f-augmenting paths

Definition

If f is a feasible flow, an **f-augmenting path** is a path P from s to t such that, for each $e \in E(P)$,

- if P follows e in the forward direction, then $f(e) < c(e)$,

f-augmenting paths

Definition

If f is a feasible flow, an ***f*-augmenting path** is a path P from s to t such that, for each $e \in E(P)$,

- if P follows e in the forward direction, then $f(e) < c(e)$,
- if P follows e in the backward direction, then $f(e) > 0$.

f-augmenting paths

Definition

If f is a feasible flow, an **f-augmenting path** is a path P from s to t such that, for each $e \in E(P)$,

- if P follows e in the forward direction, then $f(e) < c(e)$, (let $\epsilon(e) = c(e) - f(e)$)
- if P follows e in the backward direction, then $f(e) > 0$. (let $\epsilon(e) = f(e)$)

The **tolerance** of P is

$$\min_{e \in E(P)} \epsilon(e).$$

f-augmenting paths

Definition

If f is a feasible flow, an **f-augmenting path** is a path P from s to t such that, for each $e \in E(P)$,

- if P follows e in the forward direction, then $f(e) < c(e)$, (let $\epsilon(e) = c(e) - f(e)$)
- if P follows e in the backward direction, then $f(e) > 0$. (let $\epsilon(e) = f(e)$)

The **tolerance** of P is

$$\min_{e \in E(P)} \epsilon(e).$$

An f-augmenting path leads to a feasible flow with larger value, by changing f on the edges of P.

f-augmenting paths

Definition

If f is a feasible flow, an **f-augmenting path** is a path P from s to t such that, for each $e \in E(P)$,

- if P follows e in the forward direction, then $f(e) < c(e)$, (let $\epsilon(e) = c(e) - f(e)$)
- if P follows e in the backward direction, then $f(e) > 0$. (let $\epsilon(e) = f(e)$)

The **tolerance** of P is

$$\min_{e \in E(P)} \epsilon(e).$$

An f-augmenting path leads to a feasible flow with larger value, by changing f on the edges of P.

How can we know when our flow is maximum?

Source/sink cuts

Definition

A **source/sink cut** $[S, T]$ consists of the edges from S to T (in this direction), where S and T partition the set of nodes (i.e., $T = \overline{S}$), with $s \in S$ and $t \in T$.

Source/sink cuts

Definition

A **source/sink cut** $[S, T]$ consists of the edges from S to T (in this direction), where S and T partition the set of nodes (i.e., $T = \overline{S}$), with $s \in S$ and $t \in T$.
The **capacity** of $[S, T]$, denoted by $\text{cap}(S, T)$, is the sum of the capacities of the edges in $[S, T]$.

Source/sink cuts

Definition

A **source/sink cut** $[S, T]$ consists of the edges from S to T (in this direction), where S and T partition the set of nodes (i.e., $T = \overline{S}$), with $s \in S$ and $t \in T$.
The **capacity** of $[S, T]$, denoted by $\text{cap}(S, T)$, is the sum of the capacities of the edges in $[S, T]$.
A **minimum cut** is a source/sink cut with minimum capacity.

Source/sink cuts

Definition

A **source/sink cut** $[S, T]$ consists of the edges from S to T (in this direction), where S and T partition the set of nodes (i.e., $T = \overline{S}$), with $s \in S$ and $t \in T$.
The **capacity** of $[S, T]$, denoted by $\text{cap}(S, T)$, is the sum of the capacities of the edges in $[S, T]$.
A **minimum cut** is a source/sink cut with minimum capacity.

Proposition

If f is a feasible flow and $[S, T]$ is a source/sink cut, then

$$\text{val}(f) \leq \text{cap}(S, T).$$

Source/sink cuts

Definition

A **source/sink cut** $[S, T]$ consists of the edges from S to T (in this direction), where S and T partition the set of nodes (i.e., $T = \overline{S}$), with $s \in S$ and $t \in T$.
The **capacity** of $[S, T]$, denoted by $\text{cap}(S, T)$, is the sum of the capacities of the edges in $[S, T]$.
A **minimum cut** is a source/sink cut with minimum capacity.

