

CMPSCI 611: Advanced Algorithms

Lecture 11 and 12: Network Flow

Andrew McGregor

Last Compiled: October 20, 2009

Outline

Network Flows

Definitions

Input:

- ▶ Directed Graph $G = (V, E)$
- ▶ Capacities $C(u, v) > 0$ for $(u, v) \in E$ and $C(u, v) = 0$ for $(u, v) \notin E$
- ▶ A source node s , and sink node t

Definitions

Input:

- ▶ Directed Graph $G = (V, E)$
- ▶ Capacities $C(u, v) > 0$ for $(u, v) \in E$ and $C(u, v) = 0$ for $(u, v) \notin E$
- ▶ A source node s , and sink node t

Output: A flow f from s to t where $f : V \times V \rightarrow \mathbb{R}$ satisfies

- ▶ Skew-symmetry: $\forall u, v \in V, f(u, v) = -f(v, u)$
- ▶ Conservation of Flow: $\forall v \in V - \{s, t\}, \sum_{u \in V} f(u, v) = 0$
- ▶ Capacity Constraints: $\forall u, v \in V, f(u, v) \leq C(u, v)$

Definitions

Input:

- ▶ Directed Graph $G = (V, E)$
- ▶ Capacities $C(u, v) > 0$ for $(u, v) \in E$ and $C(u, v) = 0$ for $(u, v) \notin E$
- ▶ A source node s , and sink node t

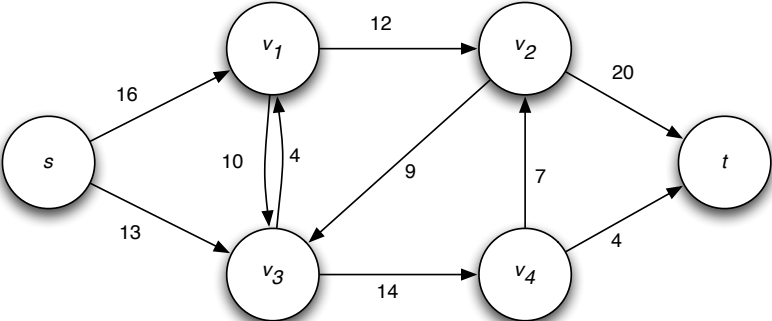
Output: A flow f from s to t where $f : V \times V \rightarrow \mathbb{R}$ satisfies

- ▶ Skew-symmetry: $\forall u, v \in V, f(u, v) = -f(v, u)$
- ▶ Conservation of Flow: $\forall v \in V - \{s, t\}, \sum_{u \in V} f(u, v) = 0$
- ▶ Capacity Constraints: $\forall u, v \in V, f(u, v) \leq C(u, v)$

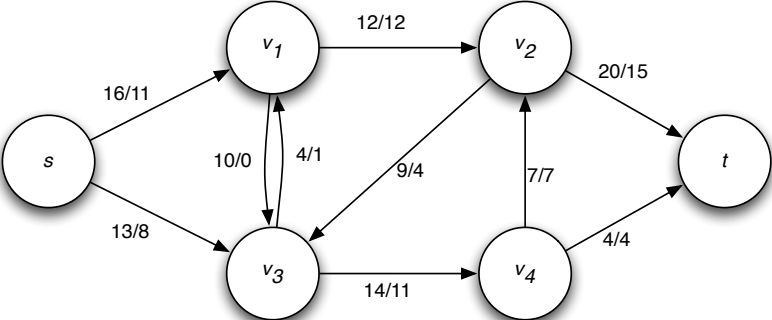
Goal: Maximize “size of the flow”, i.e., the total flow coming leaving s :

$$|f| = \sum_{v \in V} f(s, v)$$

Capacity



Capacity/Flow



Cut Definitions

Definition

An $s - t$ cut of G is a partition of the vertices into two sets A and B such that $s \in A$ and $t \in B$.

Cut Definitions

Definition

An $s - t$ cut of G is a partition of the vertices into two sets A and B such that $s \in A$ and $t \in B$.

Definition

The **capacity of a cut** (A, B) is

$$C(A, B) = \sum_{u \in A, v \in B} C(u, v)$$

Definition

The **flow across a cut** (A, B) is

$$f(A, B) = \sum_{u \in A, v \in B} f(u, v)$$

Cut Definitions

Definition

An $s - t$ cut of G is a partition of the vertices into two sets A and B such that $s \in A$ and $t \in B$.

Definition

The **capacity of a cut** (A, B) is

$$C(A, B) = \sum_{u \in A, v \in B} C(u, v)$$

Definition

The **flow across a cut** (A, B) is

$$f(A, B) = \sum_{u \in A, v \in B} f(u, v)$$

Note that because of capacity constraints: $f(A, B) \leq C(A, B)$

All cuts have same flow

Lemma

For any flow f : for all cuts (A, B) , $f(A, B)$ equals the total flow out of s

Proof.

- ▶ By induction on size of A where $s \in A$

All cuts have same flow

Lemma

For any flow f : for all cuts (A, B) , $f(A, B)$ equals the total flow out of s

Proof.

