CMPSCI 611: Advanced Algorithms

Lecture 12: Network Flow Part II

Andrew McGregor

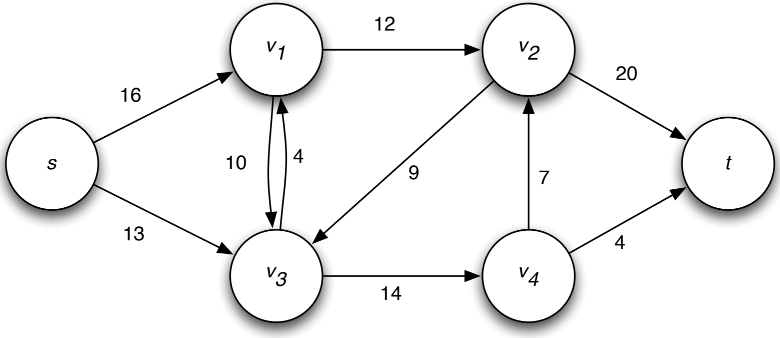
Last Compiled: April 4, 2024

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

Capacity



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

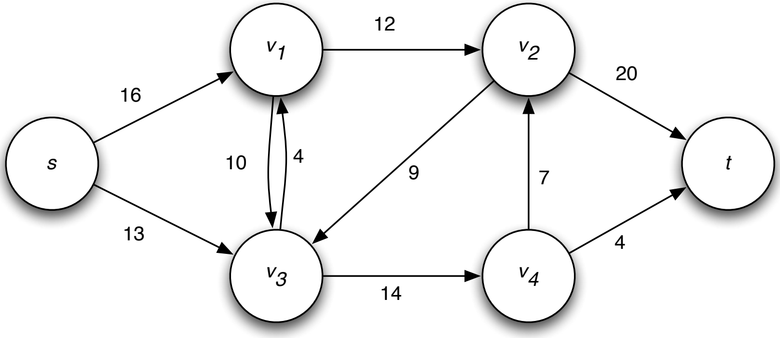
Output: A flow f from s to t where $f : V \times V \to \mathbb{R}$ satisfies

- 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: $\forall u, v \in V,\ f(u, v) \leq C(u, v)$

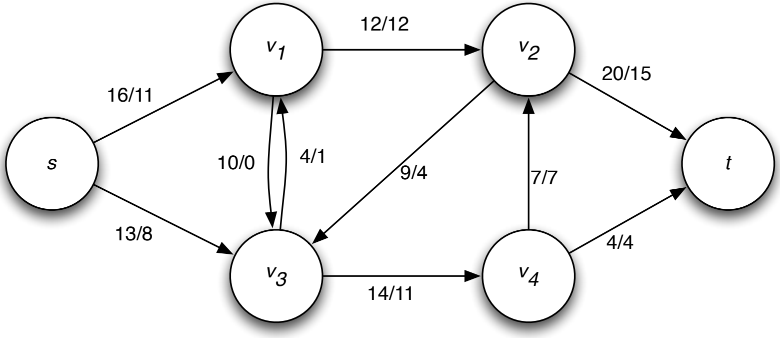
Goal: Maximize *"size of the flow"*, i.e., the total flow coming leaving s:

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

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 $t \in B$.

Definition

The capacity of a cut (A, B) is

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

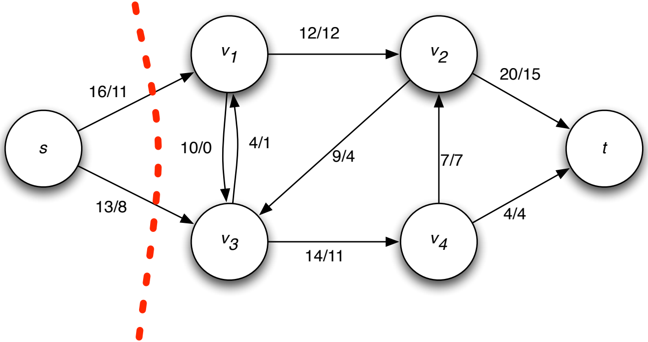
Definition

The flow across a cut (A, B) is

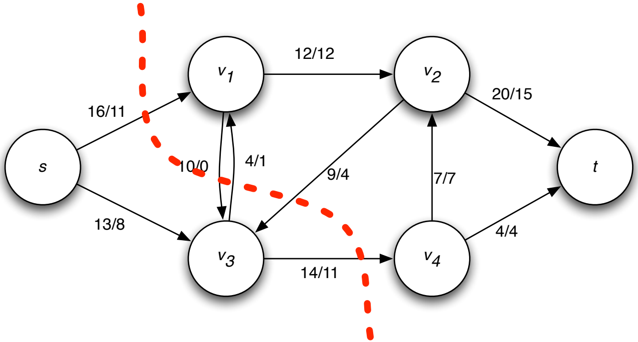
$$f(A, B) = \sum_{u \in A, v \in B} f(u, v)$$

