Maximum Flow

Road → directed graph to
find the shortest path
from one point to another

can also be interpreted as
a "flow network"

↓

answer questions about
material flows.

source
production
steady
rate

←

sink
consumption
same
rate

flow = rate at which material moves

flow networks can model many problems

liq through pipes

parts through assembly lines

current through electrical networks

information through comm. net.

directed edge in a flow network → conduit/channel for the material

stated capacity

maximum rate at which material can flow.

vertices are conduit junctions

Other than source and sink,
material flows through
the vertices without
collecting in them.

---

rate at which a material
enters a vertex must
equal the rate
at which it leaves

---

flow conservation

Max flow problem → compute the
greatest rate
for shipping material
from the source to
the sink without
violating any capacity

constraints

24.1 → formalize notions of flows and flow networks
define max flow problem

24.2 Ford - fulkerson method.

24.3 Application: undirected bipartite graph.

24.1 Flow networks

Flow networks and flows

A flow network $G = (V, E)$ is a directed graph in which each edge $(u, v) \in E$ has

1. a non-negative <u>capacity</u> $c(u, v) \geq 0$.

2. if $(u, V) \in E$ then $(v, u) \notin E$ — workaround later

for convenience $c(v, u) = 0$.

and 3. we disallow self loops.

→ 4. Each flow network contains two distinguished vertices: a <u>source</u> s and a <u>sink</u> t.

→ 5. Each vertex lies on some path from the source to the sink.

i.e. $\forall\, v \in V$, flow network contains a path

$$s \rightsquigarrow v \rightsquigarrow t$$

Because each vertex other than s has at least one entering edge,

$$\boxed{|E| \geq |V| - 1}$$

Flows

Let $G = (V, E)$ be a flow network with a capacity function c.

Let s be the source of the network, t be the sink.

A flow is a real valued function

$$f : V \times V \to \mathbb{R}$$

that satisfies:

① Capacity constraint: For all $(u, v) \in V$

$$0 \leq f(u, v) \leq c(u, v)$$

flow from one vertex to another must be non-negative and must not exceed the given capacity

② Flow conservation

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

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

Total flow into a vertex other than the source or sink must equal the total flow out of that vertex.

flow in = flow out.

If $(u,v) \notin E$, there can be no flow and $f(u,v) = 0$.

$f(u,v)$ = flow from vertex u to vertex v.

<u>Value $|f|$ of a flow f</u> is defined as:

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

$|f|$ = total flow out of the source − total flow into the source.

Typically a flow network does not have edges into the source

$$\therefore \underbrace{\sum_{v \in S} f(v,s)}_{\text{flow into the source}} = 0$$

can be possible in residual networks

Max-flow problem:

Input: flow network G with source s and sink t

Output: find a flow of max value

<u>Example of flow</u> Lucky Puck company

source s	sink t
factory Vancouver	warehouse Winnipeg

cities ~ vertices

$c(u, v) \longrightarrow$ no. of crates per day

Lucky Puck has no control over routes and capacities, so they cannot alter the flow network

Determine the largest number p of crates per day that they can ship.

→ no point in producing more pucks

Don't care about how long it takes for a given puck to get from the factory to the warehouse.

Only p crates per day

flow in this network → models the flow of shipments

→ capacity constraints

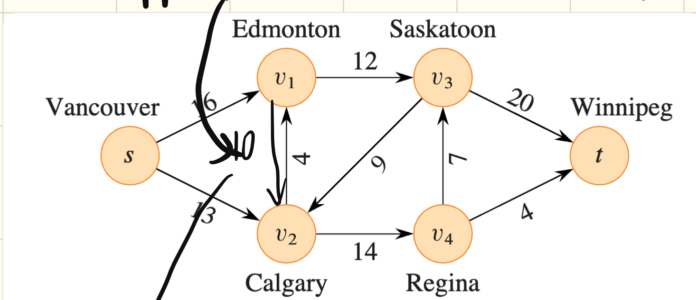
→ flow conservation

↓

if not accumulation

Modeling problems with antiparallel edges

Suppose trucking firm offers



(a)