Proposition

If f is a feasible flow and $[S, T]$ is a source/sink cut, then

$$\text{val}(f) \leq \text{cap}(S, T).$$

Proof idea: denoting by $f^+(S)$ and $f^-(S)$ the total flow on edges leaving S and entering S, respectively, we have

$$f^+(S) - f^-(S) = \sum_{v \in S} (f^+(v) - f^-(v)) = f^+(s) - f^-(s) = \text{val}(f).$$

Max-flow Min-cut Theorem

As a consequence of the previous proposition:

value of a maximum flow $\leq$ capacity of a minimum cut.

Max-flow Min-cut Theorem

As a consequence of the previous proposition:

value of a maximum flow $\leq$ capacity of a minimum cut.

Theorem (Max-flow Min-cut Theorem)

In every network, the value of a maximum flow equals the capacity of a minimum cut.

Max-flow Min-cut Theorem

As a consequence of the previous proposition:

value of a maximum flow $\leq$ capacity of a minimum cut.

Theorem (Max-flow Min-cut Theorem)

In every network, the value of a maximum flow equals the capacity of a minimum cut.

To prove this, we will give an algorithm that, for any given network, finds a feasible flow and a source/sink cut with the property that the value of the flow equals the capacity of the cut.

Ford–Fulkerson Algorithm

Input: A feasible flow f.

Output: An f-augmenting path, or a source/sink cut with capacity equal to $\mathrm{val}(f)$.

Ford–Fulkerson Algorithm

Input: A feasible flow f.

Output: An f-augmenting path, or a source/sink cut with capacity equal to $\text{val}(f)$.

The idea is to find nodes reachable from s by paths with positive tolerance, and see if we can get to t.

Ford–Fulkerson Algorithm

Input: A feasible flow f.

Output: An f-augmenting path, or a source/sink cut with capacity equal to $\text{val}(f)$.

The idea is to find nodes reachable from s by paths with positive tolerance, and see if we can get to t.

Initialization: $R = \{s\}$ (reached nodes), $\quad S = \emptyset$ (searched notes).

Ford–Fulkerson Algorithm

Input: A feasible flow f.

Output: An f-augmenting path, or a source/sink cut with capacity equal to val(f).

The idea is to find nodes reachable from s by paths with positive tolerance, and see if we can get to t.

Initialization: $R = \{s\}$ (reached nodes), $S = \emptyset$ (searched notes).

Iteration: Choose $v \in R \setminus S$.

- For each exiting edge vw with $f(vw) < c(vw)$ and $w \notin R$, add w to R.
- For each entering edge uv with $f(uv) > 0$ and $u \notin R$, add u to R.

Add v to S.

Ford–Fulkerson Algorithm

Input: A feasible flow f.

Output: An f-augmenting path, or a source/sink cut with capacity equal to $\text{val}(f)$.

The idea is to find nodes reachable from s by paths with positive tolerance, and see if we can get to t.

Initialization: $R = \{s\}$ (reached nodes), $S = \emptyset$ (searched notes).

Iteration: Choose $v \in R \setminus S$.

- For each exiting edge vw with $f(vw) < c(vw)$ and $w \notin R$, add w to R.
- For each entering edge uv with $f(uv) > 0$ and $u \notin R$, add u to R.

Add v to S. Consider three cases:

- If $t \in R$, then the path reaching t is an f-augmenting path.

Ford–Fulkerson Algorithm

Input: A feasible flow f.

Output: An f-augmenting path, or a source/sink cut with capacity equal to $\mathrm{val}(f)$.

The idea is to find nodes reachable from s by paths with positive tolerance, and see if we can get to t.

Initialization: $R = \{s\}$ (reached nodes), $S = \emptyset$ (searched notes).

Iteration: Choose $v \in R \setminus S$.

- For each exiting edge vw with $f(vw) < c(vw)$ and $w \notin R$, add w to R.
- For each entering edge uv with $f(uv) > 0$ and $u \notin R$, add u to R.