- ▶ By induction on size of A where $s \in A$
- ▶ **Base Case:**

All cuts have same flow

Lemma

For any flow f : for all cuts (A, B) , $f(A, B)$ equals the total flow out of s

Proof.

- ▶ By induction on size of A where $s \in A$
- ▶ **Base Case:** $A = \{s\}$ and $f(s, V - s) = |f|$

All cuts have same flow

Lemma

For any flow f : for all cuts (A, B) , $f(A, B)$ equals the total flow out of s

Proof.

- ▶ By induction on size of A where $s \in A$
- ▶ **Base Case:** $A = \{s\}$ and $f(s, V - s) = |f|$
- ▶ **Induction Hypothesis:** $f(A, B) = |f|$ for all A such that $|A| = k$

All cuts have same flow

Lemma

For any flow f : for all cuts (A, B) , $f(A, B)$ equals the total flow out of s

Proof.

- ▶ By induction on size of A where $s \in A$
- ▶ **Base Case:** $A = \{s\}$ and $f(s, V - s) = |f|$
- ▶ **Induction Hypothesis:** $f(A, B) = |f|$ for all A such that $|A| = k$
- ▶ Consider cut (A', B') where $|A'| = k + 1$.

All cuts have same flow

Lemma

For any flow f : for all cuts (A, B) , $f(A, B)$ equals the total flow out of s

Proof.

- ▶ By induction on size of A where $s \in A$
- ▶ **Base Case:** $A = \{s\}$ and $f(s, V - s) = |f|$
- ▶ **Induction Hypothesis:** $f(A, B) = |f|$ for all A such that $|A| = k$
- ▶ Consider cut (A', B') where $|A'| = k + 1$. Let $u \in A' - s$:

$$f(A' - u, B' + u) = f(A', B') + \sum_{v \in A'} f(v, u) - \sum_{v \in B'} f(u, v)$$

All cuts have same flow

Lemma

For any flow f : for all cuts (A, B) , $f(A, B)$ equals the total flow out of s

Proof.

- ▶ By induction on size of A where $s \in A$
- ▶ **Base Case:** $A = \{s\}$ and $f(s, V - s) = |f|$
- ▶ **Induction Hypothesis:** $f(A, B) = |f|$ for all A such that $|A| = k$
- ▶ Consider cut (A', B') where $|A'| = k + 1$. Let $u \in A' - s$:

$$f(A' - u, B' + u) = f(A', B') + \sum_{v \in A'} f(v, u) - \sum_{v \in B'} f(u, v)$$

- ▶ By skew-symmetry and conservation of flow

$$\sum_{v \in A'} f(v, u) - \sum_{v \in B'} f(u, v) = \sum_{v \in A'} f(v, u) + \sum_{v \in B'} f(v, u) = \sum_{v \in V} f(v, u) = 0$$

All cuts have same flow

Lemma

For any flow f : for all cuts (A, B) , $f(A, B)$ equals the total flow out of s

Proof.

- ▶ By induction on size of A where $s \in A$
- ▶ **Base Case:** $A = \{s\}$ and $f(s, V - s) = |f|$
- ▶ **Induction Hypothesis:** $f(A, B) = |f|$ for all A such that $|A| = k$
- ▶ Consider cut (A', B') where $|A'| = k + 1$. Let $u \in A' - s$:

$$f(A' - u, B' + u) = f(A', B') + \sum_{v \in A'} f(v, u) - \sum_{v \in B'} f(u, v)$$

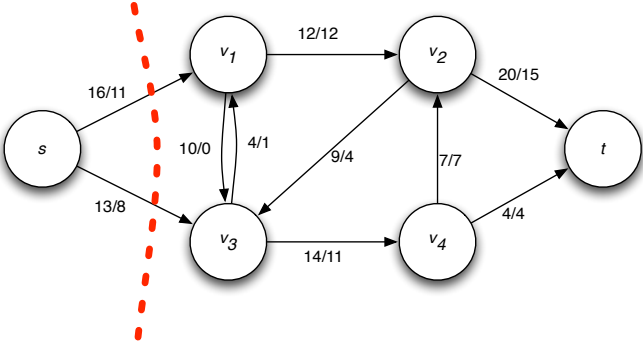
- ▶ By skew-symmetry and conservation of flow

$$\sum_{v \in A'} f(v, u) - \sum_{v \in B'} f(u, v) = \sum_{v \in A'} f(v, u) + \sum_{v \in B'} f(v, u) = \sum_{v \in V} f(v, u) = 0$$

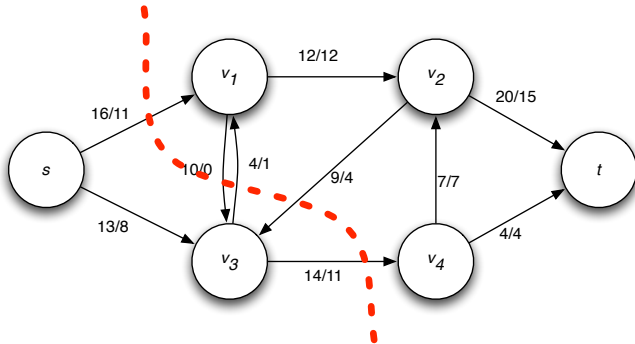
- ▶ Hence, $f(A', B') = f(A' - u, B' + u) = |f|$ by induction hypothesis.