useful to add

→ problem

$\downarrow\uparrow$

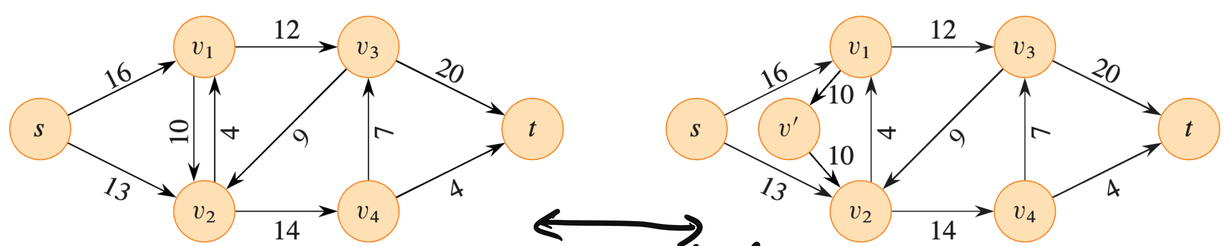
$(v_1, v_2) \in E$

$\Downarrow$ in a flow network

$(v_2, v_1) \notin E$

(v_1, v_2)
(v_2, v_1) } → antiparallel edges

To model a flow problem with antiparallel edges, transform the network into an equivalent one with no antiparallel edges



(a) ⟷ (b)

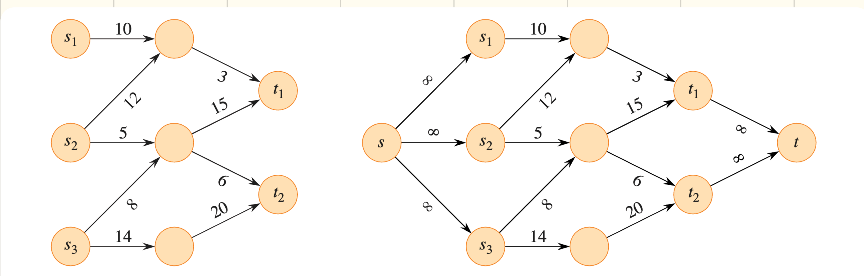
equivalent

max flow same

Networks with multiple sources and sinks

Lucky puck Company might have a set of m factories and n warehouses.

Problem of determining a maximum flow with multiple sources & sinks $\xrightarrow[\text{to}]{\text{Reduces}}$ ordinary max-flow problem



(a) (b)

Supersource s with directed edges (s, s_i)

$c(s, s_i) = \infty$

supersink (t_i, t)

$c(t_i, t) = \infty$

24.1-1

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

remains same

every vertex must obey flow conservation

24.1-2

Flow properties and definitions for multiple sources:

$$f : V \times V \longrightarrow \mathbb{R}^{+}$$

such that

① Capacity constraint

$\forall u, v \in V$

$$0 \leqslant f(u, v) \leqslant c(u, v)$$

② Flow conservation

$\forall \quad u \in V - \{S \cup T\}$

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

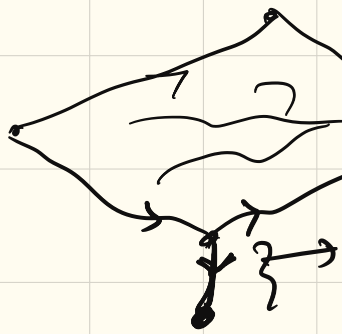
$$|f| = \sum_{s \in S} \left(\sum_{v \in V} f(s, v) - \sum_{v \in V} f(v, s) \right)$$

any flow

equivalent to single source with supersink supersource?

