

# Flow Network

A **flow network** is a directed graph  $G = (V, E)$ , and a capacity function

$$c : V \times V \rightarrow R$$

such that:

1. For each  $(u, v) \in E$  the capacity  $c(u, v) \geq 0$ .
2. If  $(u, v) \notin E$  then  $c(u, v) = 0$ .
3. There is a source  $s$  and a sink  $t$ .
4. Each vertex is on a directed path from  $s$  to  $t$ .

# Flow

A flow in a flow network  $G$  with capacity function  $c$  is a function  $f : V \times V \rightarrow R$  such that:

1. For all  $(u, v) \in E$  (capacity constraint)

$$f(u, v) \leq c(u, v)$$

2. For all  $u, v \in V$ , (symmetry)

$$f(u, v) = -f(v, u).$$

3. For all  $u \in V - \{s, t\}$ , (flow conservation)

$$\sum_{v \in V} f(u, v) = 0.$$

The **value** of the flow is

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

# Problems

Given a flow network  $G$  with source  $s$  and sink  $t$ , find the **maximum** flow.

Multi-source, multi-sink problem.

Minimum-cost flow: ship  $d$  units from  $s$  to  $t$  such that the total cost is minimized (each edge charges a certain amount for a unit of flow through it).

Multi-commodity flow: ship  $k$  commodities, each with its own sink, source, and flow value (demand).

# The Ford-Fulkerson Method

Ford\_Fulkerson\_Method( $G, c, s, t$ )

1. Initialize flow to 0;
2. While “possible”
  - (a) Improve flow (push more flow from  $s$  to  $t$ );

# Residual Network

Given a flow  $f$  on a flow network  $G$  the **residual** capacity of an edge  $(u, v)$ ,  $c_f(u, v)$ , is the additional amount of flow that can be send from  $u$  to  $v$  before exceeding the capacity  $c(u, v)$ , i.e.

$$c_f(u, v) = c(u, v) - f(u, v)$$

(Note that flow is defined on pairs of vertices - not edges!)

Given a flow  $f$  on a flow network  $G$  the **residual network**  $G_f$  of  $G$  induced by  $f$  is a flow graph on  $G$  with flow capacities  $c_f(u, v)$ .

(Note that the residual graph might have edges that are not in  $G$ , but if  $(u, v)$  is an edge either  $(u, v)$  or  $(v, u)$  is an edge of  $G$ .)

# Improving Flow

Let  $f$  be a flow function on  $G$ .

Let  $f'$  be a flow function on the residual graph of  $G_f$  induced by  $f$ .

Define the “flow” function  $f + f'$  as

$$(f + f')(u, v) = f(u, v) + f'(u, v)$$

.

**Lemma 1.** *The function  $f + f'$  is a flow function in  $G$ , with flow value*

$$|f + f'| = |f| + |f'|.$$

## Proof.

1. For all  $(u, v) \in E$ ,

$$\begin{aligned}(f + f')(u, v) &= f(u, v) + f'(u, v) \\ &\leq f(u, v) + (c(u, v) - f(u, v)) \\ &= c(u, v)\end{aligned}$$

2. For all  $u, v \in V$ ,

$$\begin{aligned}(f + f')(u, v) &= f(u, v) + f'(u, v) \\ &= -f(v, u) - f'(v, u) \\ &= -(f + f')(v, u)\end{aligned}$$

3. For all  $u \in V - \{s, t\}$ ,

$$\begin{aligned}\sum_{v \in V} (f + f')(u, v) &= \sum_{v \in V} (f(u, v) + f'(u, v)) \\ &= \sum_{v \in V} f(u, v) + \sum_{v \in V} f'(u, v) \\ &= 0 + 0\end{aligned}$$

$$\begin{aligned}|f + f'| &= \sum_{v \in V} (f + f')(s, v) \\ &= \sum_{v \in V} (f(s, v) + f'(s, v)) \\ &= \sum_{v \in V} f(s, v) + \sum_{v \in V} f'(s, v) \\ &= |f| + |f'|\end{aligned}$$

□

Given a graph  $G = (V, E)$  and a flow  $f$  in  $G$ , an **augmenting path**  $p$  is a simple (directed) path connecting  $s$  to  $t$  in the residual graph  $G_f$ .