First Cut



Second Cut



Max-Flow Min-Cut

Theorem (Max-Flow Min-Cut)

For any flow network and flow f , the following statements are equivalent:

- 1. f is a maximum flow.*
- 2. There exists an $s - t$ cut (A, B) such that $|f| = C(A, B)$*

Residual Networks and Augmenting Paths

Residual network encodes how you can change the flow between two nodes given the current flow and the capacity constraints.

Residual Networks and Augmenting Paths

Residual network encodes how you can change the flow between two nodes given the current flow and the capacity constraints.

Definition

Given a flow network $G = (V, E)$ and flow f in G , the **residual network** G_f is defined as

$$G_f = (V, E_f) \text{ where } E_f = \{(u, v) : C(u, v) - f(u, v) > 0\}$$

$$C_f(u, v) = C(u, v) - f(u, v)$$

Note that $(u, v) \in E_f$ implies either $C(u, v) > 0$ or $C(v, u) > 0$.

Residual Networks and Augmenting Paths

Residual network encodes how you can change the flow between two nodes given the current flow and the capacity constraints.

Definition

Given a flow network $G = (V, E)$ and flow f in G , the **residual network** G_f is defined as

$$G_f = (V, E_f) \text{ where } E_f = \{(u, v) : C(u, v) - f(u, v) > 0\}$$

$$C_f(u, v) = C(u, v) - f(u, v)$$

Note that $(u, v) \in E_f$ implies either $C(u, v) > 0$ or $C(v, u) > 0$.

Definition

An **augmenting path** for flow f is a path from s to t in graph G_f . The **bottleneck capacity** $b(p)$ is the minimum capacity in G_f of any edge of p .

Residual Networks and Augmenting Paths

Residual network encodes how you can change the flow between two nodes given the current flow and the capacity constraints.

Definition

Given a flow network $G = (V, E)$ and flow f in G , the **residual network** G_f is defined as

$$G_f = (V, E_f) \text{ where } E_f = \{(u, v) : C(u, v) - f(u, v) > 0\}$$

$$C_f(u, v) = C(u, v) - f(u, v)$$

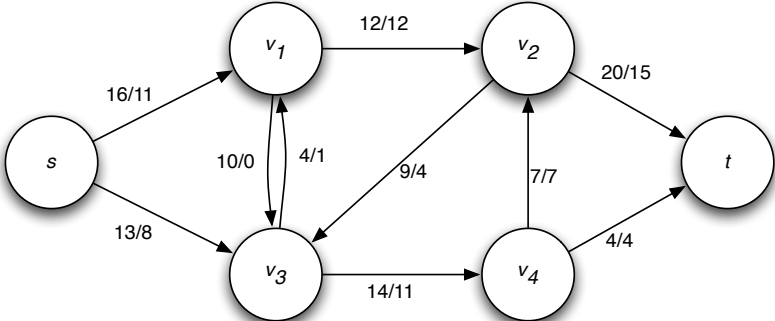
Note that $(u, v) \in E_f$ implies either $C(u, v) > 0$ or $C(v, u) > 0$.

Definition

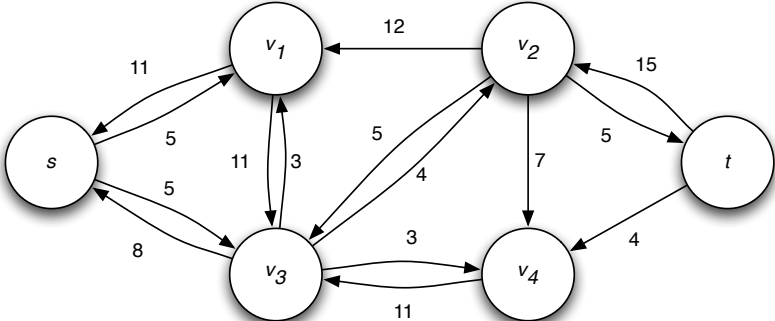
An **augmenting path** for flow f is a path from s to t in graph G_f . The **bottleneck capacity** $b(p)$ is the minimum capacity in G_f of any edge of p .

We can increase flow by $b(p)$ along an augmenting path.