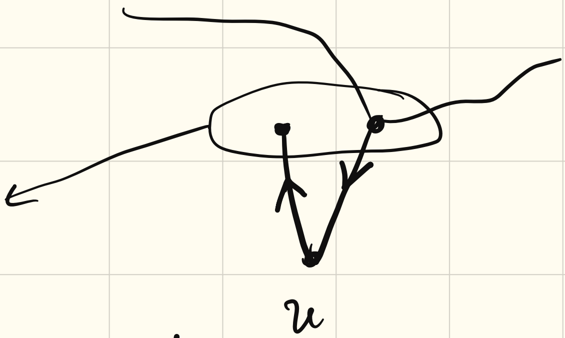
- How to prove?

24.1 -3



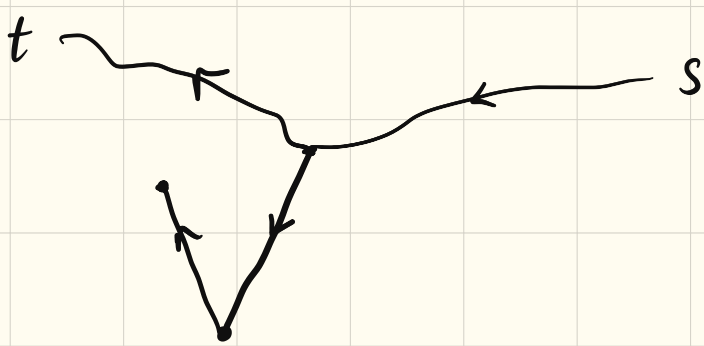
can be made zero

Has to be made zero because flow conservation?



If no path $s \rightsquigarrow u \rightsquigarrow t$ exists, one of these must also violate the same

X NO?



24.1-4

$$\alpha f : V \times V \to \mathbb{R}$$

$$\alpha f\,(u,v) = \alpha \cdot f(u,v)$$

$$f_1,\quad f_2 \rightsquigarrow \text{flows}$$

$$\underbrace{\alpha f_1 + (1-\alpha) f_2}_{\text{also flow?}}$$

$$0 \le f_1(u,v) \le c(u,v)$$
$$0 \le \alpha f_1 \le \alpha c$$
$$0 \le (1-\alpha) f_2 \le (1-\alpha) c$$

$$\oplus$$

$$0 \le \alpha f_1 + (1-\alpha) f_2 \le c$$

Flow conservation

$u \neq s, t$

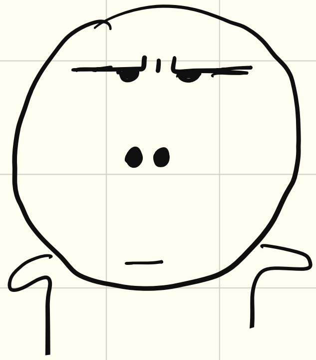
$$\sum_{v \in V} f_1(u, v) = \sum_{v \in V} f_1(v, u)$$

$$\alpha \left(\qquad \right) = \alpha \left(\qquad \right)$$

$$1 - \alpha \left(\qquad \right) = 1 - \alpha \left(\qquad \right)$$

$$\sum_{v \in V} \left(\alpha f_1 + (1-\alpha) f_2 \right)(u, v) = \sum_{v \in V} \left(\alpha f_1 + (1-\alpha) f_2 \right)(v, u)$$

24.1 - 5



24.1 - 6

Model the map as flow network

capacity of all edges = 1

Solution $\longrightarrow$ find flow, if

$|f| \geq 2$

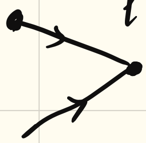
✓

If not, not possible

24.1 - 7

$\frac{\sum f(u,v)}{l(v)}$ vertices

$l(v)$



For each vertex v, transform it to an edge (v, v') with capacity $l(v)$.

G' has $2V$ vertice

has $V+E$ edges

The Ford-Fulkerson Method

→ not an algorithm

→ several implementations with different running times.

③ important ideas

→ residual networks

→ augmenting paths

→ cuts

→ max-flow min-cut theorem

characterizes max-flow in terms of cuts of the flow network.

