Maximum Flow Problems II

AU4606: Network Optimization

AI4702: Network Intelligence and Optimization

Xiaoming Duan
Department of Automation
Shanghai Jiao Tong University

October 9, 2023

Last lecture

- Maximum flow problems: formulation
  - What is a maximum flow problem?
  - An application
- Maximum flow problems: preliminaries
  - Residual graphs
  - s-t cut
- Augmenting path algorithms
  - Generic augmenting path algorithms
  - Theoretical properties

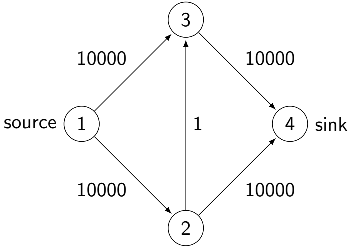
Today

1. Issues with generic augmenting path algorithms
   - A bad example
   - Roadmap for upcoming discussions
2. Most improving augmenting path algorithm
3. Capacity scaling algorithm
4. Shortest augmenting path algorithm

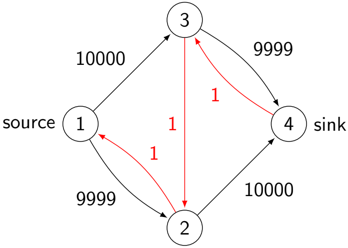
Today

1. Issues with generic augmenting path algorithms
   - A bad example
   - Roadmap for upcoming discussions
2. Most improving augmenting path algorithm
3. Capacity scaling algorithm
4. Shortest augmenting path algorithm

Generic augmenting path algorithm: bad example



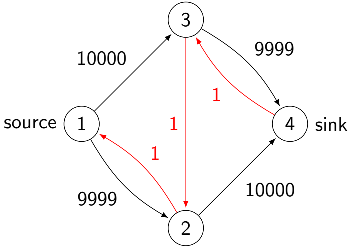
(a) A maximum flow problem



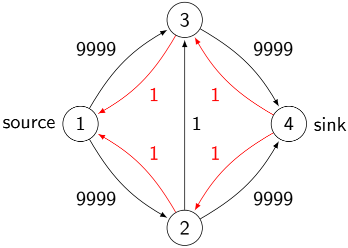
(b) Residual graph 1

An augmenting path $1 - 2 - 3 - 4$ exists and 1 unit flow can be sent

Generic augmenting path algorithm: bad example



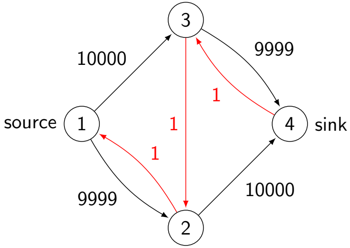
(a) Residual graph 1



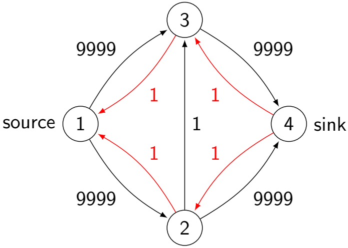
(b) Residual graph 2

An augmenting path $1 - 3 - 2 - 4$ exists and 1 unit flow can be sent

Generic augmenting path algorithm: bad example



(a) Residual graph 1

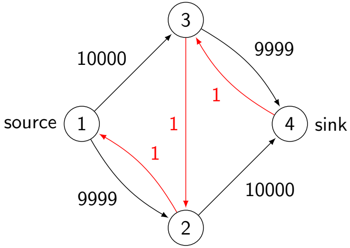


(b) Residual graph 2

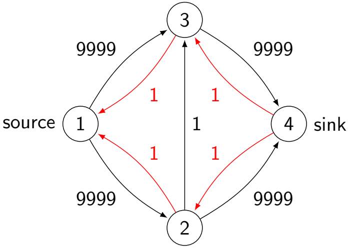
An augmenting path $1 - 3 - 2 - 4$ exists and 1 unit flow can be sent

An augmenting path $1 - 2 - 3 - 4$ exists and 1 unit flow can be sent

Generic augmenting path algorithm: bad example



(a) Residual graph 1

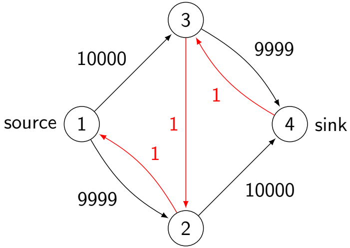


(b) Residual graph 2

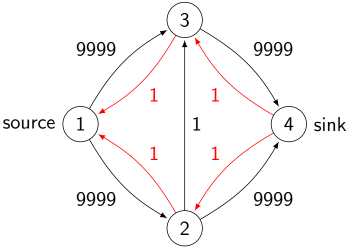
An augmenting path $1 - 3 - 2 - 4$ exists and 1 unit flow can be sent