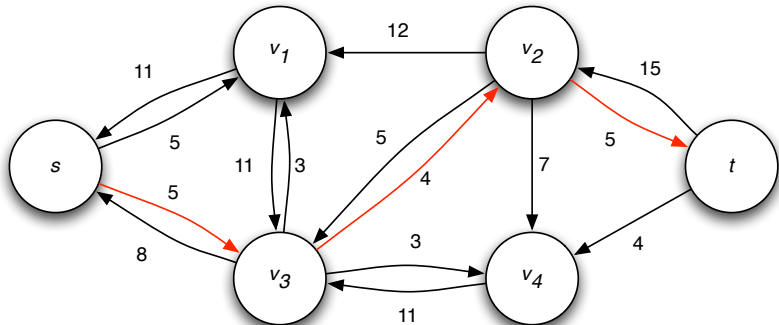
Capacity/Flow



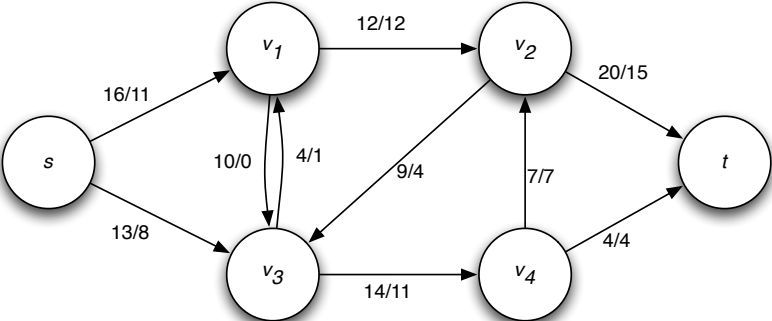
Residual



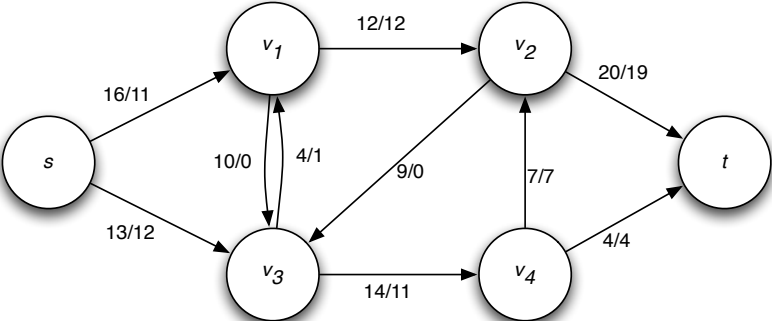
Augmenting Path



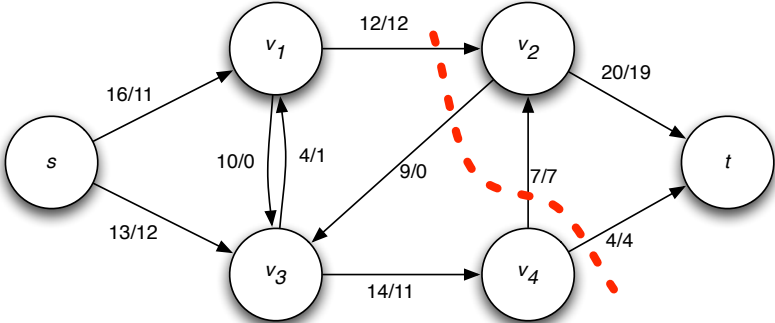
Old Flow



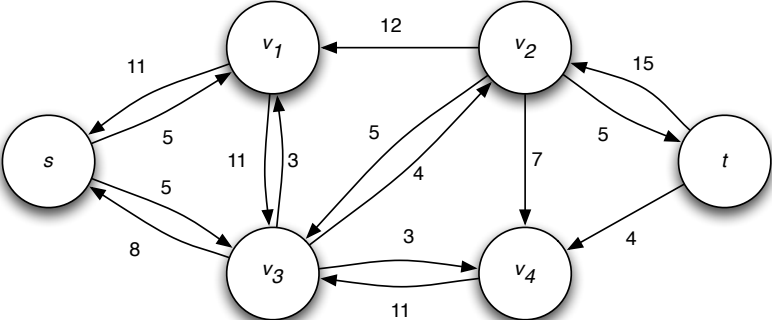
New Flow



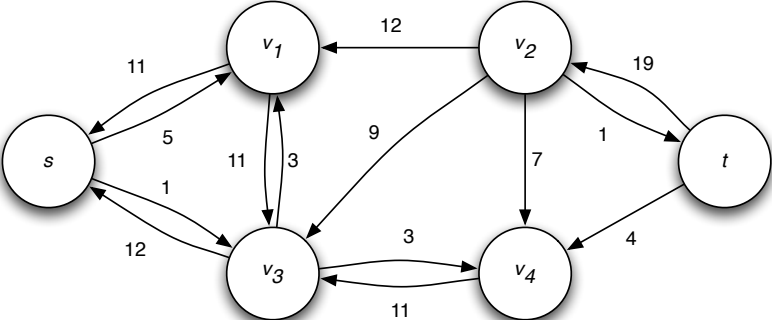
Min Capacity Cut Proves this is Optimal



Old Residual Graph



New Residual Graph



Max-Flow Min-Cut

Theorem (Max-Flow Min-Cut)

For any flow network and flow f , the following statements are equivalent:

- 1. f is a maximum flow.*
- 2. There exists an $s - t$ cut (A, B) with $|f| = C(A, B)$.*
- 3. There doesn't exist an augmenting path in G_f .*

Proof.