Ford-Fulkerson

↳ Iteratively increases the value of flow

→ starts with $f(u,v) = 0 \quad \forall \; u, v \in V$

$$|f|_{initial} = 0$$

→ each iteration ⟶ increase flow value in G by finding an "augmenting path" in an associated residual network G_f

edges of the augmenting path in G_f

↓

indicate which edges in G to update the flow in order to increase the flow value

Each iteration increases $|f|$ } but $f(u,v)$ may increase or <u>decrease</u> for a particular edge

doing so may increase flow for other edges

FORD-FULKERSON-METHOD (G, s, t) → repeatedly augment the flow until G_f has no augmenting path left

1. initialise flow to 0
2. while $\exists$ an augmenting path in the residual network G_f
3. augment flow f along p.
4. return f → Max-flow min-cut theorem shows that upon termination, this process yields a maximum flow.

Residual networks

residual network G_f → consists of edges whose capacities represent how the flow can change on edges of G.

An edge of the flow network can admit an additional flow = capacity − current flow on that edge

If +ve, this edge goes into G_f with a "residual capacity"

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

The only edges of G that belong to G_f $\rightsquigarrow$ those that can admit more flow.

G_f $\rightsquigarrow$ can also contain edges that are not in G.

algorithm might decrease the flow on a particular edge

In order to represent a possible decrease in the +ve flow $f(u, v)$ on an edge in G,

$\downarrow$

G_f contains an edge (v, u) with residual capacity

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

sending flow back $\equiv$ decreasing flow on the edge

necessary operation in many algo

Residual capacity $c_f(u, v)$

$$c_f(u, v) = \begin{cases} c(u, v) - f(u, v) & \text{if } (u, v) \in E \\ f(v, u) & \text{if } (v, u) \in E \\ 0 & \text{o/w} \end{cases}$$

In flow network, $(u, v) \in E \Rightarrow (v, u) \notin E$, so only one case is true e.g. :

$$c(u,v) = 16$$

$$f(u,v) = 11$$

then

$C_f(u,v) = 5$

|||

$f(u,v)$ can increase by upto $C_f(u,v) = 5$ units before exceeding the capacity on edge (u,v)

OR (not exclusive)

$C_f(v,u) = 11$

Up to 11 units of flow can return from $v \rightarrow u$

Residual network of G induced by f is $G_f = (V, E_f)$ where

$$E_f = \{ (u,v) \in V \times V : c_f(u,v) > 0 \}$$

Each edge of the residual network (residual edge) can admit a flow that is greater than 0.

edges in E_f are either edges in E or their reversals and thus:

$$|E_f| \leq 2|E|$$

Residual network ~ flow network

antiparallel edges allowed.

flow in residual network $\longrightarrow$ same defn but w.r.t c_f

A flow in a residual network provides a roadmap for adding flow to the original network.

If $f \longrightarrow$ flow in G

$f' \longrightarrow$ flow in G_f $\longrightarrow$ corresponding residual network

$f \uparrow f'$: augmentation of flow f by f'

$$f \uparrow f' : V \times V \longrightarrow \mathbb{R}$$

$$(f \uparrow f')(u,v) = \begin{cases} f(u,v) + f'(u,v) - f'(v,u) & \text{if } (u,v) \in E \\ 0 & \text{o/w} \end{cases}$$

Intuition of augmentation:

flow on (u,v) $\rightsquigarrow$ increases by $f'(u,v)$
but decreases by
$f'(v,u)$

Pushing flow on reverse edge → cancellation

Augmenting a flow in G by a flow in G_f yields a new flow in G with a greater flow value:

① $f \uparrow f'$ $\rightsquigarrow$ is a flow

② $|f \uparrow f'| = |f| + |f'|$

$f \uparrow f'$ is a flow

$f \uparrow f' : V \times V \to \mathbb{R}$

① Capacity constraints

TP: $(u,v) \in E$

$$0 \le f(u,v) + f'(u,v) - f'(v,u) \le c(u,v)$$

$(u,v) \in E$

$$\Rightarrow f'(u,v) = c(u,v) - f(u,v)$$

$$f'(v,u) = 0 \; \times$$

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

$$f'(v,u) \le f(u,v)$$

$$f \uparrow f' \geq f'(u,v) \geq 0$$

Eh boring I am skipping proof.

