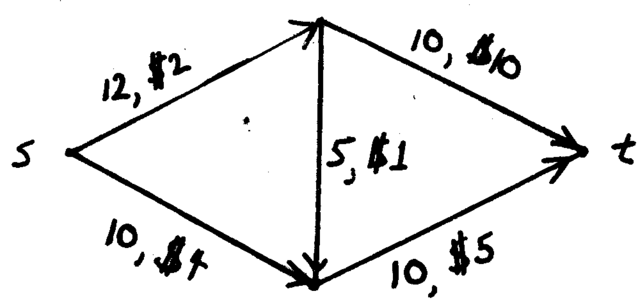
Minimum-Cost Network Flow

In addition to a capacity, each edge has a real-valued cost per unit of flow.

A minimum-cost (maximum) flow is a maximum flow whose total cost (sum of edge flows times edge costs) is minimum.

Problem: find a minimum-cost flow in a given network.



n = # vertices
m = # edges
U = max capacity (if integers)
C = max cost (if integers)

Two Naive Approaches

(1) Repeat: augment along a cheapest path in the residual network.

Each augmentation takes a shortest path computation.

Shortest paths can be found using Dijkstra's algorithm if costs are kept non-negative using price transformation (primal-dual method of linear programming).

Time: $O(nU(m+n\log n))$ (not polynomial)

(2) repeat $\left\{ \begin{array}{l} \text{In network of zero-cost residual edges,} \\ \quad \text{find a maximum flow.} \\ \text{Augment the flow and update the prices.} \end{array} \right.$

(find all paths of a given cost at once)

Time: $O(nC(nm\log(n^2/m)))$ (not polynomial)

$G = (V, E)$ symmetric directed graph

$(v, w) \in E$ iff $(w, v) \in E$

$|V| = n$, $|E| = m$, $m \geq n \geq 2$ $E(v) = \{w \mid (v, w) \in E\}$

arc capacities $u(v, w) : (v, w) \in E$

arc costs $c(v, w) : (v, w) \in E$

cost function is antisymmetric: $c(v, w) = -c(w, v)$

(multiple edges allow more general costs)

Circulation $f : E \rightarrow R$

$f(v, w) \leq u(v, w)$ $\forall (v, w) \in E$ capacity constraints

$f(v, w) = -f(w, v)$ $\forall (v, w) \in E$ flow antisymmetry

$\sum_{w \in E(v)} f(v, w) = 0$ $\forall v \in V$ flow conservation

Cost of f: $\frac{1}{2} \sum_{(v, w) \in E} f(v, w)\, c(v, w)$

Reformulated problem: Find a circulation of minimum cost: add a return arc from s to t of infinite capacity and large negative cost $(-nC)$.

residual capacity $u_f(v,w) = u(v,w) - f(v,w)$ or $f(w,v)$

residual arc (v,w): $u_f(v,w) > 0$

residual cycle: a (simple) cycle of residual arcs

<u>length</u> of cycle = number of arcs, $\ell(\Gamma)$

<u>cost</u> of cycle = sum of arc costs = $c(\Gamma)$

<u>mean</u> <u>cost</u> of cycle = $c(\Gamma)/\ell(\Gamma)$

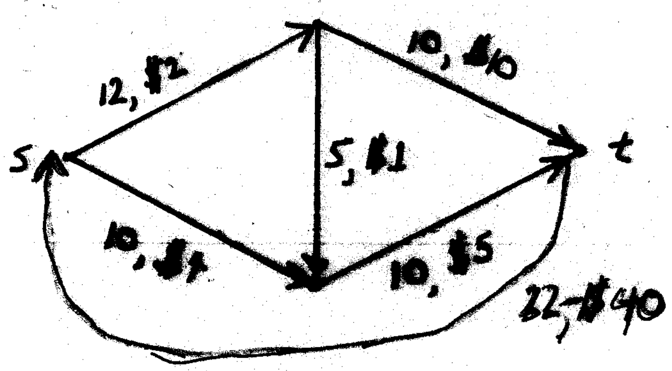
<u>negative</u> cycle: $c(\Gamma) < 0$

Minimum-Cost Network Flow

In addition to a capacity, each edge has a real-valued cost per unit of flow.

A minimum-cost (maximum) flow is a maximum flow whose total cost (sum of edge flows times edge costs) is minimum.

Problem: find a minimum-cost flow in a given network.



n = # vertices

m = # edges

U = max capacity (if integers)

C = max cost (if integers)

Theorem (Busacker and Saaty, 1965): A circulation f is minimum-cost iff there is no negative residual cycle.

Algorithm (Klein, 1967)

0. Find any circulation f (by a max flow computation)

1. While $\exists$ negative cycle Γ, cancel Γ by increasing f on all arcs of Γ by $\min\{u_f(v,w) : (v,w) \in \Gamma\}$.

#iterations can be exponential (or infinite)

How to choose cycles for canceling to minimize #iterations, running time?

minimum cost?

minimum length?

maximum length?

maximum capacity?

maximum cost decrease?

Our Results

Minimum-mean cycle canceling: Always cancel a cycle of minimum mean cost.

<u>Theorem</u>: # cancellations $= O(nm^2 \log n)$. If costs are integers of maximum magnitude C, # cancellations $= O(nm \log(nC))$.

Time to find a minimum mean cycle $= O(nm)$ (Karp, 1978)

A variant of this approach gives a "practical" algorithm with a running time of $O(nm \log n \min\{\log(nC), m \log n\})$.

Scaling Approach

Add a bit of precision (to capacities or costs) at a time. Start with exact solution to approximate problem. Use it as an approximate solution to a more exact problem. Improve the solution to an exact solution.

Scaling capacities (Edmonds & Karp; Rock)

For each bit of capacity precision, must find $O(m)$ cheapest paths.
Find a cheapest path using Dijkstra's shortest path algorithm; use price transform.

Time: $O(m \log U (m + n \log n))$

Scaling costs (Rock; Bland & Jensen)

For each bit of cost precision, must find $O(n)$ maximum flows.

Time: $O(n \log C (nm \log(n^2/m)))$