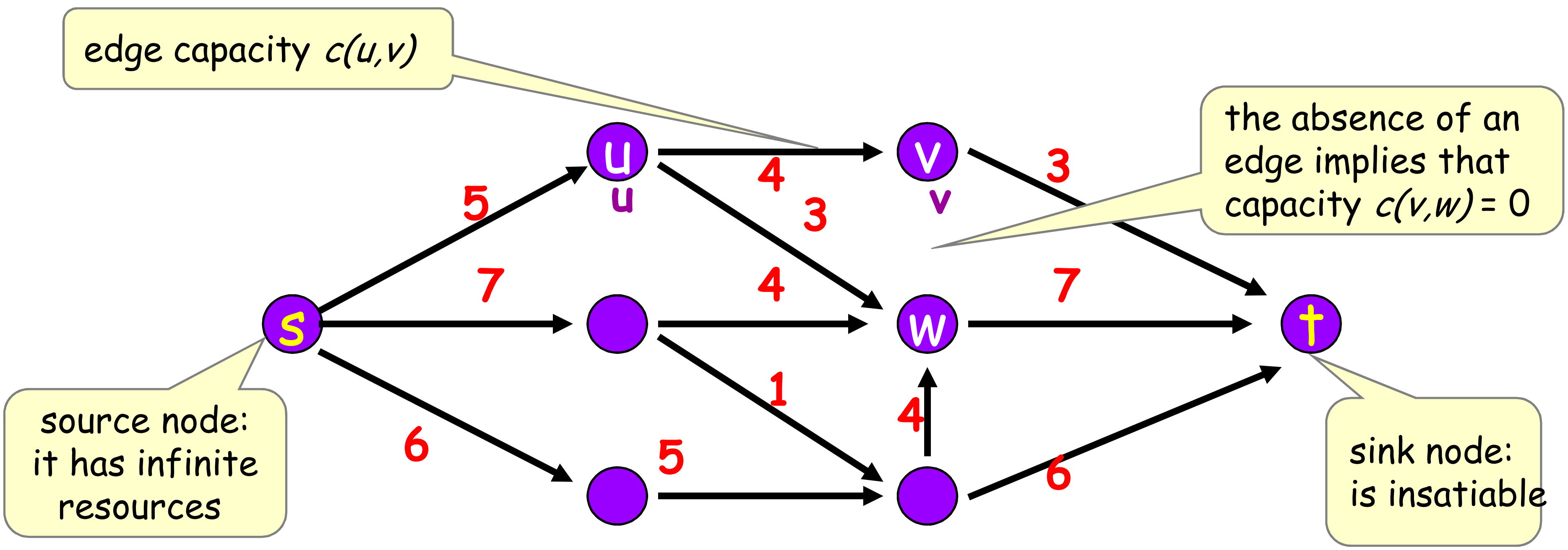
flow networks

- general characteristics of these networks:
  - source : "materials" are produced at a steady rate
  - sink : "materials" are consumed at the same rate
  - flows through conduits are constrained to max values
- applications
  - liquid flow through pipes
  - current flow through an electrical circuit
  - information in a communications network
  - production factory with various tools
  - heat conduction through a material
  - controlling network/internet traffic

- *maximum flow problem* : what is the largest flow of materials from source to sink that does not violate any capacity constraints?

