CS170–Spring 2019 — Homework 11 Solutions

Ran Liao, SID 3034504227

April 9, 2019

Collaborators:Jingyi Xu, Renee Pu

1 Study Group

Name	SID
Ran Liao	3034504227
Jingyi Xu	3032003885
Renee Pu	3032083302

2 Zero-Sum Battle

(a)

$$\max p$$

$$p \leq -10x_1 + 4x_2 + 6x_3 \text{ (payoff when trainer B chooses the ice Pokemon)}$$

$$p \leq 3x_1 - 1x_2 - 9x_3 \text{ (payoff when trainer B chooses the water Pokemon)}$$

$$p \leq 3x_1 - 3x_2 + 2x_3 \text{ (payoff when trainer B chooses the fire Pokemon)}$$

$$x_1 + x_2 + x_3 = 1$$

$$x_1 \geq 0$$

$$x_2 \geq 0$$

$$x_3 \geq 0$$

The optimal strategy is $(0.335, 0.563, 0.102)$ and the payoff is -0.48.

(b)

$$\min p$$

$$p \geq -10y_1 + 3y_2 + 3y_3 \text{ (payoff when trainer A chooses the dragon Pokemon)}$$

$$p \geq 4y_1 - 1y_2 - 3y_3 \text{ (payoff when trainer A chooses the steel Pokemon)}$$

$$p \geq 6y_1 - 9y_2 + 2y_3 \text{ (payoff when trainer A chooses the rock Pokemon)}$$

$$y_1 + y_2 + y_3 = 1$$

$$y_1 \geq 0$$

$$y_2 \geq 0$$

$$y_3 \geq 0$$

The optimal strategy is $(0.268, 0.323, 0.409)$ and the payoff is -0.48.

3 Zero Sum Games

(a) x_1 is the probability that Alice will play strategy 1.
x_2 is the probability that Alice will play strategy 2.
p is Alice's payoff

(b)

$$\max p$$
$$p \le 4x_1 + 2x_2 \text{ (payoff when Bob take strategy 1)}$$
$$p \le x_1 + 5x_2 \text{ (payoff when Bob take strategy 2)}$$
$$x_1 + x_2 = 1$$
$$x_1 \ge 0$$
$$x_2 \ge 0$$
$$p \ge 1$$

(c)

$$\max p$$
$$p \le 4x_1 + 2(1 - x_1) = 2x_1 + 2$$
$$p \le x_1 + 5(1 - x_1) = -4x_1 + 5$$
$$0 \le x_1 \le 1$$
$$p \ge 1$$

(d)

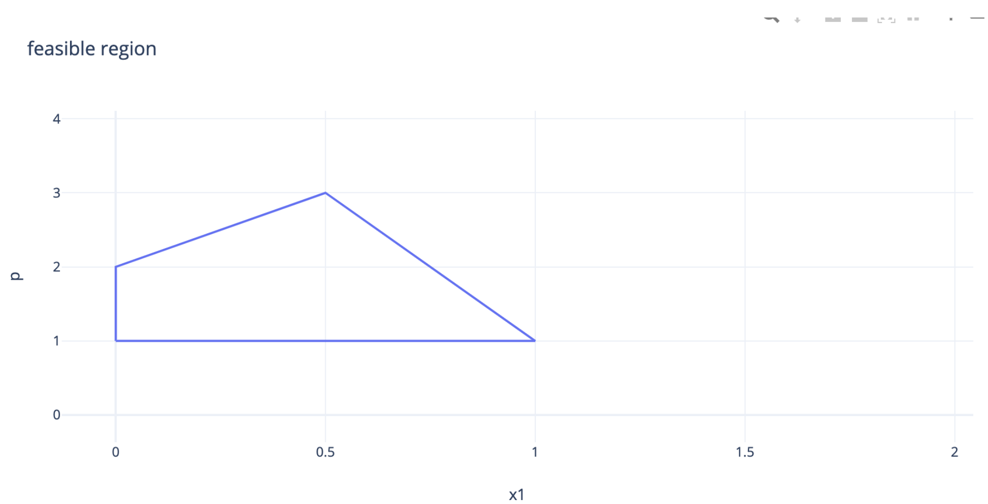

Figure 1: Feasible Region

(e) The optimal solution is $(\frac{1}{2}, \frac{1}{2})$ and the value of game is 3.

4 Repairing a Flow

(a) **Main Idea**

Build the residual network based on maximum flow f. Then find a path from t to s that go through edge (v, u) in residual network. Push 1 unit flow into this path and fix the wrong capacity from c_{uv} to $c_{uv} - 1$. Then run the original max-flow algorithm starting from this residual network.