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

First Cut



Second Cut



All cuts have same flow

Lemma

For any flow f: for all s-t cuts (A, B), $f(A, B)$ equals $|f|$.

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 $u \in A' - s$:

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

- 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.

□

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.

Definition

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

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

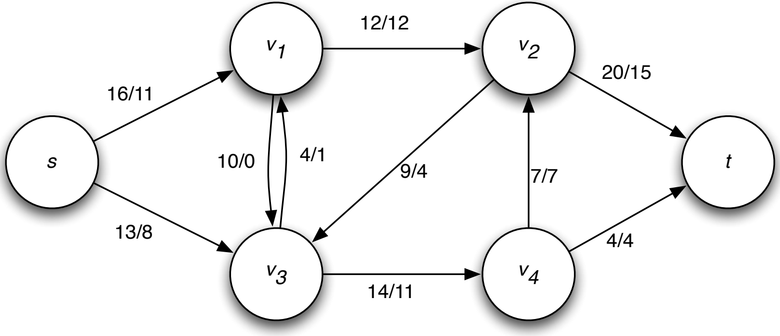
$$C_f(u, v) = C(u, v) - f(u, v)$$

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

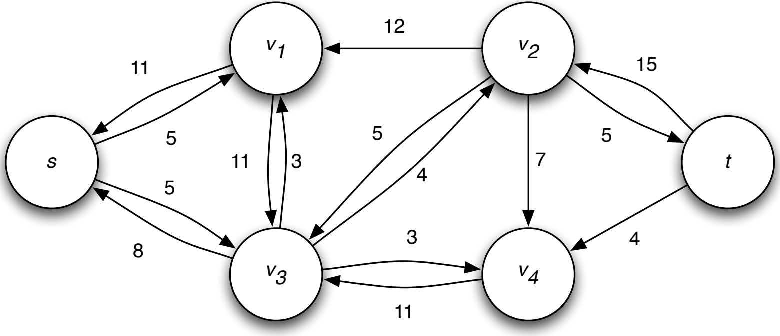
Definition

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.