Max-Flow Min-Cut

Theorem (Max-Flow Min-Cut)

For any flow network and flow f , the following statements are equivalent:

- 1. f is a maximum flow.*
- 2. There exists an $s - t$ cut (A, B) with $|f| = C(A, B)$.*
- 3. There doesn't exist an augmenting path in G_f .*

Proof.

- ▶ $(2 \Rightarrow 1)$: Increasing flow, increases $f(A, B)$ which violates capacity constraints

Max-Flow Min-Cut

Theorem (Max-Flow Min-Cut)

For any flow network and flow f , the following statements are equivalent:

- 1. f is a maximum flow.*
- 2. There exists an $s - t$ cut (A, B) with $|f| = C(A, B)$.*
- 3. There doesn't exist an augmenting path in G_f .*

Proof.

- ▶ (2 \Rightarrow 1): Increasing flow, increases $f(A, B)$ which violates capacity constraints
- ▶ (1 \Rightarrow 3): If there's an augmenting path p then flow can be increased by $b(p)$ units

Max-Flow Min-Cut

Theorem (Max-Flow Min-Cut)

For any flow network and flow f , the following statements are equivalent:

1. f is a maximum flow.
2. There exists an $s - t$ cut (A, B) with $|f| = C(A, B)$.
3. There doesn't exist an augmenting path in G_f .

Proof.

- ▶ (2 \Rightarrow 1): Increasing flow, increases $f(A, B)$ which violates capacity constraints
- ▶ (1 \Rightarrow 3): If there's an augmenting path p then flow can be increased by $b(p)$ units
- ▶ (3 \Rightarrow 2): Suppose G_f has no augmenting path. Define cut

$$A = \{v : v \text{ is reachable from } s \text{ in } G_f\} \text{ and } B = V - A$$

$$\forall u \in A, v \in B, f(u, v) = C(u, v). \text{ Hence } C(A, B) = f(A, B) = |f|$$



Ford-Fulkerson Algorithm

Algorithm

1. *flow* $f = 0$
2. *while there exists an augmenting path* p *for* f
 - 2.1 *find augmenting path* p
 - 2.2 *augment* f *by* $b(p)$ *units along* p
3. *return* f

Ford-Fulkerson Algorithm

Algorithm

1. *flow* $f = 0$
2. *while* there exists an augmenting path p for f
 - 2.1 *find* augmenting path p
 - 2.2 *augment* f by $b(p)$ units along p
3. *return* f

Theorem

The algorithm finds a maximum flow in time $O(|E||f^|)$ if capacities are integral where $|f^*|$ is the size of the maximum flow.*

Ford-Fulkerson Algorithm

Algorithm

1. flow $f = 0$
2. while there exists an augmenting path p for f
 - 2.1 find augmenting path p
 - 2.2 augment f by $b(p)$ units along p
3. return f

Theorem

The algorithm finds a maximum flow in time $O(|E||f^*|)$ if capacities are integral where $|f^*|$ is the size of the maximum flow.

Proof.

$O(|E|)$ time to find each augmenting path via BFS and $|f^*|$ iterations because each augmenting path increases flow by at least 1. □

Ford-Fulkerson Algorithm with Edmonds-Karp Heuristic

Algorithm

1. *flow* $f = 0$
2. *while there exists an augmenting path* p *for* f
 - 2.1 *find shortest (unweighted) augmenting path* p
 - 2.2 *augment* f *by* $b(p)$ *units along* p
3. *return* f

Ford-Fulkerson Algorithm with Edmonds-Karp Heuristic

Algorithm

1. *flow* $f = 0$
2. *while there exists an augmenting path* p *for* f
 - 2.1 *find shortest (unweighted) augmenting path* p
 - 2.2 *augment* f *by* $b(p)$ *units along* p
3. *return* f

Theorem

The algorithm finds a maximum flow in time $O(|E|^2|V|)$

Proof of Running Time (1/3)

Definition

Let $\delta_f(s, u)$ be length of shortest unweighted path from s to u in the G_f .

Proof of Running Time (1/3)

Definition

Let $\delta_f(s, u)$ be length of shortest unweighted path from s to u in the G_f .

Definition

(u, v) is **critical** if it's on augmenting path p for f and $C_f(u, v) = b(p)$.

Proof of Running Time (1/3)

Definition

Let $\delta_f(s, u)$ be length of shortest unweighted path from s to u in the G_f .

Definition

(u, v) is **critical** if it's on augmenting path p for f and $C_f(u, v) = b(p)$.

Lemma

$\delta_f(s, v)$ is non-decreasing as f changes.

Proof of Running Time (1/3)

Definition

Let $\delta_f(s, u)$ be length of shortest unweighted path from s to u in the G_f .

Definition

(u, v) is **critical** if it's on augmenting path p for f and $C_f(u, v) = b(p)$.

Lemma

$\delta_f(s, v)$ is non-decreasing as f changes.

