CMSC 27200 - Spring 2024 — Lorenzo Orecchia
due 5/10/24 via Gradescope

H8.Mark Xiong

- The assignment is due at Gradescope on 5/10/24.
- A LaTeX template will be provided for each homework. You are strongly encouraged to type your homework into this template using LaTeX. If you are writing by hand, please fill in the solutions in this template, inserting additional sheets as necessary. This will help facilitate the grading.
- You are permitted to discuss the problems with up to 2 other students in the class (per problem); however, you must write up your own solutions, in your own words. Do not submit anything you cannot explain. If you do collaborate with any of the other students on any problem, please list all your collaborators in the appropriate spaces.
- Similarly, please list any other source you have used for each problem, including other textbooks or websites.
- Show your work. Answers without justification will be given little credit.
- Your homework is resubmittable. Please refer to the course syllabus on Canvas for a more detailed description of this. For any problem that you have not changed from your last submission, please make sure to indicate this in your submission to help our graders grade faster.

Problem 1 (Knapsack and Linear Programs) Recall the 0-1 Knapsack problem discussed in class: we are given a set of items with values and weights: $\{(v_i, w_i)\}_{i=1}^n$ and a weight budget W, and we are interested in finding the choice of items (i.e. the subset S of $[n]$) maximizing the sum of the values of the items:

$$\sum_{i \in S} v_i,$$

subject to the weight constraints:

$$\sum_{i \in S} w_i < W.$$

Consider the following linear program:

$$\begin{aligned} \max_{x \in \mathbb{R}^n} \quad & \sum_{i \in [n]} x_i v_i \\ & \sum_{i \in [n]} x_i w_i \leq W \\ & 0 \leq x_i \leq 1 \qquad \forall i \in [n] \end{aligned} \tag{1}$$

Here, each variable x_i is meant to indicate whether or not we are including a specific item in our solution.

(a) Explain why the above linear program (1) doesn't solve the original 0-1 Knapsack problem, and provide an example instance of the 0-1 Knapsack problem that this LP gives an invalid solution for.

(b) Describe the problem, in words, that is solved by an instance of (1).

(c) Describe (in words) and give the pseudocode of an algorithm that finds the optimal assignment of x in (1).

(d) Compute the dual of (1).

Collaborators:

Solution: The solution is as follows:

(a) The above linear program does not solve the original 0-1 Knapsack problem because the variables x_i are not restricted to be binary. This means that the LP can select a fractional amount of each item, which is not allowed in the 0-1 Knapsack problem. For example, consider the following instance of the 0-1 Knapsack problem: $n = 2$, $W = 1$, $v_1 = 1$, $v_2 = 1$, $w_1 = 1$, $w_2 = 1$. The optimal solution to the 0-1 Knapsack problem is to select either item 1 or item 2, but not both. However, the LP can select $x_1 = 0.5$ and $x_2 = 0.5$, which is not a valid solution to the 0-1 Knapsack problem.

(b) The problem solved by an instance of (1) is to find the optimal fractional assignment of items to maximize the total value of the items while satisfying the weight constraint.

(c) The algorithm to solve the fractional knapsack problem is a greedy algorithm that involves sorting items by they value/weight ratio in descending order and then add them to the knapsack until the limit is reached. The pseudocode is as follows:

Algorithm 1 Fractional Knapsack

Input: Items with values and weights $\{(v_i, w_i)\}_{i=1}^n$ and a weight budget $\underline{W}$
Output: The total value of the items in the knapsack

```
Sort items by value/weight ratio in descending order
Initialize total value V = 0 and total weight W = 0
for each item i do
    if W + w_i ≤ W̲ then
        Add item i to the knapsack
        Update V = V + v_i and W = W + w_i
    else
        Add a fraction of item i to the knapsack
        Update V = V + v_i · (W̲ − W)/w_i
        Break
    end if
end for
return V
```

(d) The dual of (1) is: Given we have a maximizing question as our primal question, the dual should be a minimization problem. For the first constraint, we have a $\leq$ inequality, so the dual should have a $\geq$ inequality, and we introduce variable y. For the second constraint, as negative values are not allowed, we have a $\geq$ inequality, and we assign z_i to the upper bound constraint. The dual of (1) is:

$$\begin{aligned} \min_{y \in \mathbb{R}^n} \quad & W \cdot y + \sum_{i=1}^{n} z_i \\ & y w_i + z_i \geq v_i \qquad \forall i \in [n] \\ & y \geq 0 \\ & z_i \geq 0 \qquad \forall i \in [n] \end{aligned}$$

Problem 2 (Bottleneck and Lower-Binding Edges) Let G be a flow network with integer edge capacities. An edge in G is upper-binding if increasing its capacity by 1 also increases the value of the maximum flow in G. Similarly, an edge is lower-binding if decreasing its capacity by 1 also decreases the value of the maximum flow in G.

