Max-Flow and Min-Cut

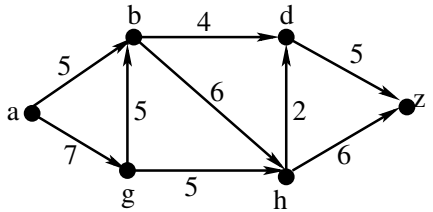
Arash Rafiey

1 December 2015

Max-Flow and Min-Cut

We say a directed loop-less graph D is a *network* or *transport network* if :

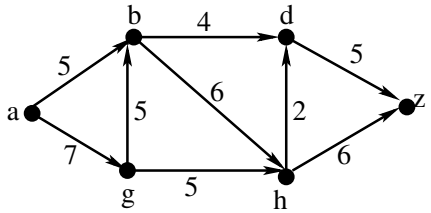
- D has a *source* vertex, a vertex without in-neighbor
- D has a *sink* vertex, a vertex without out-neighbor
- D is weighted (non-negative integer values), every arc e has a weight (capacity), $c(e)$.



Max-Flow and Min-Cut

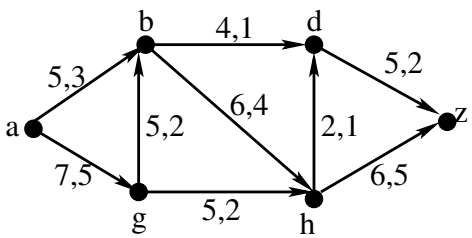
We say a directed loop-less graph D is a *network* or *transport network* if :

- D has a *source* vertex, a vertex without in-neighbor
- D has a *sink* vertex, a vertex without out-neighbor
- D is weighted (non-negative integer values), every arc e has a weight (capacity), $c(e)$.

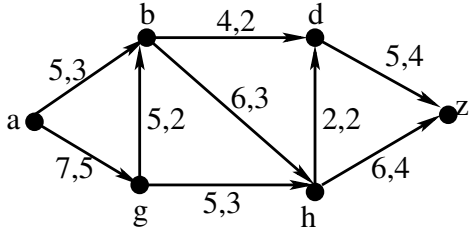


Function $F : E(D) \longrightarrow N^+$ is a *flow* if :

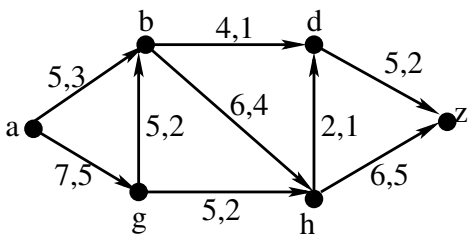
- $f(e) \leq c(e)$ for every $e \in E$.
- for each $v \in V$ other than source and sink, $\sum_{w \in V} f(wv) = \sum_{u \in V} f(vu)$. (if wv is not an arc $f(wv) = 0$)



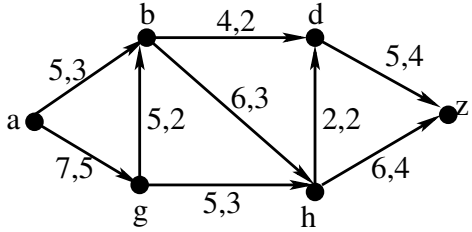
A



B

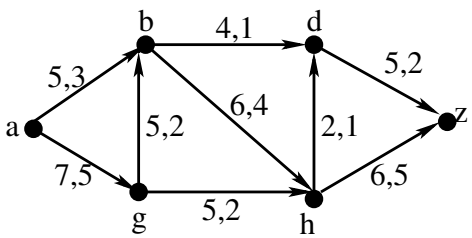


A

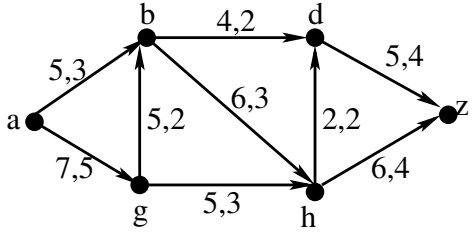


B

A is not a flow and B is a flow.



A



B

A is not a flow and B is a flow.