Augmenting Paths

Given a flow network G and a flow f

an augmenting path p is a simple path from s to t in the residual network G_f.

→ flow on an edge (u, v) of an augmenting path may increase by upto $c_f(u, v)$ without violating the capacity of whichever of (u, v) and (v, u) belongs to the original flow network G.

Maximum amount by which we can ↑ flow on each edge in an augmenting path p → residual capacity of p.

Residual capacity of p := $\min\{ c_f(u,v) :$
$c_f(p)$ $\qquad (u,v) \in p \}$

Let $G = (V, E)$ be a flow network, let f be a flow in G, let p be an augmenting path in G_f.

Define a function $f_p : V \times V \longrightarrow \mathbb{R}$ by

$$f_p(u,v) = \begin{cases} c_f(p) & \text{if } (u,v) \in p \\ 0 & \text{o/w} \end{cases}$$

Then, f_p is a flow in G_f with $|f_p| = |c_f(p)| > 0$

Capacity constraint:

$$f_p(u,v) = \begin{cases} \min\{c_f(u,v),\ (u,v) \in p\} & (u,v) \in p \\ 0 & o/w \end{cases}$$

$$0 \leq f_p(u,v) \leq c_f(u,v)$$

Flow conservation

$\forall\ u \in V - \{s,t\}$ in G_f

$$\sum_{v \in V} f_p(v,u) = \sum_{v \in p} f_p(v,u)$$

$$= \sum_{v \in p} c_f(p) = c_f(p) \times \text{path length}$$

$$\sum_{v \in V} f_p(u,v) = \sum_{v \in p} f_p(u,v)$$

$$= c_f(p) \times \text{path length}$$

Flow value

$$|f_p| = \underbrace{\sum_{v \in V} f_p(s, v)}_{\text{simple path} \therefore 1 \text{ vertex}} - \underbrace{\sum_{v \in V} f_p(v, s)}_{= 0 \because \text{simple path}}$$

$$= f_p(s, v) \quad (s, v) \in p$$

$$= c_f(p)$$

Corollary

Let $G = (V, E)$ be a flow network, let f be a flow in G, and let p be an augmenting path in G_f. Let f_p be defined as above, and suppose that f is augmented by f_p. Then the function $f \uparrow f_p$ is a flow in G with value $|f \uparrow f_p| = |f| + |f_p| > |f|$

Cuts of flow networks

Ford-Fulkerson method → repeatedly augment flow along augmenting paths until you find a max flow

How do u know that when the algorithm terminates, you have a max flow?

Max-flow Min-cut theorem

flow is maximum iff its residual network contains no augmenting path.

Cut $\longrightarrow$ A cut (S, T) of flow network $G = (V, E)$ is a partition of V into S and $T = V - S$ such that $s \in S$ and $t \in T$.

If f is a flow, then the net flow $f(S, T)$ across the cut (S, T) is defined to be