(a) Does every flow network G have at least one upper-binding edge? If so, sketch a proof. If not, give a counterexample.

(b) Does every flow network G have at least one lower-binding edge? If so, sketch a proof. If not, give a counterexample.

(c) Recall that an edge is saturated under a flow f if $f(e) = c(e)$. Prove or disprove: Given a flow network G, if an edge e is saturated under a max flow f, then e is an upper-binding edge.

(d) Given a flow f on a flow network, we define the divergence $div(v)$ at a vertex as:

$$\sum_{(v,u)\in E} f_{vu} - \sum_{(w,v)\in E} f_{wv}.$$

Sketch a proof that given any subset $S \subseteq V$:

$$\sum_{v\in S} div(v) = \sum_{(u,v)\in E(S,\overline{S})} f_{uv} - \sum_{(u,v)\in E(\overline{S},S)} f_{uv},$$

i.e. the total amount of flow out of S minus the total amount of flow into of S is equal to the total divergence inside of S.

(This is a discrete version of what is known as the divergence theorem in multivariable calculus, though understanding this is not needed to solve this problem).

(e) Formally prove the following statement: Given a flow network G, if an edge e is an upper-binding edge, then e is saturated under any max flow f.

Collaborators:

Solution: The solution is as follows:

1. No. Consider a network where there're two paths from s to t, we have edge su with capacity 1 and edge ut with capacity 1, and we have edge sv and edge vt with capacity 1 as well. Increasing the capacity of edge su by 1 does not increase the value of the maximum flow in G as t is only able to receive a flow of 2.

2. Yes. Accordingyo Max-Flow Min-Cut theorem, the value of the maximum flow in a network is equal to the capacity of the minimum cut. Therefore, if we decrease the capacity of any edge in the minimum cut, the value of the maximum flow will decrease as well, so we have at least one edge in every minimum cut must be lower-binding.

3. This is not true: Let's have a similar network as in part (a), but instead of having sv and vt with capacity 1, we have st directly with capacity 2. In this case, all edges are saturated under the max flow which is 3, but increasing the capacity of edge su or ut by 1 does not increase the value of the maximum flow in G. So by the definition of upper-binding edge, e is not an upper-binding edge.

4. The proof is as follows: The divergence at a vertex v is the total flow into v minus the total flow out of v, which is the net outflow of v. The sum of divergence over all vertices in S is equal to all outflows from vertice in S subtracting all inflows to vertices in S. Those flows that flow from S tp S would

not count as they do not contribute to the net boundary flow. Those flows that effectively cross the boundary of S would contribute to the net boundary flow, and is therefore captured as

$$\sum_{v \in S} div(v) = \sum_{(u,v) \in E(S,\overline{S})} f_{uv} - \sum_{(u,v) \in E(\overline{S},S)} f_{uv},$$

This concludes our proof.

5. The proof is as follows: We will prove this by contradition. Assume for contradition that edge e is an upper-binding edge, but is not saturated under any max flow f. This means that increasing the capacity of e by 1 would increase the value of the maximum flow in G, but $f(e) < c(e)$. However, this contradicts the definition of an upper-binding edge, which states that increasing the capacity of e by 1 would increase the value of the maximum flow in G, as it's not saturated so increasing the capacity would not increase the flow in the network. Therefore, edge e must be saturated under any max flow f.

Problem 3 (A Flowy Metric) Consider an undirected graph $G = (V, E)$ with capacities $c_e \geq 0$ on all edges $e \in E$. G has 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, in our case, c_e can be any non-negative number.

(a) Show that for any two vertices $s, t \in V$, the max flow from s to t is at least 1.

(b) Define the length of a flow f to be $\texttt{length}(f) = \sum_{e \in E} |f_e|$. Define the flow distance $d_{\text{flow}}(s, t)$ to be the minimum length of any s-t flow f that sends one unit of flow from s to t and satisfies all capacity constraints (i.e. $|f_e| \leq c_e$ for all $e \in E$). Write an LP that computes $d_{\text{flow}}(s, t)$.

(c) Prove that if $c_e = 1$ for all $e \in E$, then $d_{\text{flow}}(s, t)$ 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. You want to show $d_{\text{flow}}(s, t) = d(s, t)$ which is equivalent to showing that $d_{\text{flow}}(s, t) \leq d(s, t)$ and $d_{\text{flow}}(s, t) \geq d(s, t)$.)