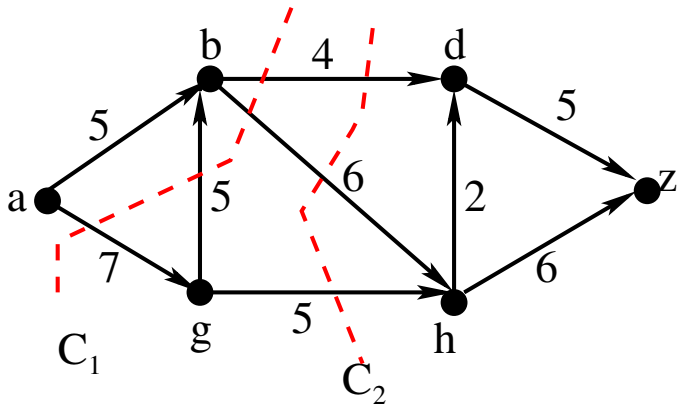
Definition

Let f be a flow for network $D = (V, E)$.

- An edge e is called *saturated* if $f(e) = c(e)$ and *unsaturated* otherwise.
- If a is a source of D then $val(f) = \sum_{v \in V} f(av)$ is called the *value* of f.

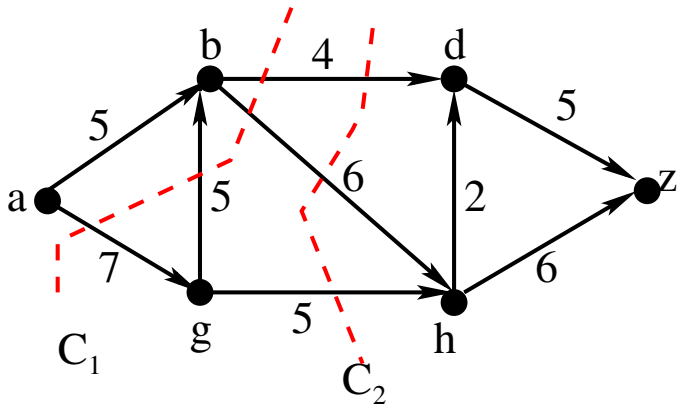
Definition

Let $D = (V, E)$ be a network. A set of arcs C is a **cut (cut-set)** if $E \setminus C$ separates source a from sink z ($a - z$-cut). (There is no directed path from a to z after removing the edges in C from E.



Definition

Let $D = (V, E)$ be a network. A set of arcs C is a **cut (cut-set)** if $E \setminus C$ separates source a from sink z ($a - z$-cut). (There is no directed path from a to z after removing the edges in C from E.



We usually denote a cut-set C by $(P, \overline{P})$. P is a set of vertices in V containing source a and $\overline{P} = V \setminus P$.
The cost of cut-set $C = (P, \overline{P})$ is denoted by

$$c(P, \overline{P}) = \sum_{v \in P, w \in \overline{P}} c(v, w).$$

Theorem

Let f be a flow in a network $D = (V, E)$. If $C = (P, \overline{P})$ is any cut in D, then $val(f) \leq c(P, \overline{P})$.

Proof.

Let a be the source. Since $id(a) = 0$, we can say for all $w \in V$, $f(wa) = 0$. Thus $val(f) = \sum_{v \in V} f(av) - \sum_{w \in V} f(wa)$. We also know for every $x \in P$, $x \neq a$, $\sum_{v \in V} f(xv) - \sum_{w \in V} f(wx) = 0$. Therefore :

$$val(f) = [\sum_{v \in v} f(av) - \sum_{w \in V} f(wa)] + \sum_{a \neq x \in P} [\sum_{v \in V} f(xv) - \sum_{w \in V} f(wx)]$$

$$= \sum_{x \in P, v \in V} f(xv) - \sum_{x \in P, w \in V} f(wx)$$

Proof.

$$\sum_{x\in P, v\in V} f(xv) - \sum_{x\in P, w\in V} f(wx)$$

$$= [\sum_{x,v\in P} f(xv) + \sum_{x\in P, v\in \overline{P}} f(xv)] - [\sum_{a,w\in P} f(wx) + \sum_{x\in P, w\in \overline{P}} f(wx)].$$

Since $\sum_{x,v\in P} f(xv)$, $\sum_{x,w\in P} f(wx)$ are summed over the same set of arcs, they are equal. Consequently,

$$val(f) = \sum_{x\in P, v\in \overline{P}} f(xv) - \sum_{x\in P, w\in \overline{P}} f(wx).$$

Therefore :
$val(f) \leq \sum_{x\in P, v\in \overline{P}} f(xv) \leq \sum_{x\in P, v\in \overline{P}} c(xv) = c(P, \overline{P})$ □

Corollary

If f is a flow in a network $D = (V, E)$, then value of the flow from source a is equal to the value of the flow in sink z.

Corollary

If f is a flow in a network $D = (V, E)$, then value of the flow from source a is equal to the value of the flow in sink z.

Proof.

This follows from the definition because the flow is not going to be lost. Whatever we send out from a will be received in z. □

Corollary

If f is a flow in a network $D = (V, E)$, then value of the flow from source a is equal to the value of the flow in sink z.