Let

$$c_f(p) = \text{Min}\{c_f(u, v) \mid (u, v) \in p\}$$

.

**Lemma 2.** *Let*

$$f_p(u, v) = \begin{cases} c_f(p) & (u, v) \in p \\ -c_f(p) & (v, u) \in p \\ 0 & \text{otherwise.} \end{cases}$$

*Then  $f_p$  is a flow function in  $G_f$  with value  $|f_p| = c_f(p) > 0$ .*

**Theorem 1.** *If there is an augmenting path  $p$  in  $G_f$  then there is a flow  $|f + f_p| > |f|$  in  $G$ .*

Ford\_Fulkerson\_Method( $G, c, s, t$ )

1. Initialize  $f(u, v) = 0$  for every edge in  $E$  ;
2. While there exists an augmenting path  $p$  in  $G_f$ 
  - (a) Augment the flow  $f$  with  $f_p$ ;

# The Max-Flow Min-Cut Theorem

An  $(S, T)$ -**cut** in a flow network  $G = (V, E)$  is a partition of  $V$  to  $S$  and  $T = V - S$  such that  $s \in S$  and  $t \in T$ .

Given a flow  $f$  in  $G$ , the **net flow** across a cut  $(S, T)$  is

$$f(S, T) = \sum_{u \in S} \sum_{v \in T} f(u, v).$$

The **capacity** of the cut is

$$c(S, T) = \sum_{u \in S} \sum_{v \in T} c(u, v).$$

**Theorem 2.** *Maximum  $s - t$  flow in  $G$  equals the minimum  $c(S, T)$  over all  $(S, T)$  cuts.*

**Lemma 3.** *Let  $f$  be a flow in  $G$ . For any  $(S, T)$  cut*

$$f(S, T) = |f|.$$

**Proof.**

$$f(S, T) = f(S, V) - f(S, S) \quad (1)$$

$$= f(S, V) \quad (2)$$

$$= f(s, V) + f(S - s, V) \quad (3)$$

$$= f(s, V) \quad (4)$$

$$= |f|. \quad (5)$$

□

**Corollary 1.** *The flow in  $G$  is bounded by the minimum  $(S, T)$ -cut capacity in  $G$ .*

# Max-Flow Min-Cut Theorem

**Theorem 3.** *Let  $f$  be a flow from  $s$  to  $t$  in  $G$ . The following conditions are equivalent:*

- 1.  $f$  is a maximum flow.*
- 2. The residual graph  $G_f$  allows no augmenting paths.*
- 3.  $|f| = c(S, T)$  for some  $(S, T)$ -cut.*

**Proof.**

(1)  $\Rightarrow$  (2): If  $G_f$  has an augmenting path the flow  $f$  can be improved.

(2)  $\Rightarrow$  (3): We construct a an  $(S, T)$  cut as follows:

Let

$$S = \{v \in V \mid \text{there is a path from } s \text{ to } v \text{ in } G_f\}$$

and let  $T = V - S$ .

This partition defines an  $(S, T)$ -cut:

$$s \in S$$

$t \in T$  otherwise there is an augmenting path in  $G_f$ .

For each  $u \in S$  and  $v \in T$  we have  $f(u, v) = c(u, v)$ , otherwise  $(u, v) \in E_f$  and  $v \in S$ .

$f(S, T) = c(S, T)$  and we proved that  $|f| = f(S, T)$ , thus

$$|f| = f(S, T) = c(S, T).$$

(3)  $\Rightarrow$  (1):

We proved that for any  $(S, T)$  cut,  $|f| \leq c(S, T)$ , thus if  $|f| = c(S, T)$  for some cut, it is a maximum flow.  $\square$

# Analysis of the Ford-Fulkerson Algorithm

**Theorem 4.** *Assuming that all the capacities are integral, the run-time of the Ford-Fulkerson algorithm is  $O(|E| \cdot |f|)$  where  $|f|$  is the maximum flow value.*

**Proof.**

Steps 1-3 take  $O(|E|)$  time.

The while loop is executed  $O(|f|)$  times, since each iteration improves the flow by at least 1.

An augmenting path in the residual graph can be found by breadth-first or depth-first search, thus each iteration of the while loop takes  $O(|E|)$  time.

□