Prof. Friedrich Eisenbrand
Martin Niemeier

Due Date: May 12, 2009
Discussions: May 06, 2009

Introduction to Discrete Optimization

Spring 2009

Solutions 10

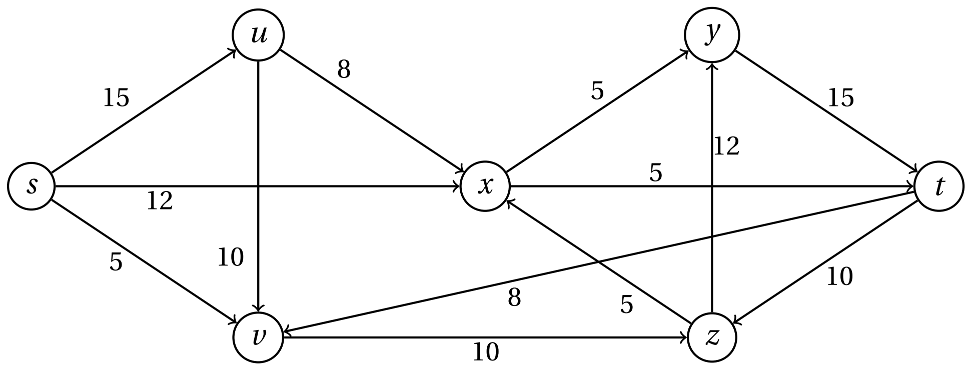
Exercise 1
The Lucky Puck Company has a factory in Vancouver that manufactures hockey pucks, and it has a warehouse in Winnipeg that stocks them. Luck Puck leases space on trucks from another company to ship the pucks from the factory to the warehouse. Because the trucks travel over specified routes between cities and have a limited capacity, Lucky Puck can ship at most $c(u,v)$ crates per day between each pair of cities u and v. Lucky Puck has no control over these routes and capacities and so cannot alter them. Their goal is to determine the largest number p of crates per day that can be shipped from the factory to the warehouse.

Show how to compute p by finding a maximum flow in a network.

Solution
Let V be the set of cities that are possible starting and endpoints for the trucks. For each pair of cities $u, v \in V$, $u \neq v$, if $c(u,v) > 0$, then add an arc (u,v) to the set A.

The number p is the value of a max $Vancouver - Winnipeg$-flow in V.

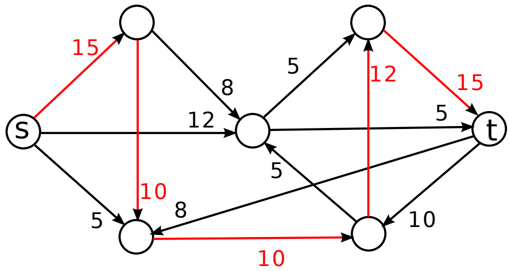
Exercise 2
Consider the following network:



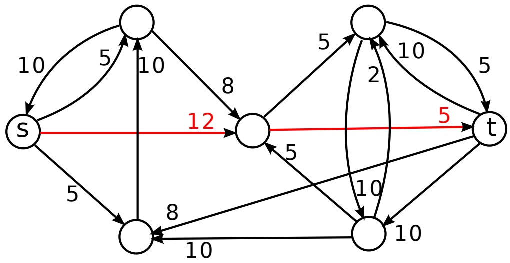
Run the Ford-Fulkerson algorithm to compute a max $s-t$-flow. For each iteration give the residual network and mark the path you choose for augmentation.

Further give a minimum $s-t$-cut in the network.

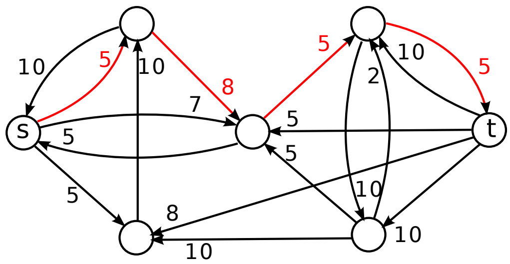
Solution
The first residual network is idential to the original network. The $s-t$-flow we choose to augment is marked red. We can augment 10 units of flow:



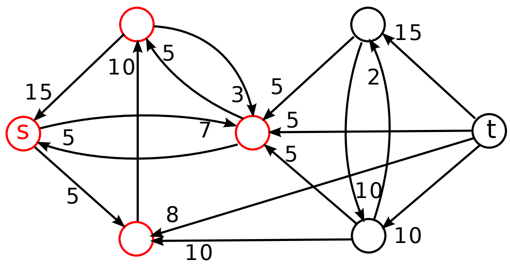
The residual network after augmentation looks as follows. The next augmenting path is marked red. We can augment 5 units of flow.



The residual network after augmentation looks as follows. The next augmenting path is marked red. We can augment 5 units of flow.



The residual network after augmentation looks as follows:



There is no more $s-t$-path in the residual network. Thus our flow is maximal and of value 20. The nodes marked red are reachable from s. Thus $\{s,u,v,x\}$ defines a minimal $s-t$-cut.

Exercise 3
Given a network $D=(V,A)$ with rational capacities $c: A \to \mathbb{Q}_{\geq 0}$, show that the Ford-Fulkerson algorithm terminates even if we do not choose a shortest path for augmentation in each iteration, i.e. give a bound on its running time (this bound needs not to be polynomial in the input size).

