Prof. Friedrich Eisenbrand
Martin Niemeier

Due Date: June 03, 2010
Discussions: May 20, May 27

# Discrete Optimization

Spring 2010

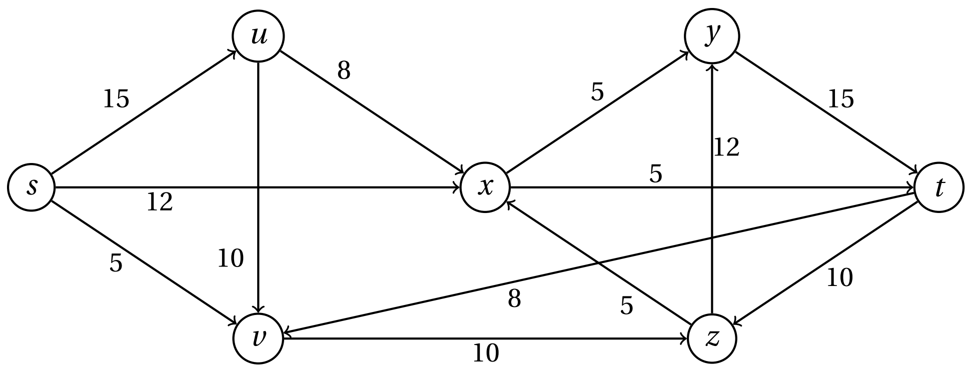
Assignment Sheet 6

You can hand in written solutions for up to two of the exercises marked with (*) or (Δ) to obtain bonus points. The duedate for this is June 03, 2010, before the exercise session starts. Math students are restricted to exercises marked with (*). Non-math students can choose between (*) and (Δ) exercises.

**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. Lucky 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.

**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.

**Exercise 3 (*)**
Recall that 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?*

**Exercise 4 (Δ)**

A *matching* in an undirected graph $G = (V, E)$ is a subset $M \subseteq E$ of the edges such that no two edges in $M$ share a common node of $V$.

The *matching-problem* is to find a matching $M$ of maximum cardinality.

Explain how to solve the matching-problem on a *bipartite graph* computing a maximum flow in an auxiliary network. What is the running time of your algorithm?

**Exercise 5 (∗)**

Consider a directed graph $G = (V, A)$, a node $s \in V$, a cost function $c : A \to \mathbb{N}_0$ and a benefit function $b : V \to \mathbb{N}_0$.

If you destroy a set $D \subseteq A$ of arcs, you have to pay a cost of $c(D)$. Let $S_D$ be the set of nodes *not* reachable from $s$ after removing the arcs of $D$ from the graph. You receive a benefit of $b(S_D)$ for destroying the edges $D$.

Solve the problem of finding an arc set $D$ such that $b(S_D) - c(D)$ is maximized using a maximum flow (min cut) algorithm. Prove that your construction is correct.