An augmenting path $1 - 2 - 3 - 4$ exists and 1 unit flow can be sent
An augmenting path $1 - 3 - 2 - 4$ exists and 1 unit flow can be sent

Generic augmenting path algorithm: bad example



(a) Residual graph 1



(b) Residual graph 2

An augmenting path $1 - 3 - 2 - 4$ exists and 1 unit flow can be sent

An augmenting path $1 - 2 - 3 - 4$ exists and 1 unit flow can be sent
An augmenting path $1 - 3 - 2 - 4$ exists and 1 unit flow can be sent
An augmenting path $1 - 2 - 3 - 4$ exists and 1 unit flow can be sent
...

Generic augmenting path algorithm

Algorithm Augmenting path

1: $f \leftarrow 0$
2: **while** $G(f)$ contains a directed path from s to t **do**
3: Identify an augmenting path P from s to t
4: $\delta(P) = \min\{r_{ij}, (i,j) \in P\}$
5: Augment $\delta(P)$ units of flow along P and update $G(f)$
6: **end while**

Can we pick paths more wisely?

1. Choose path with maximum residual capacity (most improving)
2. Choose path with sufficiently large residual capacity (capacity scaling)
3. Choose shortest augmenting path (Edmonds–Karp)

Today

1. Issues with generic augmenting path algorithms
   - A bad example
   - Roadmap for upcoming discussions

2. Most improving augmenting path algorithm

3. Capacity scaling algorithm

4. Shortest augmenting path algorithm

Most improving augmenting path algorithm

Recall residual capacity $\delta(P)$ of an augmenting path P in a residual graph:

$$\delta(P) = \min\{r_{ij}, (i,j) \in P\}$$

Algorithm Most improving augmenting path

1: $f \leftarrow 0$
2: **while** $G(f)$ contains a directed path from s to t **do**
3: Identify an augmenting path P from s to t that has maximum $\delta(P)$
4: Augment $\delta(P)$ units of flow along P and update $G(f)$
5: **end while**

Detour: geometric improvement approach

Intuition:

- If making sufficient progress each step, algorithm converges fast

Geometric improvement

Let z^k be the integer objective value of a maximization problem at the k-th iteration of an algorithm with $z^0 = 0$, and $z^* \leq H$ be the integer optimal value. If

$$z^{k+1} - z^k \geq \alpha(z^* - z^k),$$

for some $0 < \alpha < 1$, then the algorithm terminates in $O(\frac{\log H}{\alpha})$ iterations

Detour: geometric improvement approach

Intuition:

- If making sufficient progress each step, algorithm converges fast

Geometric improvement

Let z^k be the integer objective value of a maximization problem at the k-th iteration of an algorithm with $z^0 = 0$, and $z^* \leq H$ be the integer optimal value. If

$$z^{k+1} - z^k \geq \alpha(z^* - z^k),$$

for some $0 < \alpha < 1$, then the algorithm terminates in $O(\frac{\log H}{\alpha})$ iterations

Show that most improving augmenting path algorithm does exactly this

Flows in the original graph and residual graph

In most improving path algorithm

1. The per-step improvement is residual capacity of most improving path
2. Can we lower bound this improvement by some fractional of the distance between maximum and current flows?

Relationship between flows in original and residual graphs

Let f and f^* be a flow and a maximum flow in G, respectively. Then the maximum flow in $G(f)$ is equal to $v(f^*) - v(f)$.

Proof sketch

- Given f and f^*, define f' in $G(f)$
- Show that f' is indeed a flow in $G(f)$
- Show that $v(f') = v(f^*) - v(f)$
- Show that f' is a maximum flow

Flow decomposition lemma

For two flows f' and f'', we write $f = f' \pm f''$ if

$$f_{ij} = f'_{ij} \pm f''_{ij}, \quad \text{for all } (i,j) \in A$$

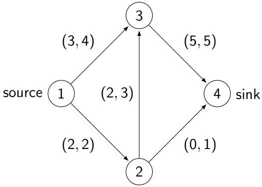
Flow decomposition lemma

Given any positive flow f, there exist flows $f^1, \ldots, f^\ell$ for some $\ell \leq m$ such that $f = \sum_{i=1}^{\ell} f^i$, where for each i, the arcs of f^i with positive flow form either a directed path from s to t or a directed cycle.

