Max flow and min cost max flow

Han Hoogeveen

May 23, 2014
Basic problem description

Given: directed graph $G = (V, A)$ with special vertices $s$ and $t$.

- The goal is to find a flow $x$ from $s$ (source) to $t$ (sink) of maximum size.
- The flow $x_{ij}$ through arc $(i, j) \in A$ must satisfy $0 \leq x_{ij} \leq c_{ij}$; here $c_{ij}$ is called the capacity of arc $(i, j)$; there can be a positive lower bound on $x_{ij}$.
- ‘Inflow=Outflow’: The total amount of flow entering $v \in V \setminus \{s, t\}$ must be equal to the total amount of flow leaving $v$. 
Basic solution algorithm: Ford-Fulkerson

1. Start with the zero-flow (feasible if all lower bound capacities are 0).
2. Given the current flow, find a path from $s$ to $t$ along which you can send flow (this is called an augmenting path).
3. Augment the flow by sending as much as possible along this $s - t$ path.
4. Continue until you cannot find an augmenting path anymore.
Finding an augmenting path

Figure 1: (0, 1) means: flow = 0; capacity = 1

How to proceed?
Residual graph

Given the current flow $x$, construct the residual graph $\tilde{G} = (V, \tilde{A})$. For each arc $(i, j) \in A$ add to $\tilde{A}$

- arc $(i, j)$ with capacity $c_{ij} - x_{ij}$ (forward arc);
- arc $(j, i)$ with capacity $x_{ij}$ (backward arc).

An augmenting path corresponds to an $s - t$ path in $\tilde{G}$; its capacity is equal to the minimum capacity of its arcs in the residual graph.

It is computationally efficient to use augmenting paths consisting of a minimum number of arcs.

If the capacities are integral, then you find an integral flow.
Optimality proof

- The algorithm terminates if there is no $s - t$ path in $\tilde{G}$.
- Define $S$ as the set of vertices reachable from $s$; let $T$ denote the remaining vertices (including $t$).
- This cut $[S, T]$ splits the graph in two parts. All flow must go through arcs $(v, w)$ with $v \in S$ and $w \in T$.
- The capacity of the cut $[S, T]$ is equal to the total capacity of the arcs $(v, w)$ with $v \in S$ and $w \in T$; this is an upper bound on the size of the flow (this holds for any cut of $G$).
- The size of the current flow is equal to capacity of the determined $[S, T]$ cut.

This proves the famous max-flow min-cut theorem.
Max flow min cost

- Extension of the max flow problem;
- Each arc \((i, j) \in A\) has cost \(k_{ij}\).
- The cost of a flow \(x\) amounts to

\[ \sum_{(i, j) \in A} k_{ij}x_{ij}. \]

- Goal: determine the cheapest flow of maximum size.
Max flow min cost (2)

Same solution method.

1. Start with a **min-cost flow** of size 0.
2. Given the current flow, construct the residual graph \( \tilde{G} \).
   The cost of arc \((i, j) \in \tilde{A}\) is defined as
   \[
   l_{ij} = \begin{cases} \ k_{ij} & \text{if } (i, j) \in A \\ -k_{ji} & \text{if } (j, i) \in A. \end{cases}
   \]
3. Find the path in \( \tilde{G} \) from \( s \) to \( t \) with minimum length.
4. Augment the flow by sending as much as possible along this \( s - t \) path.
5. Continue until you cannot find an augmenting path anymore.
Initial flow

- A min-cost flow is a flow that has minimum cost within the set of flows of that size.
- A flow is a min-cost flow if and only if the residual graph contains no cycles with negative length.
- You can start with the flow $x_{ij} = 0$ for each $(i, j) \in A$, unless the corresponding residual graph has cycles of negative length; if such cycles exist, then these have to be filled with flow first.
Figure 1: All arcs have capacity 1; the depicted figure is the cost

How to proceed?
Successive augmenting paths do not get shorter.

Instead of using Bellman-Ford (running in $O(n^3)$ time) you can use Dijkstra’s $O(n^2)$ algorithm for finding the shortest paths, after adjusting the lengths (Johnson’s algorithm).

How to find the minimum cost flow with a given size?

How to find the overall minimum cost flow?

How to include lower bounds on the flow through the arcs?

How to find a feasible flow that leaves $b(v)$ units of flow in each vertex $v$ ($b(v)$ can be negative)?

How to find the maximum flow that leaves $b(v)$ units of flow in each vertex $v$?