Collaborators:

Solution: The solution goes as follows:

(a) The max flow from s to t is at least 1 because the capacity of any cut in G is at least 1. By the Max-Flow Min-Cut theorem, the value of the maximum flow in a network is equal to the capacity of the minimum cut. Therefore, the value of the maximum flow from s to t is at least 1.

(b) To compute $d_{\text{flow}}(s, t)$, We could have the following LP: As we want to minimize the length of any s-t flow that sends one unit of flow from s to t, we can minimize the sum of the absolute values of the flows on all edges. So we have

$$\min_{f \in \mathbb{R}^{|E|}} \quad \sum_{e \in E} |f_e|$$

For constrains, we have the following:

(1) we need to preserve the flow conservation, so we have $\sum_{(u,v) \in E} f_{uv} - \sum_{(v,w) \in E} f_{vw} = 0$ for all $v \in V \setminus \{s, t\}$.

(2) we need to send exactly one unit of flow from s to t, so we have $\sum_{(s,v) \in E} f_{sv} - \sum_{(v,s) \in E} f_{vs} = 1$. As the graph is undirected, we also need to have $\sum_{(t,v) \in E} f_{tv} - \sum_{(v,t) \in E} f_{vt} = -1$.

(3) Regarding the capacity constraints, we have $-c_e \leq f_e \leq c_e$ for all $e \in E$.

To linearlize the absolute value, we can introduce two variables f_e^+ and f_e^- for each edge e, and add the following constraints: $f_e = f_e^+ - f_e^-$ and $f_e^+, f_e^- \geq 0$. Therefore, we have the following LP:

$$\begin{aligned}
\min_{f^+, f^- \in \mathbb{R}^{|E|}} \quad & \sum_{e \in E} f_e^+ + f_e^- \\
& \sum_{(u,v) \in E} f_{uv} - \sum_{(v,w) \in E} f_{vw} = 0 \qquad \forall v \in V \setminus \{s, t\} \\
& \sum_{(s,v) \in E} f_{sv} - \sum_{(v,s) \in E} f_{vs} = 1 \\
& \sum_{(t,v) \in E} f_{tv} - \sum_{(v,t) \in E} f_{vt} = -1 \\
& -c_e \leq f_e^+ - f_e^- \leq c_e \qquad \forall e \in E \\
& f_e^+, f_e^- \geq 0 \qquad \forall e \in E
\end{aligned}$$

(c) For the upper bound $d_{flow}(s,t) \leq d(s,t)$, we could take the shortest path from s to t consisting of $d(s,t)$ edges, and assign a flow of 1 to each edge. This flow satisfies all capacity constraints, and has length $d(s,t)$, so $d_{flow}(s,t) \leq d(s,t)$. For the lower bound $d_{flow}(s,t) \geq d(s,t)$, since any flow from s to t must send one unit of flow from s to t, the length of any flow from s to t is at least the length of the shortest path from s to t, otherwise it would imply a shorter path exists, which contradicts the definition of the shortest path. Therefore, $d_{flow}(s,t) \geq d(s,t)$, and we have $d_{flow}(s,t) = d(s,t)$.

Problem 4 (Road Trip 4: Road Trip Game) Lorenzo is going on another trip, and he's taking Ruimin along with him. They've already figured out the shortest path to take, and the optimal gas stations to stop at along the way - all they need left is some entertainment. Thankfully, Lorenzo has thought of a great game to keep them entertained.

The game is specified by an undirected, unweighted graph $G = (V, E)$ and two vertices $s, t \in V$. Lorenzo picks an edge $e \in E$, and Ruimin simultaneously picks a cut $S \subseteq V$ which must separate s and t i.e. $s \in S$ but $t \notin S$. We define the edge boundary of a cut S to be the set of edges crossing the cut:

$$E(S, \overline{S}) := \{uv \in E \mid u \in S, v \in \overline{S}\}.$$

Lorenzo wins the game if the edge he picked is in Ruimin's cut's edge boundary, and Ruimin wins otherwise. Formally, for the selected e and S, Lorenzo's payoff $P(e, S)$ is given by

$$P(e, S) = \begin{cases} 1 & \text{if } e \in E(S, \overline{S}), \\ 0 & \text{otherwise.} \end{cases}$$

Lorenzo is trying to maximize $P(e, S)$, and Ruimin is trying to minimize it.
We are interested in mixed strategies for this game: Lorenzo will assign each edge e some probability $p_e \geq 0$, and randomly select an edge accordingly. Similarly, Ruimin's strategy will involve randomly selecting a cut c, where each cut c is selected with probability p_c.