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

The capacity of the cut (S, T) is

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

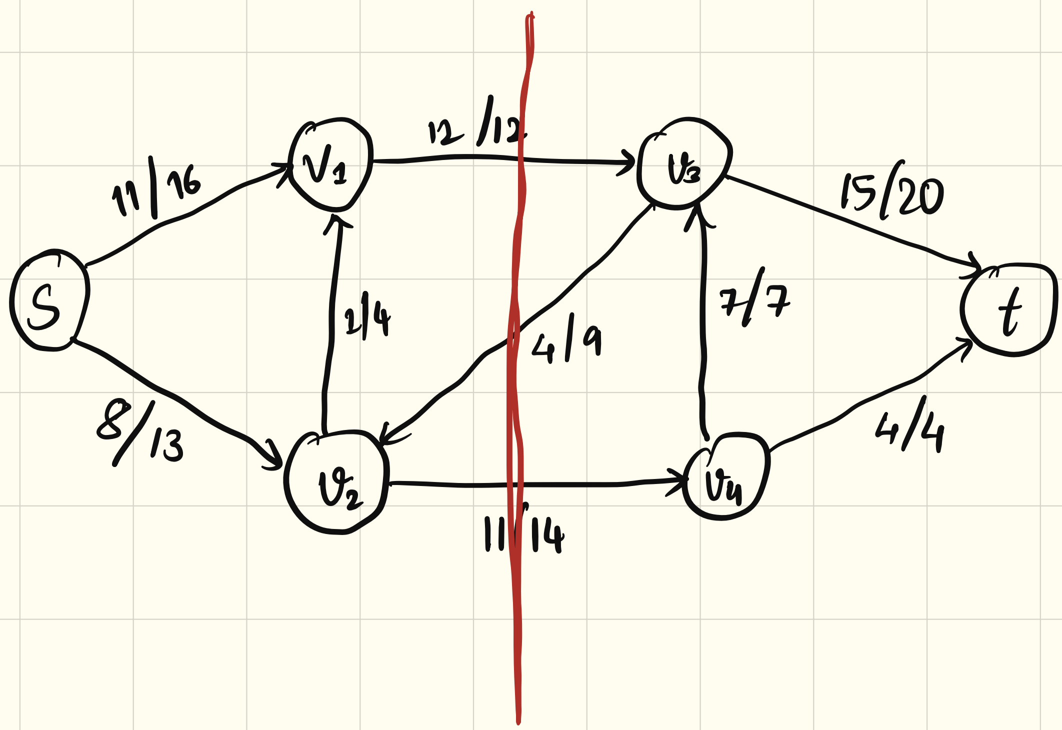
Minimum cut of a network is a cut whose capacity is minimum over all cuts of the network

intentional & important

flow across a cut : $S \to T$
$T \to S$

capacity of a cut : only $S \to T$

e.g :



cut = $(\{S, v_1, v_2\}, \{v_3, v_4, t\})$

net flow $= f(v_1, v_3) + f(v_2, v_4)$
$- f(v_3, v_2)$

$= 12 + 11 - 4$

$= 19$

capacity $= \cancel{12 + 9 + 14}$

$= \cancel{35}$

$= c(v_1, v_3) + c(v_2, v_4)$

$= 12 + 14$

$= 26.$

Lemma

Let f be a flow in a flow network G with source s and sink t, and let (S, T) be any cut of G. Then, the net flow across (S, T) is

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

Proof: नहीं

Corollary

The value of any flow f in a flow network G is bounded from above by the capacity of any cut of G.

Proof: Let (S, T) be any cut of G and let f be any flow.

By above lemma,

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

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

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

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

$$= c(S, T)$$

Max-flow min-cut theorem

If f is a flow in a flow network $G = (V, E)$ with source s and sink t, then the following conditions are equivalent:

1. f is a maximum flow in G.
2. The residual network contains no augmenting paths.
3. $|f| = c(S, T)$ for some cut (S, T) of G.

The Basic Ford - Fulkerson Algorithm

FORD - FULKERSON (G, s, t)

1 for each edge $(u,v) \in G.E$

2 $\quad (u,v).f = 0$

3 while $\exists$ a path p from s to t in the residual network G_f

4. $\quad c_f(p) = \min\{ c_f(u,v) : (u,v) \text{ is in } p \}$

5. $\quad$ for each edge (u,v) in p

6. $\quad\quad$ if $(u,v) \in G.E$

7. $\quad\quad\quad (u,v).f = (u,v).f + c_f(p)$

8. $\quad\quad$ else $(u,v).f = (u,v).f - c_f(p)$

9 return f

1 – 2 ⇝ initialize flow to 0

3 – 8 – while loop repeatedly finds an augmenting path p in G_f and augments flow f along p by residual capacity $c_f(p)$

6 – 8 – update the flow

Analysis of Ford - fulkerson

The running time of FORD-FULKERSON depends on:

augmenting path p and how it is found (line 3)

If edge capacities are irrational numbers → it is possible to choose an augmenting path so the algorithm never terminates.

If the algorithm finds the augmenting path using a BFS → polynomial time