Proof.

This follows from the definition because the flow is not going to be lost. Whatever we send out from a will be received in z. □

Corollary

Let f be a flow in a network $D = (V, E)$ and let $C = (P, \overline{P})$ be a cut where $val(f) = c(P, \overline{P})$. Then f is a maximum flow for D and C is a minimum cut.

Corollary

If f is a flow in a network $D = (V, E)$, then value of the flow from source a is equal to the value of the flow in sink z.

Proof.

This follows from the definition because the flow is not going to be lost. Whatever we send out from a will be received in z. □

Corollary

Let f be a flow in a network $D = (V, E)$ and let $C = (P, \overline{P})$ be a cut where $val(f) = c(P, \overline{P})$. Then f is a maximum flow for D and C is a minimum cut.

Proof.

By previous theorem for any flow f_1 for D we have $val(f_1) \leq c(P, \overline{P}) = val(f)$. Therefore f is a maximum flow. Similarly for any cut $(P', \overline{P'})$ we have $c(P, \overline{P}) = val(f) \leq c(P', \overline{P'})$ and hence C is a minimum cut.

□

Corollary

Let f be a flow in a network $D = (V, E)$ and let $C = (P, \overline{P})$ be a cut. Then $val(f) = c(P, \overline{P})$ if and only if

- *(a) $f(e) = c(e)$ for every arc xy, where $x \in P$ and $y \in \overline{P}$.*
- *(b) $f(e) = 0$ for every arc vw, where $v \in \overline{P}$ and $w \in P$.*

Furthermore, f is a maximum flow and C is a minimum cut.

Definition

Let D be a network. We say a sequence $a = v_0, e_1, v_1, e_2, v_2, \ldots, e_n, v_n = z$ (a is source and z is a sink) is an oriented path, where each e_i is either in a forward direction or backward direction.

Definition

Let D be a network. We say a sequence $a = v_0, e_1, v_1, e_2, v_2, \ldots, e_n, v_n = z$ (a is source and z is a sink) is an oriented path, where each e_i is either in a forward direction or backward direction.

Definition

Let f be a flow in network $D = (V, E)$. An *f-augmenting path* p is an oriented path (from a to z) where for each edge e of p we have

$f(e) < c(e)$ for e forward edge

$f(e) > 0$ for e backward edge.

Definition

Let p be an f-augmenting path in network $D = (V, E)$. For each edge e on p

$$\Delta_e = \begin{cases} c(e) - f(e) & \text{for } e \text{ a forward edge} \\ f(e) & \text{for } e \text{ a backward edge} \end{cases}$$

Δ_e is called the tolerance of e.

Definition

Let p be an f-augmenting path in network $D = (V, E)$. For each edge e on p

$$\Delta_e = \begin{cases} c(e) - f(e) & \text{for } e \text{ a forward edge} \\ f(e) & \text{for } e \text{ a backward edge} \end{cases}$$

Δ_e is called the tolerance of e.

Theorem

Let f be a flow in network $D = (V, E)$ and let p be an f-augmenting path in D with $\Delta_p = \min_{e \in p} \Delta_e$.
Define $f_1 : E \to \mathcal{N}$ by

$$f_1(e) = \begin{cases} f(e) + \Delta_p & e \in p,\ e \text{ a forward edge} \\ f(e) - \Delta_p & e \in p,\ e \text{ a backward edge} \\ f(e) & e \in E,\ e \notin p \end{cases}$$

Then f_1 is a flow in D with $val(f_1) = val(f) + \Delta_p$.

Max-Flow Algorithm

We need to find an f-augmenting path first. We can do it using a modified version of BFS so called *BFS-f-augmenting* algorithm starting from source a.

At each step we create a new level by adding vertex w to a vertex v from the previous level if vw is a forward edge and $f(vw) < e(vw)$ or v, w is a backward edge (wv is an arc of the network) and $f(wv) > 0$. Once we reach z we stop.

Ford-Fulkerson Max-Flow Algorithm

Max-Flow(($D = (V, E)$)

1. Define flow f for every edge e by setting $f(e) = 0$.
2. Repeat :
3. Apply BFS-f-augmenting to find an f-augmenting path p
4. Let $\Delta_p = \min_{e \in p} \Delta_e$
5. **for** each edge $e \in p$
6. **if** e is a forward edge
7. $f(e) := f(e) + \Delta_p$.
8. **else** (e is a backward edge)
9. $f(e) := f(e) - \Delta_p$.
10. Until no f-augmenting path p can be found.
11. Return f.