COS 423 Lecture 22
Minimum-cost matchings and flows

Given a directed graph with source s, sink t, arc capacities $u(v, w)$, and antisymmetric arc costs $c(v, w)$ ($= -c(w, v)$), the *cost* of a pseudoflow f is $\Sigma\{f(v, w)c(v, w) \mid (v, w) \in E\}/2$. (The factor of 2 is to compensate for the double-counting due to negative flows.)

Goal: Find a maximum flow of minimum cost.

Equivalent problems

G has no source, no sink. A *circulation* is a pseudoflow such that $e(v) = 0$ for all v.

Goal: Find a minimum-cost circulation

G is bipartite, all arcs from Y to X have capacity 0, all arcs from X to Y have capacity ∞ (or sufficiently large), each vertex $x \in X$ has a *supply* $s(x)$, each vertex $y \in Y$ has a *demand* $d(x)$, all supplies and demands ≥ 0

Transportation problem: Find a pseudoflow of minimum cost such that $e(x) \geq -s(x)$ if $x \in X$, $e(y) = d(y)$ if $y \in Y$

From flow to circulation: Add arc from t to s of capacity $\Sigma u(s, v)$ and cost $-\Sigma|c(v, w)|$. Minimum-cost circulation gives a minimum-cost maximum flow on original graph

From circulation to flow: Saturate all negative-cost arcs to give a pseudoflow. Give each arc a capacity = residual capacity. Add a source s with an arc to each vertex of positive excess, of capacity = excess, and a sink t with an arc from each vertex of negative excess, of capacity = – excess. Minimum-cost maximum flow gives a minimum-cost circulation on original graph

From transportation problem to flow: Add a source with an arc to each $x \in X$, of capacity $s(x)$; add a sink with an arc from each $y \in Y$, of capacity $d(y)$. Minimum-cost maximum flow gives solution to transportation problem on original graph if value $\Sigma d(y)$; if not, infeasible

From circulation to transportation problem: Construct a graph with one vertex v for each vertex v in original graph of supply $\Sigma u(v, w)$ and one vertex (v, w) of demand $u(v, w)$ for each arc (v, w) in original graph, and arcs $(v, (v, w))$, $(w, (v, w))$, having costs 0 and $c(v, w)$, respectively. Solution to transportation gives minimum-cost circulation in original graph: flow on (v, w) = flow on $(w, (v, w))$

Negative-cost cycles

Theorem: A circulation has minimum cost iff there is no negative-cost residual cycle.

Proof: If there is such a cycle, the circulation is not minimum-cost. If f is a circulation and f' is a circulation of smaller cost, consider f' – f. This is a circulation, and it can be decomposed into cycles of flow, each of which is a residual cycle for f and at least one of which has negative cost.

Theorem: Let f be a flow of value F. Among flows of value F, f has minimum cost iff f has no negative-cost residual cycle.

Proof: Like the proof of the previous theorem.

Cycle-canceling to find a minimum-cost circulation

Start with the zero flow. While there is a negative-cost residual cycle, choose such a cycle and send as much flow around it as possible.

Termination?

#iterations?

Minimum-cost augmentations to find a minimum-cost flow

Requires that all arcs of positive capacity have non-negative cost

Start with the zero flow. Repeatedly augment along a minimum-cost augmenting path. Each flow will have minimum cost among flows of the same value; no negative-cost residual cycle will ever exist; cost of augmenting path never decreases

Termination?

#iterations?

Minimum-cost augmentation via Dijkstra's algorithm

Maintain a price $p(v)$ for each vertex v, initially 0.

Define the *reduced cost* $c^*(v, w)$ of arc (v, w) to be $c(v, w) + p(v) - p(w)$

Compute cheapest residual paths using reduced costs; subtract costs of cheapest paths from prices; augment

All reduced costs remain non-negative: arcs on augmenting path have reduced cost 0 after price update

$O(n\lg n + m)$ time per augmentation, vs. $O(nm)$, polynomial-time if all capacities are polynomial in n.

For minimum-cost maximum-size bipartite matching, this gives an $O(n^2\lg n + nm)$-time algorithm: transportation problem with supplies and demands 1, maximize demand satisfaction at minimum cost. Flow remains integral, at most n augmentations.

Can augment along all paths of a given cost at once: compute residual graph containing all arcs on minimum-cost augmenting paths (zero-reduced-cost arcs), find a maximum flow on this graph, add to current flow

Polynomial-time if all arc costs are polynomial in *n*.

But what if costs and capacities are huge?

Polynomial time via cycle-canceling

Cancel & tighten algorithm: Maintain prices but allow reduced costs to be slightly negative ($\geq -\varepsilon$) Cancel cycles all of whose arcs have negative reduced cost. Once no such cycle exists, change prices to reduce negativity

Canceling a cycle can add new residual arcs, but they all have positive reduced cost. Each cancelation saturates at least one arc of negative reduced cost → $\leq m$ cancellations between price changes

Once no cancellation is possible, subgraph S of arcs having negative reduced cost is acyclic: can reduce maximum negativity by adjusting prices appropriately

Price update (*tighten* step): Compute, for each vertex v, the maximum number of arcs $h(v)$ on a path to v in S: $h(v) = 0$ if no incoming arcs, $\max\{h(x) \mid (x, v) \text{ in } S\} + 1$ if some incoming arc

For each vertex v, set $p(v) \leftarrow p(v) - h(v)\varepsilon/n$. This increases the cost of every negative-reduced-cost arc by at least ε/n, and decreases the cost of every non-negative-reduced-cost arc by at most $(n - 1)\varepsilon/n$ $(0 \leq h(v) \leq n - 1)$

$\rightarrow$ After the price update, we can replace ε by $\varepsilon(1 - 1/n)$

After n tighten steps, ε has decreased by a constant factor

Assume integer prices. Start with all prices zero and $\varepsilon = -C$, where C is the negative of the most-negative arc price. After $O(n \lg(nC))$ tightenings, $\varepsilon < 1/n$. Then we are done!

#cancellations: $\leq m$ per tightening = $O(nm \lg(nC))$

Lemma: If every residual arc has reduced cost greater than $–1/n$, then there is no negative-cost residual cycle

Proof: The reduced cost of a cycle equals its original cost. Since the reduced cost of a cycle must be greater than –1 and integral, it must be non-negative.

Finding cycles to cancel

Do a DFS along negative-reduced-cost arcs. If no negative-cost arc out of a vertex, delete the vertex. If a back arc is traversed, cancel the corresponding cycle and start over. After at most n arc traversals and O(n) time, either a cycle is canceled or a vertex is deleted

→ $O(nm)$ time for all cancellations between two tightenings

→ $O(n^2 m \lg(nC))$ total time

With the right data structure (link/cut trees), can reduce the running time to $O(nm \lg n \lg(nC))$

Strongly polynomial bound on cancellations?

Given a circulation, how to choose prices to minimize ε, what is minimum ε?

Mean cycle cost: total cost/#arcs

Lemma: Minimum ε = – minimum mean cycle cost

Proof: Exercise

Can compute minimum mean cycle cost and corresponding prices in O(nm) time: do this every n tightening steps

Then every O($n$lg$n$) tightening steps, one more arc becomes *fixed*: in the future, its flow never changes

→ O(nm^2log$n$) cancellations

Conceptually simple, strongly polynomial algorithm: Repeatedly cancel a cycle of minimum mean cost, until there is no negative-cost cycle