CS 170 HW 9

Due **2019-10-30, at 10:00 pm**

1 Study Group

List the names and SIDs of the members in your study group.

2 Vacation

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 LP for the relaxed version of maximum independent set.

Solution: The relaxed version of maximum independent set is as follows:

$$\begin{aligned} \max \quad & \sum_{v \in V} x_v \\ \text{subject to: for all } (u,v) \in E: \quad & x_u + x_v \leq 1 \\ \text{for all } v \in V: \quad & x_v \geq 0 \end{aligned}$$

(b) Find the dual LP of the LP you made in part (a). What problem does the dual represent?

Solution: The dual is as follows:

$$\begin{aligned} \min \quad & \sum_{(u,v) \in E} y_{uv} \\ \text{subject to: for all } v \in E: \quad & \sum_{(v,u) \in E} y_{vu} \geq 1 \\ \text{for all } (u,v) \in V: \quad & y_{uv} \geq 0 \end{aligned}$$

The dual is the relaxed version of minimum edge cover.

(c) True or false: For any connected graph, the optimum value for a) always equals the optimum value for b). If true, prove. If false, give a counterexample.

Solution: True. This just follows from strong duality for LPs, if we can show the primal optimal is bounded. The primal optimal is bounded because every vertex v has an edge leading out of it, so $x_v \leq 1$, so the sume is at most $|V|$.

(d) Consider the 'non-relaxed'/'tightened' version of your LPs for part a) and part b). You can build the tightened LP by adding the constraint that all of the variables also must be integers.

True or false: For any connected graph, the optimum value for the tightened version of a) always equals the optimum value for the tightened version of b). If true, prove. If false, give a counterexample.

Solution: The tightened version of a) is maximum independent set. The tightened version of b) is minimum edge cover.

Consider K_3, the three-vertex graph with an edge between every pair. The maximum independent set has size 1, while the smallest edge cover has size 2. So this is a countexample to the statement.

3 Generalized Max Flow

Consider the following generalization of the maximum flow problem.
You are given a directed network $G = (V, E)$ where edge e has capacity c_e. Instead of a single (s, t) pair, you are given multiple pairs $(s_1, t_1), ..., (s_k, t_k)$, where the s_i are sources of G and t_i are sinks of G. You are also given k (positive) demands $d_1, ..., d_k$. The goal is to find k flows $f^{(1)}, ..., f^{(k)}$ with the following properties:

(a) $f^{(i)}$ is a valid flow from s_i to t_i.

(b) For each edge e, the total flow $f_e^{(1)} + f_e^{(2)} + ... + f_e^{(k)}$ does not exceed the capacity c_e.

(c) The size of each flow $f^{(i)}$ is at least the demand d_i.

(d) The size of the *total* flow (the sum of the flows) is as large as possible.

Write a linear problem using the variables $f_e^{(i)}$ whose optimal solution is exactly the solution to this problem. For each constraint as well as the objective in your linear program briefly explain why it is correct. (Note: Since linear programs can be solved in polynomial time, this implies a polynomial-time algorithm for the problem)

Solution: We have variables $f_{(u,v)}^{(i)}$ for all $1 \leq i \leq k$ and $(u, v) \in E$. The total flow across each edge should be at most the capacity of the edge. Hence, we have the constraint

$$\forall (u,v) \in E \qquad \sum_{i=1}^{k} f_{(u,v)}^{(i)} \leq c_{(u,v)}$$

Also, flow conservation for the flow $f^{(i)}$ must hold for all vertices except s_i and t_i. This gives the set of constraints

$$\forall 1 \leq i \leq k, \forall v \in V \backslash \{s_i, t_i\} \qquad \sum_{(u,v) \in E} f_{(u,v)}^{(i)} = \sum_{(v,w) \in E} f_{(v,w)}^{(i)}$$

Finally, the for each i, $f^{(i)}$ must be greater than the corresponding demand.

$$\forall 1 \leq i \leq k \qquad \sum_{(s_i,u) \in E} f_{(s_i,u)}^{(i)} \geq d_i$$

The objective is simply to maximize the sum of all the flows.

$$\max \sum_{i=1}^{k} \sum_{(s_i,u)\in E} f^{(i)}_{(s_i,u)}$$

4 Repairing a Flow

In a particular network $G = (V, E)$ whose edges have integer capacities c_e, we have already found a maximum flow f from node s to node t where f_e is an integer for every edge. However, we now find out that one of the capacity values we used was wrong: for edge (u, v) we used c_{uv} whereas it 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 redo the flow computation from scratch, but there's a faster way.

Describe an algorithm to repair the max-flow in $O(|V| + |E|)$ time. Also give a proof of correctness and runtime justification.

Solution:

Main Idea: In the residual graph with the original capacity for (u, v), use DFS to find a path from t to v and from u to s. Join these paths by adding the edge (v, u) in between them to get a path p. Update f by pushing 1 unit of flow from t to s on p (i.e. for each $(i, j) \in p$, reduce f_{ji} by 1). Finally, use DFS to see if the residual graph of the resulting graph, using the new capacity for (u, v), has an $s - t$ path. If so, push 1 unit of flow on this path.

Proof of Correctness: Consider the residual graph of G. If (u, v) is at capacity, then there is a flow of at least 1 unit going from v to t, and since f_e are integral there is a path from v to t with at least one unit of flow moving through it. So the residual graph will have a path from t to v. Similarly, there will be a path from u to s in the residual graph, so the algorithm can always find p correctly.

Note that the size of the maximum flow in the new network will be either the same or 1 less than the previous maximum flow (because changing one edge can change the capacity of the min-cut by at most 1). In the former case, after pushing flow from t to s on p, it is possible to push more flow from s to t, so there must be an $s - t$ path in the new residual graph, which the algorithm will find and push at least 1 unit of flow on because all capacities and flow values are integral. Thus the algorithm finds the new max flow correctly. In the latter case, we will already have the max flow after pushing flow backwards on p, so our algorithm is still correct.

