# Homework #9
# Introduction to Algorithms/Algorithms 1
# 600.363/463
# Spring 2014
# Solutions

April 8, 2014

## 1 Problem 1 (20 points)

Let $G = (V, E)$ be a flow network with source node $s$ and sink node $t$ and non-negative integer capacities, so that $c(a, b) \in \{0, 1, 2, \dots\}$ for all $(a, b) \in E$. Assume that $G$ has a unique minimum cut. Suppose that we have already found the optimal flow on graph $G$ and that this flow is an integral flow (i.e., for all $(a, b) \in E$, $f(a, b)$ is an integer). Call this flow $f$. Suppose now that we are informed that one capacity in this graph needs to be changed. In particular, we are going to change the capacity of edge $(u, v) \in E$ so that its new capacity is $c(u, v) - 1$, while all other edges retain their old capacities. Edge $(u, v)$ is such that $(u, v)$ crosses the unique minimum cut in graph $G$, and our flow $f$ uses edge $(u, v)$ to its full capacity, i.e., $f(u, v) = c(u, v)$. One might think that computing the flow with this updated capacity would require recomputing the flow from scratch (e.g., by running Ford-Fulkerson all over again), but it turns out that we can compute the new optimal flow in $O(|V| + |E|)$ time.

Give an algorithm that

(i) Accepts a flow network $G = (V, E)$ (with non-negative integer capacities that is guaranteed to have a unique minimum cut), a maximum flow $f$ for that flow network, and an edge $(u, v) \in E$ for which $f(u, v) = c(u, v)$.

(ii) Returns a new flow for the same graph with new capacity function

$$c'(a, b) = \begin{cases} c(a, b) - 1 & \text{if } a = u, b = v \\ c(a, b) & \text{otherwise.} \end{cases}$$

(iii) Runs in $O(|V| + |E|)$ time.

Prove the correctness of your algorithm and prove its runtime.

## Solution

Intuitively, this problem is very similar to the situation in Ford-Fulkerson where we would like to "take flow back" along an edge. Consider the flow network $G$, with edges labeled according to how much flow is sent along them by the maximum flow $f$, so that $w(a,b) = f(a,b)$ for all $(a,b) \in E$. We need to adjust the flow in $G$ so that all capacities are respected again (and so that flow is still conserved) and so that the new flow over edge $(u,v)$ is adjusted down by one unit. Note that this would be an easy thing to do if we knew that there existed a path from the source $s$ to sink $t$ that included edge $(u,v)$ on which, for every edge $(a,b)$, our original flow $f$ had value $f(a,b) \geq 1$. If such a path existed (and if we could find it), then we could simply subtract one unit of flow from every edge on the path. The resulting new flow would obey all capacity constraints, since we would have adjusted the flow over $(u,v)$ down by one and adjusted the flow into and out of that edge so that conservation of flow is preserved. Note that the existence of such a flow is guaranteed by the fact that all edges in the original flow are guaranteed to have integral flows over them and by the conservation of flow (so that the fact that edge $(u,v)$ originally had $f(u,v)$ units of flow coming into it guarantees that there exists total of $f(u,v)$ unites of flow coming into node $u$, and since $f(u,v)$ is an integer and all flows coming into $u$ must be integers, $f(u,v) > 0$ guarantees that there exists at least one node $a$ for which the original flow has $f(a,u) \geq 1$. Repeating this argument inductively back to $s$ and by a similar argument about the flow leaving node $v$, we can show that the path we want does indeed exist.

Finding such a path is simple– simply run BFS on the graph $H = (V, E')$, where $V$ is precisely the set of ndoes from flow network $G$ and $E' = \{(a,b) \in E : f(a,b) > 0\}$ and where the edge weights are given by $w(a,b) = f(a,b)$. Finding a shortest path in $H$ from $s$ to $u$ and a shortest path in $H$ from $v$ to $t$ (note that we don't necessarily have to use shortest paths, here, but that would be at the cost of a bit more work in our proof) and putting these paths together via edge $(u,v)$ gives the path we wanted in the previous paragraph.

One concern remains: it is possible that our new flow isn't actually a maximum flow! But we can rule this out by noting that we were guaranteed that edge $(u,v)$ was guaranteed to be in the minimum cut in graph $G$. Reducing $c(u,v)$ only makes the capacity of this cut smaller, so it is still the minimum cut in the graph. The new flow that we construct reduces the flow over $f(u,v)$ by one unit, so that the new flow is at capacity on edge $(u,v)$. By the min cut max flow theorem, our flow, which is equal to this minimum cut, must be a maximum flow.

## 2 Problem 2 (20 points)

1. **(10 points)** Prove or disprove the following claim: If $f$ is a maximum flow on a graph $G$ with source node $s$ and sink node $t$, then $f(a,t) = c(a,t)$ for all edges $(a,t) \in E$.

   **Solution:** this claim is false. Consider the following counter-example: we have four nodes, $\{s,a,b,t\}$, and four edges $\{(s,a),(s,b),(a,t),(b,t)\}$, with capacities $c(s,a) = 1$, $c(s,b) = 1$, $c(a,t) = 1$, $c(b,t) = 10$. The maximum flow in this graph is 2, and does not saturate edge $(b,t)$.

2. **(10 points)** Suppose you are given a flow network $G = (V,E)$ with source node $s \in V$ and sink node $t \in V$ with unit capacities (i.e., $c(a,b) = 1$ for all $(a,b) \in E$). Consider the following problem: given some non-negative integer $k$, we would like to construct a new flow network $G' = (V,E')$, constructed by removing $k$ edges from $G$ (so $E' \subset E$ with $|E'| = |E| - k$), so that the maximum $s$–$t$ flow in $G'$ is as small as possible.

   Give an algorithm which constructs such a $G'$ in $O(|E|f^*)$ time, where $f^*$ denotes the value of the maximum flow on $G$. Prove the correctness of your algorithm and its runtime.

   **Solution** It suffices to compute the minimum cut on graph $G$, and remove the $k$ largest-capacity edges from this cut. The min cut–max flow theorem states that the maximum flow in graph $G$ is precisely equal to the minimum cut, and that all flows in the graph are bounded above by all cuts. Removing the largest-capacity edges from the minimum cut in graph $G$ results in $G'$ having a minimum cut equal to the value of the minimum cut in $G$, minus the total value of the $k$ removed edges (which is $k$, unless the minimum cut in $G$ consists of fewer than $k$ edges). By the min-cut max-flow theorem, the maximum flow in $G'$ is precisely equal to the minimum cut in $G'$, which we made as small as possible. One concern remains, which is that perhaps there is some other set of edges in the graph whose removal could result in a still worse minimum cut, but this possibility is ruled out by the fact that all edges are unit weight– any other cut in $G$ has capacity at least that of the minimum cut, and since any such cut can be comprised only of unit-cost edges, removing $k$ edges from any other cut must result in a new cut with capacity at least that of the cut that we introduced by removing $k$ edges from $G$.

   We can compute the minimum cut in graph $G$ by computing the minimum flow, which takes $O(|E||f^*|)$ time, followed by breadth-first search in the residual network starting at the source node, to find all nodes reachable from the source in the residual network. This set of nodes, call it $S$, is precisely the partition that gives rise to the minimum cut (see the proof of min cut max flow in CLRS).

## Optional exercises

Solve the following problems and exercises from CLRS: 26.2-7, 26.2-11, 26.3-2, 26.3-5