} = (p_1, p_2, 1 - p_1 - p_2)$ as her strategy, what should Bob play to minimize the payoff to Alice? What is Alice's payoff if he does this.

   Solution: Bob can either play rock, or paper, or scissors. He'll play whichever gives the lowest payoff to Alice. All other mixed strategies are averages of these three pure strategies. In this case Alice's payoff will be

$$\min\Big\{p_1 \cdot 0 + \quad p_2 \cdot 1 + \quad (1-p_1-p_2)\cdot(-1),$$
$$p_1 \cdot (-1) + \quad p_2 \cdot (0.5) + \quad (1-p_1-p_2)\cdot(2),$$
$$p_1 \cdot (2) + \quad p_2 \cdot (-1) + \quad (1-p_1-p_2)\cdot(-1)\Big\}$$

2. Hence, write down the linear program that Alice must solve in order to find her best strategy $\mathbf{p}^\star$. You don't have to solve for the value of this game.

   Solution: Alice will try to maximize her payoff, given Bob plays the strategy in part B. So she will try to solve

$$\begin{aligned} & \max v \\ \text{subject to} \quad & p_1 \cdot 0 + p_2 \cdot 1 + (1-p_1-p_2)\cdot(-1) \geq v \\ & p_1 \cdot (-1) + p_2 \cdot (0.5) + (1-p_1-p_2)\cdot(2) \geq v \\ & p_1 \cdot (2) + p_2 \cdot (-1) + (1-p_1-p_2)\cdot(-1) \geq v \\ & p_1 + p_2 \leq 1 \\ & p_1 \geq 0, p_2 \geq 0 \end{aligned}$$

   Note that taking the new variable v and making all the three expressions at least as large as v, and then maximizing v makes sure that v is the minimum of the three expressions as desired in the previous part.

Coding up shortest paths as an LP: You can code up the s-t shortest-path problem as an LP. The input is a directed graph G with edge weights $w(e) \geq 0$, start node s, and a target t. We want to find a path from s to t of least weight.

1. *What are the variables?* Suppose we have a variable d_v for every vertex v, representing its distance from s. What are the constraints you should write?

   Solution: For any edge $e = (u, v)$ we put the constraint that $d_v \leq d_u + w(e)$. In other words, since one way to reach v is to reach u first and then go on edge e to v, the shortest-path distances must satisfy this property. We also add the constraint $d_s = 0$.

2. The objective? We claim it is *maximize* d_t. Why does this make sense?

   Solution: Any solution will have values d_v that are *no larger* than the shortest path from s to v, since by induction (on nodes in order of their true distance from s) the node u that comes right before v in the true shortest path from s will never have too large a value, and we have constrained $d_v \leq d_u + w(u, v)$. So d_t cannot be more than the shortest s-t distance.

   Also, if you set d_v to be the actual shortest-path distance from s to v, then this satisfies all the constraints in the LP. So the optimal value of d_t cannot be less than the shortest s-t distance either. Hence it must be equal.

   Another way to think about this: take a set of stones (one for each node) and strings (one for each edge), and tie a string of length $w(u, v)$ between stones u and v. Now pull s and t as far as possible from each other: the strings that become taut are those on the shortest path from s to t, and the furthest s and t can be pulled from each other equals their shortest-path distance.

Coding up shortest paths as an LP (Approach II): Have a variable x_e for each edge e, with constraints that $0 \leq x_e \leq 1$. (Think of $x_e = 1$ meaning we use that edge, and $x_e = 0$ meaning we don't, but of course the LP might assign fractional values.) Our goal is to minimize $\sum_e w(e)x_e$, subject to:

- One unit of "flow" leaves s: $\sum_{e=(s,v)} x_{sv} = 1$
- One unit of "flow" enters t: $\sum_{e=(v,t)} x_{vt} = 1$.
- For all $v \notin \{s, t\}$, we have flow-in = flow-out: $\sum_{e=(u,v)} x_{uv} = \sum_{e=(v,u)} x_{vu}$.

This is a min-cost flow (send 1 unit of s-t-flow with least cost). You can put edge-capacity 1, but since we send only 1 unit of flow, the capacity constraints don't matter.

1. Solve the LP to get an optimal solution. If we get back an integer solution (i.e., if $x_e \in \{0, 1\}$ for each edge e) argue that the edges with $x_e = 1$ give a shortest s-t path.

   Solution: The flow gives a path. And if there is a shorter path, that would be a solution with smaller cost to the LP as well, contradicting the optimality of x.

2. Suppose the optimal LP solution returns a fractional flow. Argue that any flow-carrying path from s to t is a shortest path.

   Solution: Take the flow f and decompose it into flows on s-t paths, say α_i flow on path P_i, with $\sum_i \alpha_i = 1$. The LP objective function is $\sum_i \alpha w(P_i)$, where $w(P_i) = \sum_{e \in P_i} w_e$. Now if some P_i with $\alpha_i > 0$ was not a shortest path, we could reduce flow on that and put more flow on other paths to get a smaller cost LP solution.