(b) **Proof of Correctness**

Pushing 1 unit back from t to s that go through edge (v, u) will make the flow go through edge (u, v) be valid. Then run the original max-flow algorithm again from this is sure will give an optimal solution.

(c) **Runtime Analysis**

Build residual network will cost $O(|E|)$ time. Find a path from t to s that go through edge (v, u) by DFS or BFS will cost $O(|V| + |E|)$ time. Then push 1 unit into this path will cost at most $|E|$ time. Since the max-flow solution after repairing capacity of edge (u, v) cannot be larger than the original one. At most 1 iteration is needed if run max-flow algorithm starting from this. So total runtime is still linear, which is $O(|V| + |E|)$.

5 Generalized Max Flow

(a) **Constraints**

(i) **Capacity Constraints**

$$\sum_{i=1}^{k} f_e^{(i)} \leq c_e \quad \text{for } \forall e$$

(ii) **Flow Conservation**

$$\sum_{i=1}^{k} \sum_{u:(u,v)\in E} f_{(u,v)}^{(i)} = \sum_{i=1}^{k} \sum_{u:(v,w)\in E} f_{(v,w)}^{(i)} \quad \text{for } \forall v \text{ except } s1, \cdots, s_k, t_1, \cdots, t_k$$

(iii) **Nonnegativity**

$$f_e^{(i)} \geq 0 \quad \text{for } \forall e, i = 1, \cdots, k$$

(iv) **Demand Constraints**

$$\sum_{u:(u,t_i)\in E} f_{(u,t_i)}^{(i)} \geq d_i \quad \text{for } i = 1, \cdots, k$$

(b) **Objective**

$$\max \sum_{i=1}^{k} \sum_{u:(u,t_i)\in E} f_{(u,t_i)}^{(i)}$$

The sum of all flows go to destination.

6 Reductions Among Flows

(a) Separate each vertex into two new vertices and denote them as v_{in} and v_{out}. Then link all incoming edges in original graph G to v_{in} and all outcoming edges to v_{out}. More formally, create a new edge (u, v_{in}) in new graph if edge (u, v) is in original graph and create a new edge (v_{out}, u) in new graph if edge (v, u) is in original graph. Finally, create a new edge (v_{in}, v_{out}) with capacity c_v.

Proof:

If F is a flow in G satisfying the vertex capacity constraint, set $f_{(v_{in},v_{out})}$ to be $\sum_{u:(u,v)\in E} f_{uv}$ and let all other flows remain same. This is a valid flow because $f_{(v_{in},v_{out})} = \sum_{u:(u,v)\in E} f_{uv} \leq c_v$. And since all other flows are unchanged, it's flow with same size.

If $F\prime$ is a flow in $G\prime$, ignore all $f_{(v_{in},v_{out})}$ will give a valid flow F in G with same size.

(b) Create an artificial vertex s and create edges $(s, s_1), \cdots, (s, s_k)$ with ∞ capacity.

Proof:

If F is a flow in G satisfying the vertex capacity constraint, there's always a possible solution in $G\prime$ with same size. Since the capacity for edges $(s, s_1), \cdots, (s, s_k)$ is infinity. It can be arbitrary large. So just assign $f_{(s,s_i)}$ with $\sum_{w:(s_i,w)\in E} f_{(s_i,w)}$.

If $F\prime$ is a flow in $G\prime$, ignore all $f_{(s,s_i)}$ will give a valid flow F in G with same size.

7 A Flowy Metric

(a) According to **Max-flow min-cut theorem**, the size of the maximum flow in a network equals the capacity of the smallest (s, t)-cut. Since any cut in G has capacity at least 1, the max flow from s to t is at least 1.

(b) Suppose a flow f send 1 unit along the shortest path from s to t, the length of f is $1 \times d(s,t) = d(s,t)$. Therefore, $d_{flow}(s,t)$ is smaller than or equal to $d(s,t)$ by definition, i.e. $d_{flow}(s,t) \leq d(s,t)$.

In general, a flow can send a fraction of unit along different path from s to t. Suppose flow f send $f_1, f_2, \cdots f_n$ units along different path with length $d_1, d_2, \cdots d_n$. And $\sum_{i=1}^{n} f_i = 1$.

$$d_{flow}(s,t) = \sum_{i=1}^{n} d_i f_i \geq \sum_{i=1}^{n} d(s,t) f_i = d(s,t) \sum_{i=1}^{n} f_i = d(s,t)$$

Since $d_{flow}(s,t) \leq d(s,t)$ and $d_{flow}(s,t) \geq d(s,t)$, d_{flow} must equal to $d(s,t)$.