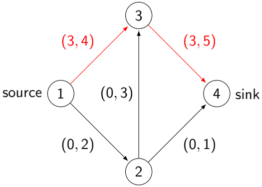
Flow decomposition lemma: an example

Flow decomposition lemma

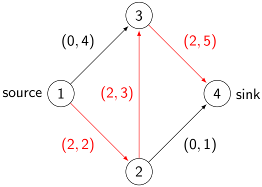
Given any positive flow f, there exist flows $f^1, \ldots, f^\ell$ for some $\ell \leq m$ such that $f = \sum_{i=1}^{\ell} f^i$, where for each i, the arcs of f^i with positive flow form either a directed path from s to t or a directed cycle.



(a) Flow f



(b) Flow f^1



(c) Flow f^2

$$f = f^1 + f^2$$

Flows in the original graph and residual graph

Relationship between flows in original and residual graphs

Let f and f^* be a flow and a maximum flow in G, respectively. Then the maximum flow in $G(f)$ is equal to $v(f^*) - v(f)$.

Flow decomposition lemma

Given any positive flow f, there exist flows $f^1, \ldots, f^\ell$ for some $\ell \leq m$ such that $f = \sum_{i=1}^{\ell} f^i$, where for each i, the arcs of f^i with positive flow form either a directed path from s to t or a directed cycle.

The above two imply immediately:

Residual capacity of a most improving path

Let f and f^* be a flow and a maximum flow in G, respectively. Then the residual capacity of a most improving path in $G(f)$ is at least $\frac{1}{m}(v(f^*) - v(f))$.

Algorithm Most improving augmenting path

1: $f \leftarrow 0$
2: **while** $G(f)$ contains a directed path from s to t **do**
3: Identify an augmenting path P from s to t that has maximum $\delta(P)$
4: Augment $\delta(P)$ units of flow along P and update $G(f)$
5: **end while**

Number of iterations (by the geometric improvement approach)

If capacities are integers, the most improving augmenting path algorithm computes a maximum flow in $O(m \log(mU))$ iterations.

Complexity of most improving augmenting path algorithm

Algorithm Most improving augmenting path

1: $f \leftarrow 0$
2: **while** $G(f)$ contains a directed path from s to t **do**
3: Identify an augmenting path P from s to t that has maximum $\delta(P)$
4: Augment $\delta(P)$ units of flow along P and update $G(f)$
5: **end while**

Number of iterations (by the geometric improvement approach)

If capacities are integers, the most improving augmenting path algorithm computes a maximum flow in $O(m \log(mU))$ iterations.

How much time needed to find a most improving augmenting path?

- Sort arcs with residual capacity in descending order: $O(m \log m)$
- Add arc to graph one and one and check if a path exists: $O(m^2)$

Complexity of most improving augmenting path algorithm

Algorithm Most improving augmenting path

1: $f \leftarrow 0$
2: **while** $G(f)$ contains a directed path from s to t **do**
3: Identify an augmenting path P from s to t that has maximum $\delta(P)$
4: Augment $\delta(P)$ units of flow along P and update $G(f)$
5: **end while**

Number of iterations (by the geometric improvement approach)

If capacities are integers, the most improving augmenting path algorithm computes a maximum flow in $O(m \log(mU))$ iterations.

How much time needed to find a most improving augmenting path?

- Sort arcs with residual capacity in descending order: $O(m \log m)$
- Add arc to graph one and one and check if a path exists: $O(m^2)$

Complexity

For integer capacities, most improving augmenting path algorithm returns a maximum flow in $O(m^3 \log(mU))$ $(O(m \log(mU)(m + n \log n)))$.

Today

1. Issues with generic augmenting path algorithms
   - A bad example
   - Roadmap for upcoming discussions
2. Most improving augmenting path algorithm
3. Capacity scaling algorithm
4. Shortest augmenting path algorithm

Capacity scaling algorithm

- Finding a most improving augmenting path is time-consuming
- How about finding a sufficiently improving path?

The idea is as follows

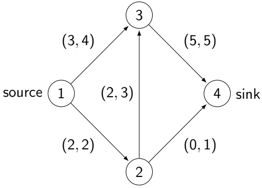
1. Set a large scaling parameter Δ
2. Keep all arcs in residual graph that have residual capacity at least Δ
3. If no augmenting path exists, reduce Δ to $\frac{\Delta}{2}$
4. Repeat from 2

Δ-residual graph

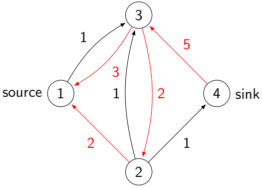
Δ-residual graph

The Δ-residual graph $G(f, \Delta)$ with respect to a flow f is a subgraph of $G(f)$ that contains all arcs with residual capacity at least Δ.