Lemma

Between occasions when (u, v) is critical, $\delta_f(s, u)$ increases by 2.

Proof of Running Time (1/3)

Definition

Let $\delta_f(s, u)$ be length of shortest unweighted path from s to u in the G_f .

Definition

(u, v) is **critical** if it's on augmenting path p for f and $C_f(u, v) = b(p)$.

Lemma

$\delta_f(s, v)$ is non-decreasing as f changes.

Lemma

Between occasions when (u, v) is critical, $\delta_f(s, u)$ increases by 2.

Proof of Running Time.

- ▶ Max distance in G_f is $|V|$ so any edge is critical at most $|V|/2$ times

Proof of Running Time (1/3)

Definition

Let $\delta_f(s, u)$ be length of shortest unweighted path from s to u in the G_f .

Definition

(u, v) is **critical** if it's on augmenting path p for f and $C_f(u, v) = b(p)$.

Lemma

$\delta_f(s, v)$ is non-decreasing as f changes.

Lemma

Between occasions when (u, v) is critical, $\delta_f(s, u)$ increases by 2.

Proof of Running Time.

- ▶ Max distance in G_f is $|V|$ so any edge is critical at most $|V|/2$ times
- ▶ At most $2|E|$ edges in residual network

Proof of Running Time (1/3)

Definition

Let $\delta_f(s, u)$ be length of shortest unweighted path from s to u in the G_f .

Definition

(u, v) is **critical** if it's on augmenting path p for f and $C_f(u, v) = b(p)$.

Lemma

$\delta_f(s, v)$ is non-decreasing as f changes.

Lemma

Between occasions when (u, v) is critical, $\delta_f(s, u)$ increases by 2.

Proof of Running Time.

- ▶ Max distance in G_f is $|V|$ so any edge is critical at most $|V|/2$ times
- ▶ At most $2|E|$ edges in residual network
- ▶ There's a critical edge in each iteration so $O(|E||V|)$ iterations

Proof of Running Time (1/3)

Definition

Let $\delta_f(s, u)$ be length of shortest unweighted path from s to u in the G_f .

Definition

(u, v) is **critical** if it's on augmenting path p for f and $C_f(u, v) = b(p)$.

Lemma

$\delta_f(s, v)$ is non-decreasing as f changes.

Lemma

Between occasions when (u, v) is critical, $\delta_f(s, u)$ increases by 2.

Proof of Running Time.

- ▶ Max distance in G_f is $|V|$ so any edge is critical at most $|V|/2$ times
- ▶ At most $2|E|$ edges in residual network
- ▶ There's a critical edge in each iteration so $O(|E||V|)$ iterations
- ▶ Each iteration takes $O(|E|)$ to find shortest path



Proof of Running Time (2/3)

Lemma

$\delta_f(s, v)$ is non-decreasing as f changes.

Proof of Running Time (2/3)

Lemma

$\delta_f(s, v)$ is non-decreasing as f changes.

Proof.

- ▶ Consider augmenting f to f'

Proof of Running Time (2/3)

Lemma

$\delta_f(s, v)$ is non-decreasing as f changes.

Proof.

- ▶ Consider augmenting f to f'
- ▶ For contradiction, pick v that minimizes $\delta_{f'}(s, v)$ subject to:

$$\delta_{f'}(s, v) < \delta_f(s, v)$$

and let u be vertex before v on shortest path in $G_{f'}$ from s to v

Proof of Running Time (2/3)

Lemma

$\delta_f(s, v)$ is non-decreasing as f changes.

Proof.

- ▶ Consider augmenting f to f'
- ▶ For contradiction, pick v that minimizes $\delta_{f'}(s, v)$ subject to:

$$\delta_{f'}(s, v) < \delta_f(s, v)$$

and let u be vertex before v on shortest path in $G_{f'}$ from s to v

- ▶ **Claim** $(u, v) \notin E_f$

Proof of Running Time (2/3)

Lemma

$\delta_f(s, v)$ is non-decreasing as f changes.

Proof.

- ▶ Consider augmenting f to f'
- ▶ For contradiction, pick v that minimizes $\delta_{f'}(s, v)$ subject to:

$$\delta_{f'}(s, v) < \delta_f(s, v)$$

and let u be vertex before v on shortest path in $G_{f'}$ from s to v

- ▶ **Claim** $(u, v) \notin E_f$
 - ▶ Otherwise $\delta_f(s, v) \leq \delta_f(s, u) + 1$

Proof of Running Time (2/3)

Lemma

$\delta_f(s, v)$ is non-decreasing as f changes.

Proof.

- ▶ Consider augmenting f to f'
- ▶ For contradiction, pick v that minimizes $\delta_{f'}(s, v)$ subject to:

$$\delta_{f'}(s, v) < \delta_f(s, v)$$

and let u be vertex before v on shortest path in $G_{f'}$ from s to v

- ▶ **Claim** $(u, v) \notin E_f$
 - ▶ Otherwise $\delta_f(s, v) \leq \delta_f(s, u) + 1$
 - ▶ But $\delta_f(s, u) \leq \delta_{f'}(s, u)$ and so $\delta_f(s, v) \leq \delta_{f'}(s, u) + 1 = \delta_{f'}(s, v)$

Proof of Running Time (2/3)

Lemma

$\delta_f(s, v)$ is non-decreasing as f changes.

