The Design and Analysis of Algorithms

Lecture 15 Network Flow II

Zhenbo Wang

Department of Mathematical Sciences, Tsinghua University

Content

Capacity Scaling Algorithm

Bipartite Matching

Disjoint Paths

Capacity Scaling Algorithm

- *Intuition*. Choosing path with highest bottleneck capacity.
- Maintain scaling parameter Δ.
- Let $G_f(\Delta)$ be the subgraph of the residual graph consisting of only arcs with capacity at least Δ.

Capacity Scaling Algorithm

$CAPACITY-SCALING(G, s, t, c)$

$\Delta \leftarrow$ largest power of $2 \leq C$.
for edge $e \in E$ **do**
    $f(e) \leftarrow 0$.
end for
while $\Delta \geq 1$ **do**
    $G_f(\Delta) \leftarrow \Delta-$residual graph.
    while there exists an augmenting path P in $G_f(\Delta)$ **do**
        $f \leftarrow AUGMENT(f, c, P)$.
    end while
    $\Delta \leftarrow \Delta/2$.
end while
return f.

Correctness

- *Assumption*. All edge capacities are integers between 1 and C.
- *Integrality invariant*. All flow and residual capacity values are integral.

Theorem 1

If capacity-scaling algorithm terminates, then f is a max-flow.

Pf. When $\Delta = 1 \Rightarrow Gf(\Delta) = Gf$.

Upon termination of $\Delta = 1$ phase, there are no augmenting paths. $\square$

Running Time

Lemma 2

The outer while loop repeats $O(\log C)$ times.

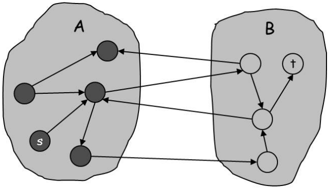
Lemma 3

Let f be the flow at the end of a Δ–scaling phase. Then, the value of the max-flow $\leq val(f) + m\Delta$.

Pf. Choose A to be the set of nodes reachable from s in $G_f(\Delta)$.

$s \in A$ and $t \notin A$.

$$\begin{aligned} v(f) &= \sum_{\text{e out ot A}} f(e) - \sum_{\text{e in to A}} f(e) \\ &\geq \sum_{\text{e out ot A}} (c(e) - \Delta) - \sum_{\text{e in to A}} \Delta \\ &\geq \sum_{\text{e out ot A}} c(e) - m\Delta \\ &= cap(A, B) - m\Delta. \quad \square \end{aligned}$$



original network

Running Time

Lemma 4

There are at most $2m$ augmentations per scaling phase.

Pf. Let f be the flow at the end of the previous scaling phase.

Lemma 3 $\Rightarrow$ the value of the max-flow $\leq val(f) + 2m\Delta$.

Each augmentation in a Δ–phase increases $val(f)$ by at least Δ $\square$.

Theorem 5

The scaling max-flow algorithm finds a max flow in $O(m \log C)$ augmentations. It can be implemented to run in $O(m^2 \log C)$ time.

Pf. Follows from Lemma 2 and lemma 4. $\square$

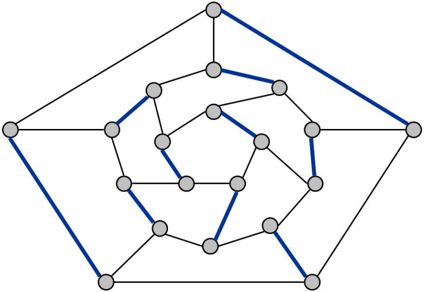
Extensions

- Shortest augmenting path algorithm: $O(n^2 m)$
- Preflow-push algorithm: $O(n^2 m)$
- Highest-label preflow-push algorithm: $O(n^2 m^{1/2})$.
- $\cdots$

Matching

Def. Given an undirected graph $G = (V, E)$ a subset of edges $M \subseteq E$ is a *matching* if each node appears in at most one edge in M.

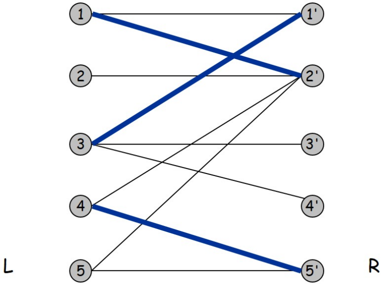
- *Max matching*. Given a graph, find a max cardinality matching.