Solution

Without loss of generality we assume that c is integer, i.e. $c(a) \in \mathbb{Z}_{\geq 0}$ for each $a \in A$. Otherwise we find an integer M such that $M \cdot c(a) \in \mathbb{Z}_{\geq 0}$ for each $a \in A$ and solve the max-flow-problem for capacities $M \cdot c(a)$ instead.

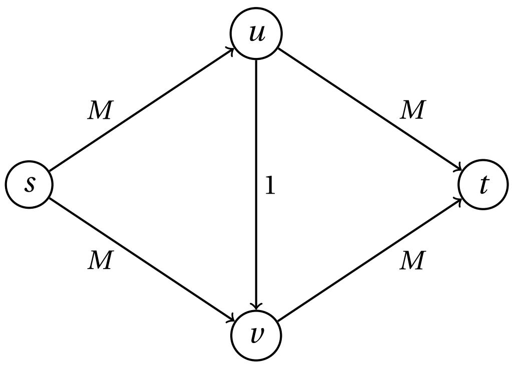
An upper bound for the value of a max flow is the sum of the arc capacities:

$$C := \sum_{a \in A} c(a).$$

In each iteration of the Ford-Fulkerson algorithm, the flow is augmented along an $s-t$-path, which increases the value of the flow by a value $\epsilon > 0$. Since all capacities are integer, we have $\epsilon \geq 1$. Thus, after at most C iterations, the Ford-Fulkerson algorithm terminates.

Exercise 4

Consider the following network:



Explain why the Ford-Fulkerson algorithm might take an exponential number of iterations ($2 \cdot M$ iterations) if the augmenting paths are chosen in a disadvantageous way.

Solution

If the Ford-Fulkerson algorithm chooses the path (s, u, v, t) for augmentation in the first iteration, the path (s, v, u, t) becomes available in the second iteration. If this path gets chosen in the second iteration, again the path (s, u, v, t) becomes available. The amount of flow that can be augmented along these flows is limited by 1.

Thus, if we Ford-Fulkerson algorithm alternating chooses these two path as long as possible, it will take $2 \cdot M$ iterations.

Exercise 5

An *undirected graph* $G = (V, E)$ is a set of *nodes* together with a set of *edges* $E \subseteq \{\{u, v\} : u, v \in V\}$. G is *connected* if for each pair of nodes $u, v \in V$ there is a path from u to v.

The *edge connectivity* of an undirected graph is the minimum number k of edges that must be removed such that the resulting graph is not connected anymore.

Show how the edge connectivity of an undirected graph $G = (V, E)$ can be determined by running a maximum-flow algorithm on at most $|V|$ flow networks, each having $O(V)$ vertices and $O(E)$ arcs.

Hint: Consider the bidirected graph $D = (V, A)$, where $A = \{(u, v) : \{u, v\} \in E\}$ (for each edge {u,v} we have the arcs (u, v) and (v, u)). Is there any relation between the edge connectivity of G and minimum size cuts in D?

Solution

For an undirected graph $G = (V, E)$ and $U \subseteq V$, let $\delta(U) = \{e \in E \ : \ e = \{u, v\}$ for some $u \in U, v \notin U\}$. Let k be the edge connectivity number of G.

We claim that

$$k = \min_{U \subset V} |\delta(U)|. \tag{1}$$

To see that $k \leq \min_{\emptyset \subset U \subset V} |\delta(U)|$, observe that for each $\emptyset \subset U \subset V$, if we remove all edges in $\delta_G(U)$ from G, the graph becomes disconnected.

To see that $k \geq \min_{\emptyset \subset U \subset V} |\delta_G(U)|$, let M, $|M| = k$, be a set of edges such that $G' = (V, E \backslash M)$ is disconnected. Let $s \in V$ and let U be the set of nodes reachable from s in G'. Then $\delta_G(U) \subseteq M$ and thus $\min_{\emptyset \subset U \subset V} |\delta_G(U)| \leq k$.

This shows (1). Thus it is sufficient to find a mimimum size cut $\delta_G(U)$ in G.

Now consider the bidirected graph $D = (V, A)$, where $A = \{(u, v) : \{u, v\} \in E\}$ with unit capacities $c : A \to \mathbb{Z}_{\geq 0}$, $c(a) = 1 \ \forall a \in A$. Observe that for each $U \subseteq V$ we have $|\delta_G(U)| = |\delta_D^{out}(U)| = |\delta_D^{in}(U)|$. Thus it is sufficient to find a minimum size cut in D.

Let $\emptyset \subset U \subset V$ such that $|\delta_G(U)| = |\delta_D^{out}(U)|$ is minimal. Fix an arbitrary node $s \in V$. If $s \in U$, then $\delta(U)$ is an $s-t$ cut for some $t \in V \backslash U$. If $s \notin U$, then $\delta(V \backslash U)$ is an $s-t$ cut for some $t \in U$. Since $|\delta_D^{out}(U)| = |\delta_D^{in}(U)| = |\delta_D^{out}(V \backslash U)| = |\delta_G(V \backslash U)|$, $\delta(V \backslash U)$ is a minimum size cut as well.

Since the size of a min $s-t$ cut equals the value of a maximum $s-t$ flow (Max flow min cut theorem), this shows that if we fix some node $s \in V$ and solve the max $s-t$ flow problem for each $t \in V$, $t \neq s$, the smallest result gives us the size of a min cut and therefore the connecitity number k of G.