Add v to S. Consider three cases:

- If $t \in R$, then the path reaching t is an f-augmenting path.
- If $R = S$, then $[S, \overline{S}]$ is a source/sink cut with capacity $\mathrm{val}(f)$.

Ford–Fulkerson Algorithm

Input: A feasible flow f.

Output: An f-augmenting path, or a source/sink cut with capacity equal to $\text{val}(f)$.

The idea is to find nodes reachable from s by paths with positive tolerance, and see if we can get to t.

Initialization: $R = \{s\}$ (reached nodes), $S = \emptyset$ (searched notes).

Iteration: Choose $v \in R \setminus S$.

- For each exiting edge vw with $f(vw) < c(vw)$ and $w \notin R$, add w to R.
- For each entering edge uv with $f(uv) > 0$ and $u \notin R$, add u to R.

Add v to S. Consider three cases:

- If $t \in R$, then the path reaching t is an f-augmenting path.
- If $R = S$, then $[S, \overline{S}]$ is a source/sink cut with capacity $\text{val}(f)$.
- Otherwise, iterate.

Proof of Max-flow Min-cut Theorem

Theorem (Max-flow Min-cut Theorem)

In every network, the value of a maximum flow equals the capacity of a minimum cut.

Proof of Max-flow Min-cut Theorem

Theorem (Max-flow Min-cut Theorem)

In every network, the value of a maximum flow equals the capacity of a minimum cut.

Proof: Apply the Ford–Fulkerson algorithm repeatedly, starting from the zero flow ($f(e) = 0$ for every e).

Proof of Max-flow Min-cut Theorem

Theorem (Max-flow Min-cut Theorem)

In every network, the value of a maximum flow equals the capacity of a minimum cut.

Proof: Apply the Ford–Fulkerson algorithm repeatedly, starting from the zero flow ($f(e) = 0$ for every e).

As long as the algorithm returns an f-augmenting path, use it to increase the value of the flow, and apply the algorithm again.

Proof of Max-flow Min-cut Theorem

Theorem (Max-flow Min-cut Theorem)

In every network, the value of a maximum flow equals the capacity of a minimum cut.

Proof: Apply the Ford–Fulkerson algorithm repeatedly, starting from the zero flow ($f(e) = 0$ for every e).

As long as the algorithm returns an f-augmenting path, use it to increase the value of the flow, and apply the algorithm again.

Eventually, the algorithm returns a source/sink cut $[S, T]$, which satisfies

$$\text{val}(f) = f^{+}(S) - f^{-}(S) = \text{cap}(S, T),$$

so we have found a maximum flow and a minimum cut.

Proof of Max-flow Min-cut Theorem

Theorem (Max-flow Min-cut Theorem)

In every network, the value of a maximum flow equals the capacity of a minimum cut.

Proof: Apply the Ford–Fulkerson algorithm repeatedly, starting from the zero flow ($f(e) = 0$ for every e).

As long as the algorithm returns an f-augmenting path, use it to increase the value of the flow, and apply the algorithm again.

Eventually, the algorithm returns a source/sink cut $[S, T]$, which satisfies

$$\text{val}(f) = f^+(S) - f^-(S) = \text{cap}(S, T),$$

so we have found a maximum flow and a minimum cut.

Caveat: If the capacities are irrational, we could get augmenting paths forever!

Proof of Max-flow Min-cut Theorem

Theorem (Max-flow Min-cut Theorem)

In every network, the value of a maximum flow equals the capacity of a minimum cut.

Proof: Apply the Ford–Fulkerson algorithm repeatedly, starting from the zero flow ($f(e) = 0$ for every e).

As long as the algorithm returns an f-augmenting path, use it to increase the value of the flow, and apply the algorithm again.

Eventually, the algorithm returns a source/sink cut $[S, T]$, which satisfies

$$\mathrm{val}(f) = f^+(S) - f^-(S) = \mathrm{cap}(S, T),$$

so we have found a maximum flow and a minimum cut.

Caveat: If the capacities are irrational, we could get augmenting paths forever!
There is a way to fix the algorithm so that this never happens.

Integrality Theorem

Corollary

If all capacities are integers, then there exists a maximum flow assigning integer values to all edges.