(a) Lorenzo knows that he can't outsmart Ruimin, so he knows that whatever strategy he picks, Ruimin is just going to read his mind and figure it out. To determine his best strategy, he is going to solve the following LP:

$$\begin{aligned} \max_{\alpha, q} \quad & \alpha \\ & \sum_{e \in C} q_e \geq \alpha \quad \forall C \in \mathcal{C} \\ & \sum_{e} q_e = 1 \\ & 0 \leq q_e \quad \forall e \in E(G) \end{aligned}$$

where $\mathcal{C}$ is the set of s-t cuts in G. Explain, in words, how this LP represents the game.

(b) Ruimin knows that Lorenzo is a cheater, so he knows that whatever strategy he picks, Lorenzo is going to sneakily take a glance at it before deciding on his own strategy. In that case, write the linear program that Ruimin must solve to determine his optimal strategy. (Assume that Lorenzo picks his strategy after Ruimin.)

(c) Demonstrate that these programs are each other's dual. (You may use the fact that the dual of a dual program is the primal.)

(d) Part (c) implies the existence of some probability p such that:

- If Lorenzo plays optimally, his probability of winning is at least p, even if Ruimin knows his strategy.
- If Ruimin plays optimally, Lorenzo's probability of winning is at most p, even if Lorenzo knows Ruimin's strategy.

Provide a short argument (2 sentences) that part (c) implies this.

(e) Sketch a proof that the p from part (d) is $p = \frac{1}{L}$ where L is the length of a shortest path from s to t. This is Lorenzo's expected payoff from the game if both players play optimally, and is known as the value of the game.

Collaborators:

Solution: The solution is as follows:

(a) The LP represents the game as follows: Lorenzo is trying to maximize the probability of winning the game, which is represented by the variable α. $\max \alpha$ represents that Lorenzon is trying to maximize the probability of him winning the game over all possible strategies that Ruimin might choose. The constraint $\sum_{e \in C} q_e \geq \alpha$ for all $C \in \mathcal{C}$ represents that the probability of winning the game is at least α for all possible cuts that Ruimin might choose. The constraint $\sum_e q_e = 1$ represents that the probabilities of all possible strategies that Lorenzo might choose must sum to 1, ensuring a valid probability distribution. The constraint $0 \leq q_e$ for all $e \in E(G)$ represents that the probabilities of all possible strategies that Lorenzo might choose must be non-negative.

(b) Given that Lorenzo can adapt after Ruimin has chosen his strategy, Ruimin must minimize Lorenzo's probability of winning the game. The LP that Ruimin must solve is:

$$\begin{aligned} \min_{\beta, r} \quad & \beta \\ & \sum_{C: e \in E(C, \overline{C})} q_C \leq \beta \quad \forall e \in E(G) \\ & \sum_{C \in \mathcal{C}} q_C = 1 \\ & q_C \geq 0 \quad \forall C \in \mathcal{C} \end{aligned}$$

where r_C represents the probability of Ruimin choosing cut C, and the constraints ensure that the maximum probability that Lorenzo wins the game is β.

(c) The dual can be shown as follows:
The dual of the LP that Lorenzo must solve is: Each constraint in the primal LP corresponds to a dual variable p_C, which represents the probability of selecting the cut C. The primal objective to maximize α corresponds to minimizing the dual objective β, reflecting the tightest upper bound that Ruimin wants to achieve on Lorenzo's probability of winning.
The dual of the LP that Ruimin must solve is: Each constraint in the primal LP corresponds to a dual variable q_e, which represents the probability of selecting the edge e. The primal objective to minimize β corresponds to maximizing the dual objective α, reflecting the tightest lower bound that Lorenzo wants to achieve on Ruimin's probability of winning. The dual of a dual program is the primal program, so the LPs are each other's dual.

(d) The duality of the LPs implies that there exists a probability p such that if Lorenzo plays optimally, his probability of winning is at least p, even if Ruimin knows his strategy, and if Ruimin plays optimally, Lorenzo's probability of winning is at most p, even if Lorenzo knows Ruimin's strategy. The strong duality in LPs implies that at optimality, the primal and dual objectives are equal, so the probability p is the same for both players.

(e) Given the optimal play, the shortest path from s to t is the optimal strategy for Ruimin as it represents the minimum number of edges needed to cut to seperate s from t. Assuming each edge on this path is equally likely to be selected by Lorenzo (since any edge being cut guarantees s, t are separated), and there're L edges on the shortest path, so the probability for each edge to be selected should be ideally $\frac{1}{L}$. Therefore, $p = \frac{i}{L}$ becomes the probability of Lorenzo winning if both players play optimally.