Minimum Cost Flow

Algorithms and Networks
This lecture

- The minimum cost flow problem: statement and applications
- The cycle cancelling algorithm
- A polynomial time variant of cycle cancelling
- The successive shortest paths algorithm
Minimum Cost Flow I

• Edges have
  – **Capacity** $c(u,v)$: bound on amount of flow that can go through the edge
  – **Cost**: $\text{cost}(u,v)$: cost that must be paid per unit of flow that goes through the edge.

• Cost of flow $f$:
  – Sum over all $(u,v)$ of: $f(u,v) \times \text{cost}(u,v)$. 
Minimum cost flow problem

- **Given**: Network $G$, $c$, cost, $s$, $t$, and a target flow value $r$.
- **Question**: Find a flow from $s$ to $t$ with value $r$, with minimum cost.
Unbounded capacities

- Some edges may have unbounded capacities
  - If there is a cycle of negative cost with only edges with unbounded capacity:
    - Arbitrary small cost (degenerate case)
  - Otherwise: simple transformation to bounded capacities
    - E.g., set each unbounded capacity to sum of all bounded capacities
Separate demands

- Similar to max-flow with multiple sources and multiple sinks
- Or: work with “demand”, which can be positive or negative
Nonnegative arc costs

• We may assume all costs are nonnegative.
• In case of negative costs: assume bounded capacity. Modify network to equivalent one with nonnegative costs:

Or work with separate demands
Applications

- Transport problems
- Minimum cost matchings
- Reconstruction of Left Ventricle from X-ray projections
  - Image: 2d bit array; known are sums of columns, rows; probabilities for each bit
  - Look for image with correct row and column sums of maximum probability
  - Can be modelled as minimum cost flow problem
Application: Optimal loading of hopping airplane

\( b_{ij} \) weight units (or passengers) can be transported from \( i \) to \( j \)
Each gives a profit of \( f_{ij} \)
Plane can never carry more than \( p \) units
How much units do we transport of each type for maximum profit?

Capacity \( i \rightarrow j: p \)
Node \( ij \) has supply \( b_{ij} \)
Cost from \( ij \) to \( i \): \(-f_{ij}\)
Node \( i \) has demand
sum over all \( b_{ji} \)

All other arcs infinite cap.
All other costs 0
Residual network

- Capacities as for maximum flow algorithms.
- If \( f(u,v) > 0 \), then \( \text{cost}_f(u,v) = \text{cost}(u,v) \), and \( \text{cost}_f(v,u) = -\text{cost}(u,v) \).
Example

Suppose we send 1 flow from a to b

In $G_f$:

Capacity 5, cost 3
Capacity 2, cost 6
Capacity 4, cost 3
Capacity 2, cost 6
Capacity 1, cost -3

Universiteit Utrecht

Algorithms and Networks: Minimum Cost Flow
Cycle cancelling algorithm

- Make a feasible flow $f$ in the network
- while $G_f$ has a negative cycle do
  - Find a negative cycle $C$ in $G_f$
  - Let $D$ be the minimum residual capacity $c_f$ of an edge on $C$
  - Add $D$ units of flow to each edge on $C$: this is a new feasible flow of smaller cost
- Output $f$. 
Cycle cancelling algorithm is correct

- **Theorem**: a flow \( f \) has minimum costs, if and only if \( G_f \) has no negative cycle.
  - If \( G \) has negative cycle, then we can improve \( f \) to one with smaller cost.
  - Suppose \( f \) is a flow, and \( f' \) is an optimal flow. \( f' - f \) is a circulation in \( G_f \), hence a linear combination of cycles, and if \( f \) is not optimal, then the total cost of these cycles is negative, so there is a negative cycle in this set: it is a cycle in \( G_f \).
More on the cycle cancelling algorithm

• No guarantee that it uses polynomial time.

• Corollary: if all costs, capacities, and target flow value are integral, then there is an optimal integer minimum cost flow.
  – The cycle cancelling algorithm finds an integer flow in this case.

• Variant: using always the minimum mean cost cycle gives a polynomial time algorithm!
Minimum mean-cost circulation algorithm

- Cycle cancelling algorithm but find always the minimum mean cost cycle and use that.
  - $O(nm)$ time to find the cycle.
  - A theorem shows that $O(nm^2 \log^2 n)$ iterations are sufficient.
  - $O(n^2m^3 \log^2 n)$ algorithm.
Successive shortest paths

- Start with flow \( f \) with \( f(u,v) = 0 \) for all \( u,v \).
- **repeat until** value\( (f) = r \)
  - Find the shortest path \( P \) in \( G_f \) from \( s \) to \( t \)
  - Let \( q \) be the minimum residual capacity of an edge on \( P \).
  - Send \( \min(q,r - \text{value}(f)) \) additional units of flow across \( P \).
On the successive shortest paths algorithm

- May use exponential running time
- Assume G has no negative edge costs.
- Gives optimal answer.
  - Invariant: $f$ has minimum cost among all flows with value $\text{value}(f)$.

Suppose we obtain $f''$ from $f$ by sending across $P$.

Let $f'''$ be a minimum cost flow with same value as $f''$.

Write $f''' - f$ as weighted sum of paths from $s$ to $t$ in $G_f$ and circuits in $G_f$.

Argue that $\text{cost}(f'' - f) \leq \text{cost}(f''' - f)$, using that:
- $P$ is shortest path
- Circuits have non-negative costs, by optimality of $f$. 

Algorithms and Networks: Minimum Cost Flow
Finally

• More efficient algorithms exist
  – Some use scaling techniques:
    • Scaling on capacities
    • Scaling on costs