Bipartite matching

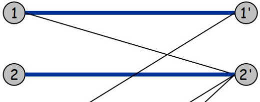
Def. A graph G is *bipartite* if the nodes can be partitioned into two subsets L and R such that every edge connects a node in L to one in R.

- *Bipartite matching*. Given a bipartite graph $G = (L \cup R, E)$, find a max cardinality matching.



matching

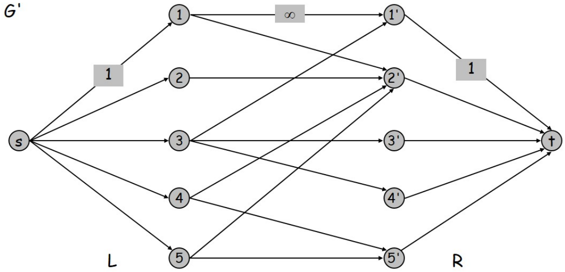
1-2', 3-1', 4-5'



max matching

Bipartite matching: max flow formulation

- Create digraph $G' = (L \cup R \cup \{s, t\}, E')$.
- Direct all edges from L to R, and assign infinite (or unit) capacity.
- Add source s, and unit capacity edges from s to each node in L.
- Add sink t, and unit capacity edges from each node in R to t.



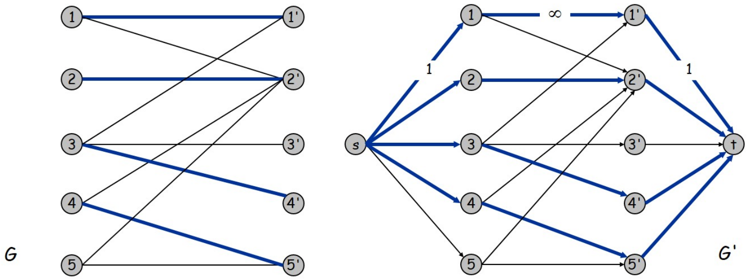
Max flow formulation: proof of correctness

Theorem

Max cardinality of a matching in G = value of max flow in G'.

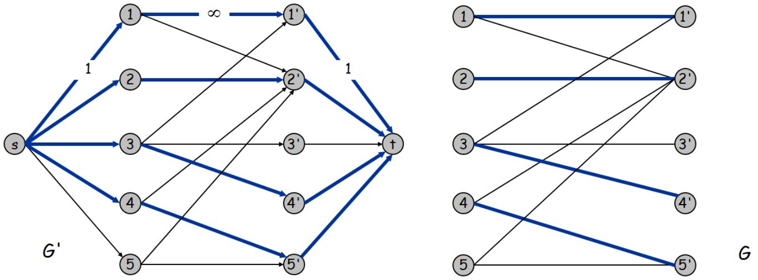
Pf.

- Given a max matching M of cardinality k.
- Consider flow f that sends 1 unit along each of k paths.
- f is a flow, and has value k.



Max flow formulation: proof of correctness

- Let f be a max flow in G' of value k.
- Integrality theorem $\Rightarrow$ k is integral and can assume f is 0-1.
- Consider M = set of edges from L to R with $f(e) = 1$.
- each node in L and R participates in at most one edge in M.
- $|M| = k$: consider cut $(L \cup s, R \cup t)$. $\square$



Bipartite Matching: Running Time

Which max flow algorithm to use for bipartite matching?

- Generic augmenting path: $O(m\,val(f^*)) = O(mn)$.
- Capacity scaling: $O(m^2 \log C) = O(m^2)$.
- Shortest augmenting path: $O(mn^{\frac{1}{2}})$

Non-bipartite matching.

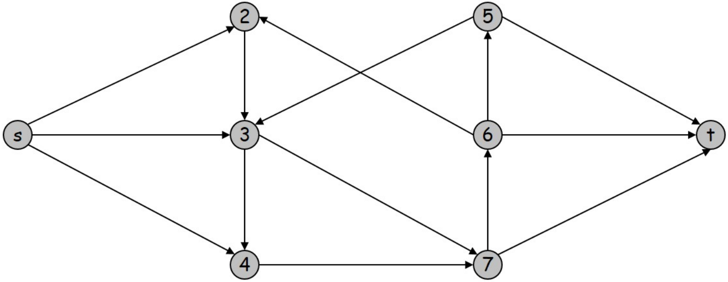
- Blossom algorithm: $O(n^4)$.
- Best known: $O(mn^{\frac{1}{2}})$.

Edge-disjoint paths

Def. Two paths are *edge-disjoint* if they have no edge in common.