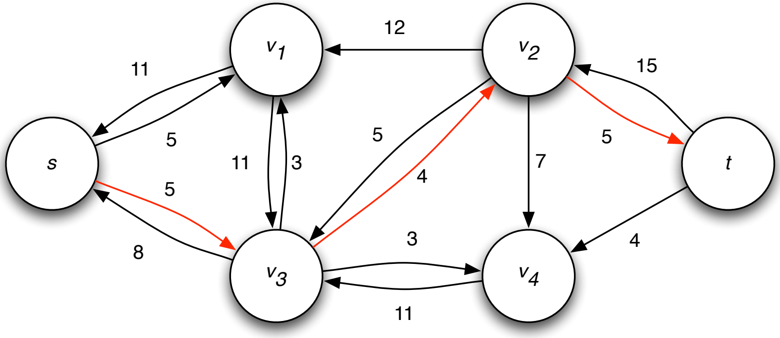
Capacity/Flow



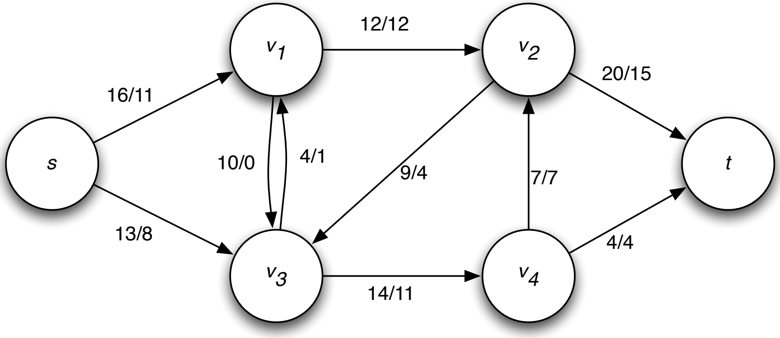
Residual



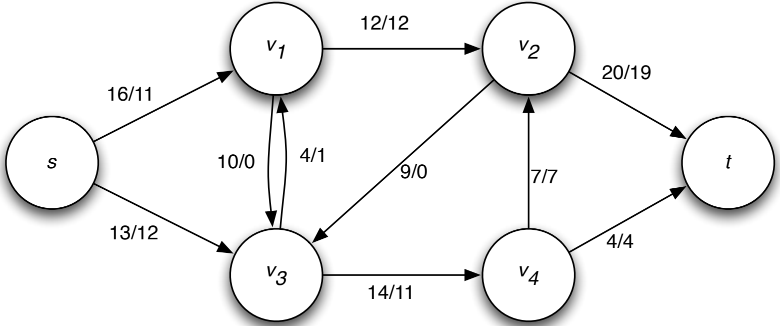
Augmenting Path



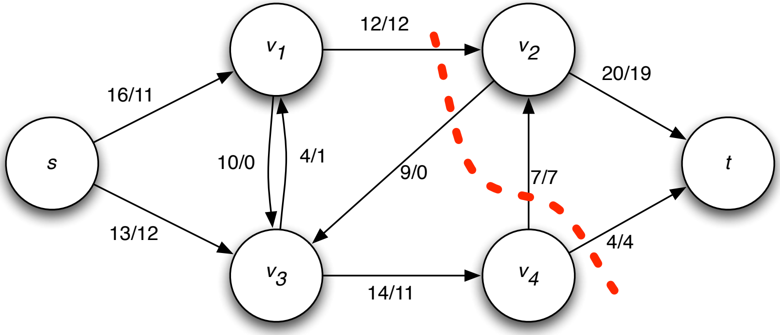
Old Flow



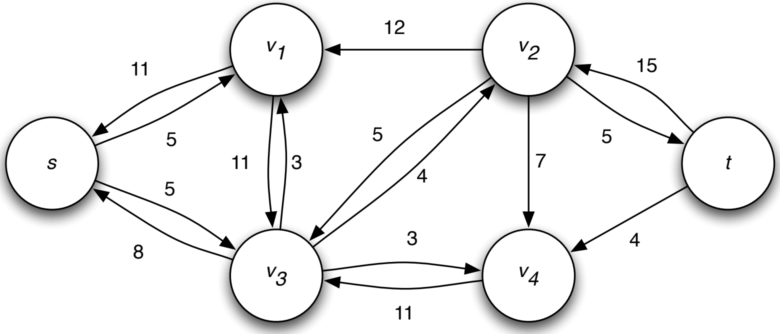
New Flow



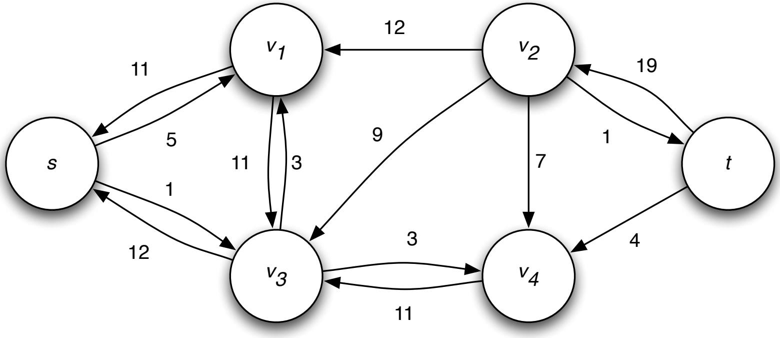
Min Capacity Cut Proves this is Optimal



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 $|f| = f(A, B) = C(A, B)$.*
3. *There doesn't exist an augmenting path in G_f.*

Proof.

- $(2 \Rightarrow 1)$: Increasing flow, increases $f(A, B)$ which violates capacity
- $(1 \Rightarrow 3)$: If p is an augmenting path, can increase flow by $b(p)$
- $(3 \Rightarrow 2)$: 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

Theorem

The algorithms 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

Theorem

The algorithms 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 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 at least 2.

Proof of Running Time.

- Max distance in G_f is $|V|$ so any edge is critical at most $1 + |V|/2$ times
- At most $2|E|$ edges in residual network
- There's a critical edge in each iteration so $r = O(|E||V|)$ iterations.
- Each iteration takes $O(|E|)$ to find shortest path

□