Runtime: The runtime is $O(|V| + |E|)$ because the algorithm just calls DFS thrice.

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 $\forall v : \sum_{u:(u,v)\in E} f_{uv} \leq c_v$.

(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.

Solution:

(a) Split every vertex v into two vertices, v_{in} and v_{out}. For each edge (u, v) with capacity c_{uv} in the original graph, create an edge (u_{out}, v_{in}) with capacity c_{uv}. Finally, if v has capacity c_v, then create an edge (v_{in}, v_{out}) with capacity c_v. If F' is a flow in this graph, then setting $F(u, v) = F'(u_{out}, v_{in})$ gives a flow in the original graph. Moreover, since the only outgoing edge from v_{in} is (v_{in}, v_{out}), and incoming flow must be equal to outgoing flow, there can be at most c_v flow passing through v. Likewise, if F is a flow in the original graph, setting $F'(u_{out}, v_{in}) = F(u, v)$, and $F'(v_{in}, v_{out}) = \sum_u F(u, v)$ gives a flow in G'. One can easily see that these flows have the same size.

(b) Create one "supersource" S with edges (S, s_i) for each s_i, and set the capacity of these edges to be infinite. Then if F is a flow in G, set $F'(S, s_i) = \sum_u F(s_i, u)$. Conversely, if F' is a flow in G', just set $F(u, v) = F'(u, v)$ for $u \neq S$, and just forget about the edges from S. One can easily see that these flows have the same size.

6 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.

Solution:

1. For any two vertices s,t, the min cut separating s and t has value at least 1. Therefore, by the max flow-min cut theorem, the maximum $s-t$ flow has value at least 1.

2. First, we show that $d_{\texttt{flow}}(s,t) \leq d(s,t)$ for all $s,t \in G$. Let f be the unit $s-t$ flow that sends one unit of flow along the shortest path from s to t. Then $\texttt{length}(f)$ is the length of this path, which is $d(s,t)$. Since $d_{\texttt{flow}}(s,t)$ is the minimum length of any unit $s-t$ flow, $d_{\texttt{flow}}(s,t) \leq \texttt{length}(f) = d(s,t)$.

   Now, we show that $d(s,t) \leq d_{\texttt{flow}}(s,t)$. Consider the minimizing flow; that is the unit $s-t$ flow f with length $d_{\texttt{flow}}(s,t)$. Any unit $s-t$ flow f can be written as

   $$f = \sum_{\text{s-t paths } p \text{ in } G} w_p f_p$$

   where f_p is the unit flow along the path p, $\sum_{\text{s-t paths } p} w_p = 1$, $w_p \geq 0$ for all $s-t$ paths p, and all paths go in the same direction from s to t through any edge e. In other words, we can argue that a flow f will send some (potentially zero) fraction of its flow through each possible $s-t$ path. For example, if there are exactly 2 paths from s to t, we could send exactly half of our flow along each path, or .3 along one path and .7 along the other. Notice this is true regardless of whether or not these paths are disjoint.

   This means that

   $$\texttt{length}(f) = \sum_{\text{s-t paths } p \text{ in } G} w_p \texttt{length}(f_p)$$

   i.e. $\texttt{length}(f)$ is the average of the lengths of many $s-t$ paths in G. In particular, there exists an $s-t$ path q in G with $\texttt{length}(f_q) \leq \texttt{length}(f) = d_{\texttt{flow}}(s,t)$. $d(s,t) \leq \texttt{length}(f_q)$, finishing the proof of the desired inequality.

3. Let f and g be the $s-u$ and $u-t$ flows with lengths $d_{\texttt{flow}}(s,u)$ and $d_{\texttt{flow}}(u,t)$ respectively. We wish we could define an $s-t$ flow $r = f+g$ and use this flow to show that $d_{\texttt{flow}}(s,t) \leq \texttt{length}(f) + \texttt{length}(g)$. $f+g$ does not have to respect the capacities c, so it is not feasible. This is strange, though, because f and g individually do satisfy the capacities c. In order for $f+g$ to violate an edge capacity, flow from both f and g must cross the violated edge. But this implies that the $f+g$ flow would form a cycle, which is wasteful. We want to remove these cycles.

   We now formalize this intuition using max flow-min cut duality. Define a new capacity vector c' with $c'_e = \max(|f_e|, |g_e|)$. Notice that $c'_e \leq c_e$ for all edges e because f and g

respect G's capacities, so any c'-respecting flow is also a c-respecting flow. Futhermore, any c'-respecting flow h has length at most

$$\texttt{length}(h) \leq \sum_e c'_e \leq \sum_e |f_e| + |g_e| = \texttt{length}(f) + \texttt{length}(g) = d_{\texttt{flow}}(s,u) + d_{\texttt{flow}}(u,t)$$

Therefore, it suffices to show that there is a unit $s-t$ c'-respecting flow. By max flow-min cut, it suffices to show that all $s-t$ cuts have capacity at least 1. Consider any set of vertices S containing s but not t. If $u \notin S$, then $1 \leq \sum_{e \in \partial S} |f_e| \leq \sum_{e \in \partial S} c'_e$. If $u \in S$, then $1 \leq \sum_{e \in \partial S} |g_e| \leq \sum_{e \in \partial S} c'_e$. In particular, the cut defined by S always has capacity at least 1. Therefore, the min $s-t$ c'-cut has capacity at least 1. Therefore, there exists a unit $s-t$ flow with length at most $d_{\texttt{flow}}(s,u) + d_{\texttt{flow}}(u,t)$, as desired.