- *Disjoint path problem.* Given a digraph $G = (V, E)$ and two nodes s and t, find the max number of edge-disjoint $s \to t$ paths.

Ex. Communication networks.



Edge-disjoint paths

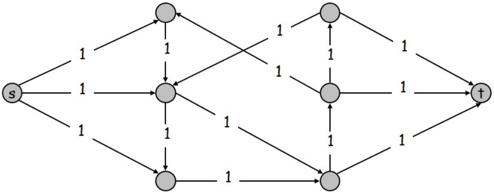
- *Max flow formulation.* Assign unit capacity to every edge.

Theorem 6

Max number edge-disjoint $s \to t$ paths equals value of max flow.

Pf. $\leq$

- Suppose there are k edge-disjoint $s \to t$ paths $P_1, \cdots, P_k$.
- Set $f(e) = 1$ if e participates in some path P_j ; else set $f(e) = 0$.
- Since paths are edge-disjoint, f is a flow of value k.



Edge-disjoint paths–Con't

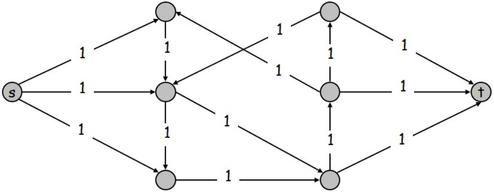
Pf. $\geq$

Suppose max flow value is k.

Integrality theorem $\Rightarrow$ there exists 0-1 flow f of value k.

- By conservation, there exists an edge (u, v) with $f(u, v) = 1$.
- Continue until reach t, always choosing a new edge.

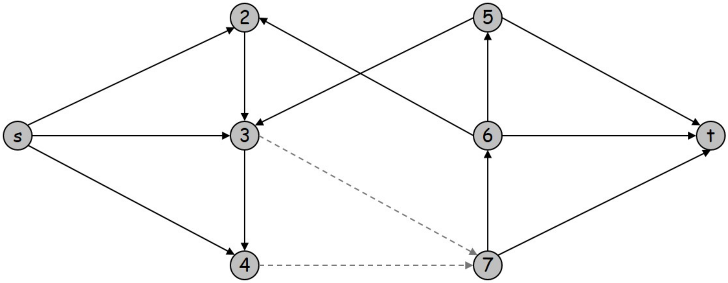
Produces k (not necessarily simple) edge-disjoint paths. $\square$



Network connectivity

Def. A set of edges $F \subseteq E$ *disconnects* t *from* s if every $s \to t$ path uses at least one edge in F.

- *Network connectivity.* Given a digraph $G = (V, E)$ and two nodes s and t, find min number of edges whose removal disconnects t from s.



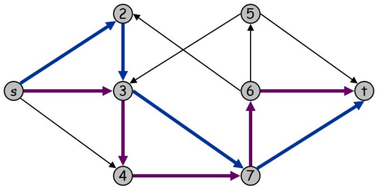
Menger's theorem

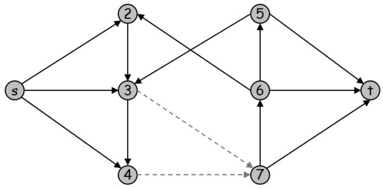
Theorem 7 (Menger 1927)

The max number of edge-disjoint $s \to t$ paths is equal to the min number of edges whose removal disconnects t from s.

Pf. $\leq$

- Suppose the removal of $F \subseteq E$ disconnects t from s, and $|F| = k$.
- Every $s \to t$ path uses at least one edge in F.
- Hence, the number of edge-disjoint paths is $\leq k$.





Menger's theorem–Con't

Pf. $\geq$

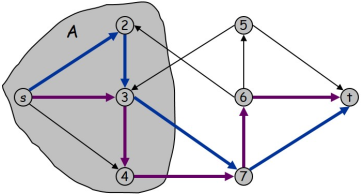
Suppose max number of edge-disjoint paths is k.

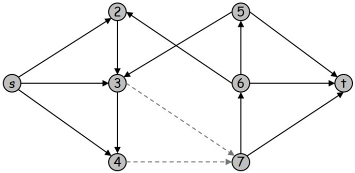
Then value of max flow = k.

Max-flow min-cut theorem $\Rightarrow$ there exists a cut (A, B) of capacity k.

Let F be set of edges going from A to B.

$|F| = k$ and disconnects t from s. $\square$





Homework

- Read Chapter 7 of the textbook.
- Exercises 12 & 18 in Chapter 7.