Proof.

- ▶ Consider augmenting f to f'
- ▶ For contradiction, pick v that minimizes $\delta_{f'}(s, v)$ subject to:

$$\delta_{f'}(s, v) < \delta_f(s, v)$$

and let u be vertex before v on shortest path in $G_{f'}$ from s to v

- ▶ **Claim** $(u, v) \notin E_f$
 - ▶ Otherwise $\delta_f(s, v) \leq \delta_f(s, u) + 1$
 - ▶ But $\delta_f(s, u) \leq \delta_{f'}(s, u)$ and so $\delta_f(s, v) \leq \delta_{f'}(s, u) + 1 = \delta_{f'}(s, v)$
- ▶ $(u, v) \notin E_f$ and $(u, v) \in E_{f'}$ implies augmentation contains (v, u)

Proof of Running Time (2/3)

Lemma

$\delta_f(s, v)$ is non-decreasing as f changes.

Proof.

- ▶ Consider augmenting f to f'
- ▶ For contradiction, pick v that minimizes $\delta_{f'}(s, v)$ subject to:

$$\delta_{f'}(s, v) < \delta_f(s, v)$$

and let u be vertex before v on shortest path in $G_{f'}$ from s to v

- ▶ **Claim** $(u, v) \notin E_f$
 - ▶ Otherwise $\delta_f(s, v) \leq \delta_f(s, u) + 1$
 - ▶ But $\delta_f(s, u) \leq \delta_{f'}(s, u)$ and so $\delta_f(s, v) \leq \delta_{f'}(s, u) + 1 = \delta_{f'}(s, v)$
- ▶ $(u, v) \notin E_f$ and $(u, v) \in E_{f'}$ implies augmentation contains (v, u)
- ▶ Since augmentation was shortest path:

$$\delta_f(s, v) = \delta_f(s, u) - 1 \leq \delta_{f'}(s, u) - 1 = \delta_{f'}(s, v) - 2$$



Proof of Running Time (3/3)

Lemma

Between occasions when (u, v) is critical, $\delta_f(s, u)$ increases by 2.

Proof of Running Time (3/3)

Lemma

Between occasions when (u, v) is critical, $\delta_f(s, u)$ increases by 2.

Proof.

- ▶ Let (u, v) be critical in the augmentation of f

Proof of Running Time (3/3)

Lemma

Between occasions when (u, v) is critical, $\delta_f(s, u)$ increases by 2.

Proof.

- ▶ Let (u, v) be critical in the augmentation of f
- ▶ Since (u, v) on shortest path: $\delta_f(s, u) = \delta_f(s, v) - 1$

Proof of Running Time (3/3)

Lemma

Between occasions when (u, v) is critical, $\delta_f(s, u)$ increases by 2.

Proof.

- ▶ Let (u, v) be critical in the augmentation of f
- ▶ Since (u, v) on shortest path: $\delta_f(s, u) = \delta_f(s, v) - 1$
- ▶ After augmentation (u, v) disappears from residual network!

Proof of Running Time (3/3)

Lemma

Between occasions when (u, v) is critical, $\delta_f(s, u)$ increases by 2.

Proof.

- ▶ Let (u, v) be critical in the augmentation of f
- ▶ Since (u, v) on shortest path: $\delta_f(s, u) = \delta_f(s, v) - 1$
- ▶ After augmentation (u, v) disappears from residual network!
- ▶ Let f'' be the next flow where $(u, v) \in G_{f''}$ and let f' be the flow right before f''

Proof of Running Time (3/3)

Lemma

Between occasions when (u, v) is critical, $\delta_f(s, u)$ increases by 2.

Proof.

- ▶ Let (u, v) be critical in the augmentation of f
- ▶ Since (u, v) on shortest path: $\delta_f(s, u) = \delta_f(s, v) - 1$
- ▶ After augmentation (u, v) disappears from residual network!
- ▶ Let f'' be the next flow where $(u, v) \in G_{f''}$ and let f' be the flow right before f''
- ▶ $(u, v) \notin G_{f'}$ but $(u, v) \in G_{f''}$ implies (v, u) used to augment f'

Proof of Running Time (3/3)

Lemma

Between occasions when (u, v) is critical, $\delta_f(s, u)$ increases by 2.

Proof.