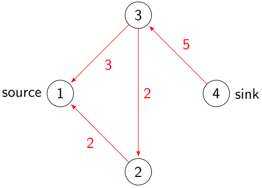
Note: $G(f, 1) = G(f)$



(a) Graph with flow f



(b) Residual graph $G(f)$



(c) 2-residual graph $G(f, 2)$

Capacity scaling algorithm

Algorithm Capacity scaling algorithm

$f \leftarrow 0$
$\Delta \leftarrow 2^{\lfloor \log_2 U \rfloor}$
while $\Delta \geq 1$ **do**
    while $G(f, \Delta)$ contains a directed path from s to t **do**
        Identify an augmenting path P from s to t in $G(f, \Delta)$
        $\delta(P) = \min\{r_{ij}, (i,j) \in P\}$
        Augment $\delta(P)$ units of flow along P and update $G(f, \Delta)$
    end while
    $\Delta \leftarrow \frac{\Delta}{2}$
end while

- The phase when Δ remains constant is called a scaling phase
- A total of at most $\lfloor \log_2 U \rfloor + 1$ scaling phases will be performed
- Since $G(f, 1) = G(f)$, we have generic augmenting path alg. lastly

Capacity scaling algorithm: complexity

- There are at most $O(\log U)$ iterations
- Need to bound the number of augmentations in each iteration next

Number of augmentations in scaling phases

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

Proof sketch

- Each augmentation in $G(f, \Delta)$ increases the flow by Δ
- Are there bounds on the maximum flow in $G(f, \Delta)$?

Complexity of capacity scaling

If the capacities are integers, the capacity scaling algorithm computes a maximum flow in $O(m^2 \log U)$

Today

1. Issues with generic augmenting path algorithms
   - A bad example
   - Roadmap for upcoming discussions
2. Most improving augmenting path algorithm
3. Capacity scaling algorithm
4. Shortest augmenting path algorithm

Shortest augmenting path algorithm

Starting from generic augmenting path algorithm, we have tried

1. Choose path with maximum residual capacity (most improving)
   - $O(m^3 \log(mU))$
2. Choose path with large residual capacity (capacity scaling)
   - $O(m^2 \log U)$

Shortest augmenting path algorithm

Starting from generic augmenting path algorithm, we have tried

1. Choose path with maximum residual capacity (most improving)
   - $O(m^3 \log(mU))$
2. Choose path with large residual capacity (capacity scaling)
   - $O(m^2 \log U)$

Can we do even better?

Pushing flows along shortest paths

Shortest augmenting path algorithm

Algorithm Shortest augmenting path algorithm

$f \leftarrow 0$
while $G(f)$ contains a directed path from s to t **do**
    Identify a shortest augmenting path P from s to t
    $\delta(P) = \min\{r_{ij}, (i,j) \in P\}$
    Augment $\delta(P)$ units of flow along P and update $G(f)$
end while

- Also known as Edmonds–Karp algorithm

Shortest augmenting path algorithm: analysis

- Distance label $d(i)$: shortest distance from i to t in residual graph

Monotonicity of distance labels

Let $d(i)$ and $d'(i)$ be the distance labels of node i in the residual graph at the beginning and end of an iteration, respectively. Then $d'(i) \geq d(i)$.

Distance labels are nondecreasing during the execution of the algorithm

Saturation of arcs

A given arc $(i, j) \in A$ becomes saturated $O(n)$ times during the execution of the algorithm.

- Each augmentation saturates at least one arc
- An arc can be saturated $O(n)$ times
- A total of $O(mn)$ augmentations can occur

The shortest augmenting path algorithm runs in $O(m^2 n)$

Upcoming

Week 1-8 (AU4606 & AI4702):

- Introduction (1 lecture)
- Preparations (3 lectures)
  - basics of graph theory
  - algorithm complexity and data structure
  - graph search algorithm
- Shortest path problems (3 lectures)
- Maximum flow problems (this and next few lectures)
- Minimum cost flow problems (2 lectures)
- Introduction to multi-agent systems (1 lecture)
- Introduction to cloud networks (1 lecture)

Week 9-16 (AU4606):

- Simplex and network simplex methods (2 lectures)
- Global minimum cut problems (3 lectures)
- Minimum spanning tree problems (3 lectures)