- ▶ Let (u, v) be critical in the augmentation of f
- ▶ Since (u, v) on shortest path: $\delta_f(s, u) = \delta_f(s, v) - 1$
- ▶ After augmentation (u, v) disappears from residual network!
- ▶ Let f'' be the next flow where $(u, v) \in G_{f''}$ and let f' be the flow right before f''
- ▶ $(u, v) \notin G_{f'}$ but $(u, v) \in G_{f''}$ implies (v, u) used to augment f'
- ▶ Therefore $\delta_{f'}(s, v) = \delta_{f'}(s, u) - 1$ and so

$$\delta_f(s, u) = \delta_f(s, v) - 1 \leq \delta_{f'}(s, v) - 1 = \delta_{f'}(s, u) - 2$$



Application to Bipartite Matching

Consider bipartite graph $G = (U, V, E)$ and define \hat{G} by directing each edge from U to V , adding nodes s, t , and edges $s \times U$ and $V \times t$. Let capacity of each edge in \hat{G} be 1.

Application to Bipartite Matching

Consider bipartite graph $G = (U, V, E)$ and define \hat{G} by directing each edge from U to V , adding nodes s, t , and edges $s \times U$ and $V \times t$. Let capacity of each edge in \hat{G} be 1.

Lemma

G has a matching of size k iff \hat{G} has a $s - t$ flow of size $|k|$.

Application to Bipartite Matching

Consider bipartite graph $G = (U, V, E)$ and define \hat{G} by directing each edge from U to V , adding nodes s, t , and edges $s \times U$ and $V \times t$. Let capacity of each edge in \hat{G} be 1.

Lemma

G has a matching of size k iff \hat{G} has a $s - t$ flow of size $|k|$.

Proof.

1. Assume M matching of size k : For each $(u, v) \in M$ deliver one unit of flow along $(s, u), (u, v)$, and (v, t) . Gives flow of size k
2. Assume there is a flow of size k :
 - 2.1 May assume the flow is integral.

Application to Bipartite Matching

Consider bipartite graph $G = (U, V, E)$ and define \hat{G} by directing each edge from U to V , adding nodes s, t , and edges $s \times U$ and $V \times t$. Let capacity of each edge in \hat{G} be 1.

Lemma

G has a matching of size k iff \hat{G} has a $s - t$ flow of size $|k|$.

Proof.

1. Assume M matching of size k : For each $(u, v) \in M$ deliver one unit of flow along (s, u) , (u, v) , and (v, t) . Gives flow of size k
2. Assume there is a flow of size k :
 - 2.1 May assume the flow is integral.
 - 2.2 At most 1 unit of flow into each $u \in U$: at most 1 unit of flow out

Application to Bipartite Matching

Consider bipartite graph $G = (U, V, E)$ and define \hat{G} by directing each edge from U to V , adding nodes s, t , and edges $s \times U$ and $V \times t$. Let capacity of each edge in \hat{G} be 1.

Lemma

G has a matching of size k iff \hat{G} has a $s - t$ flow of size $|k|$.

Proof.

1. Assume M matching of size k : For each $(u, v) \in M$ deliver one unit of flow along $(s, u), (u, v)$, and (v, t) . Gives flow of size k
2. Assume there is a flow of size k :
 - 2.1 May assume the flow is integral.
 - 2.2 At most 1 unit of flow into each $u \in U$: at most 1 unit of flow out
 - 2.3 At most 1 unit of flow out of each $v \in V$: at most 1 unit of flow in

Application to Bipartite Matching

Consider bipartite graph $G = (U, V, E)$ and define \hat{G} by directing each edge from U to V , adding nodes s, t , and edges $s \times U$ and $V \times t$. Let capacity of each edge in \hat{G} be 1.

Lemma

G has a matching of size k iff \hat{G} has a $s - t$ flow of size $|k|$.

Proof.

1. Assume M matching of size k : For each $(u, v) \in M$ deliver one unit of flow along $(s, u), (u, v)$, and (v, t) . Gives flow of size k
2. Assume there is a flow of size k :
 - 2.1 May assume the flow is integral.
 - 2.2 At most 1 unit of flow into each $u \in U$: at most 1 unit of flow out
 - 2.3 At most 1 unit of flow out of each $v \in V$: at most 1 unit of flow in
 - 2.4 Set of edges in $U \times V$ carrying flow is a matching

Application to Bipartite Matching

Consider bipartite graph $G = (U, V, E)$ and define \hat{G} by directing each edge from U to V , adding nodes s, t , and edges $s \times U$ and $V \times t$. Let capacity of each edge in \hat{G} be 1.

Lemma

G has a matching of size k iff \hat{G} has a $s - t$ flow of size $|k|$.

Proof.

1. Assume M matching of size k : For each $(u, v) \in M$ deliver one unit of flow along $(s, u), (u, v)$, and (v, t) . Gives flow of size k
2. Assume there is a flow of size k :
 - 2.1 May assume the flow is integral.
 - 2.2 At most 1 unit of flow into each $u \in U$: at most 1 unit of flow out
 - 2.3 At most 1 unit of flow out of each $v \in V$: at most 1 unit of flow in
 - 2.4 Set of edges in $U \times V$ carrying flow is a matching
 - 2.5 $(s + U, t + V)$ is an $s - t$ cut and so flow across the cut must be k



For Next Time...

- ▶ Finish reading first five chapters
- ▶ Be prompt for midterm on Thursday (22nd October)