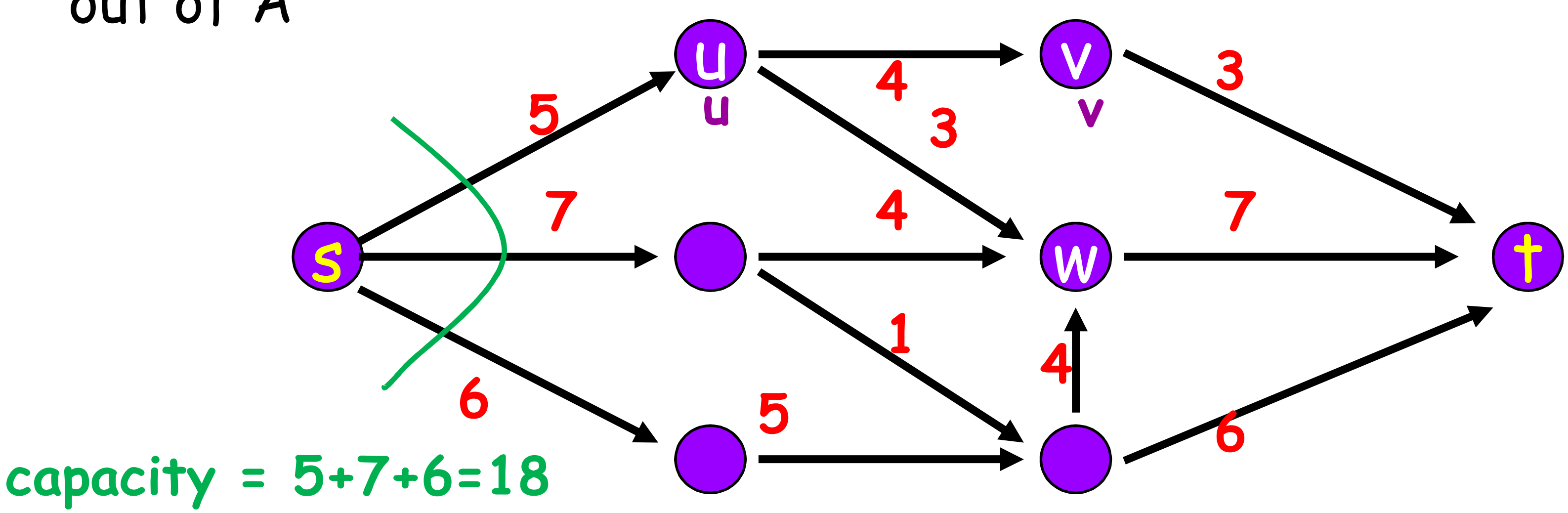
a flow network

- is a directed graph $G=(V, E)$ where each edge $(u,v) \in E$ has a non-negative capacity $c(u,v)$.
- also is specified a source node s and sink node t.
- for every vertex $v \in$ E
  there is a path from s through v to the sink node t.
    - this implies that the graph is connected.



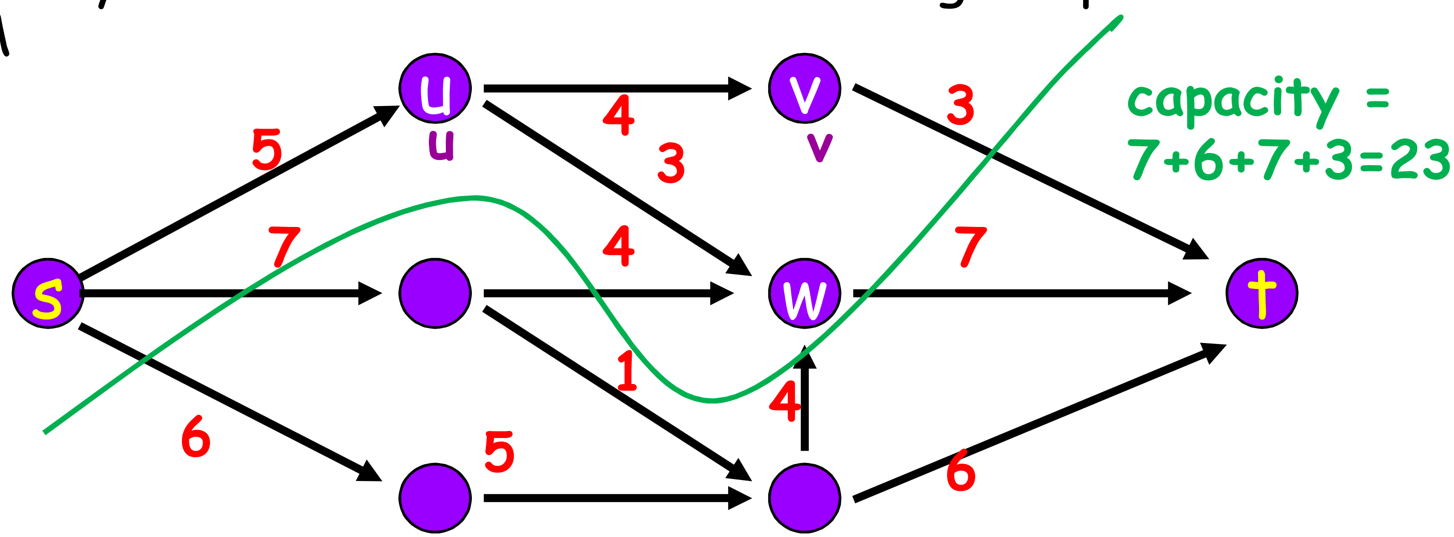
a flow network

- is a directed graph $G=(V, E)$ where each edge $(u,v) \in E$ has a non-negative capacity $c(u,v)$.
- also is specified a source node s and sink node t.
- for every vertex $v \in$ E
  there is a path from s through v to the sink node t.
  - this implies that the graph is connected.
- an s-t cut is a partitioning of V in two blocks, A and B, with $s \in$ A and $t \in$ B
- the capacity of an s-t cut is the sum of edge capacities out of A



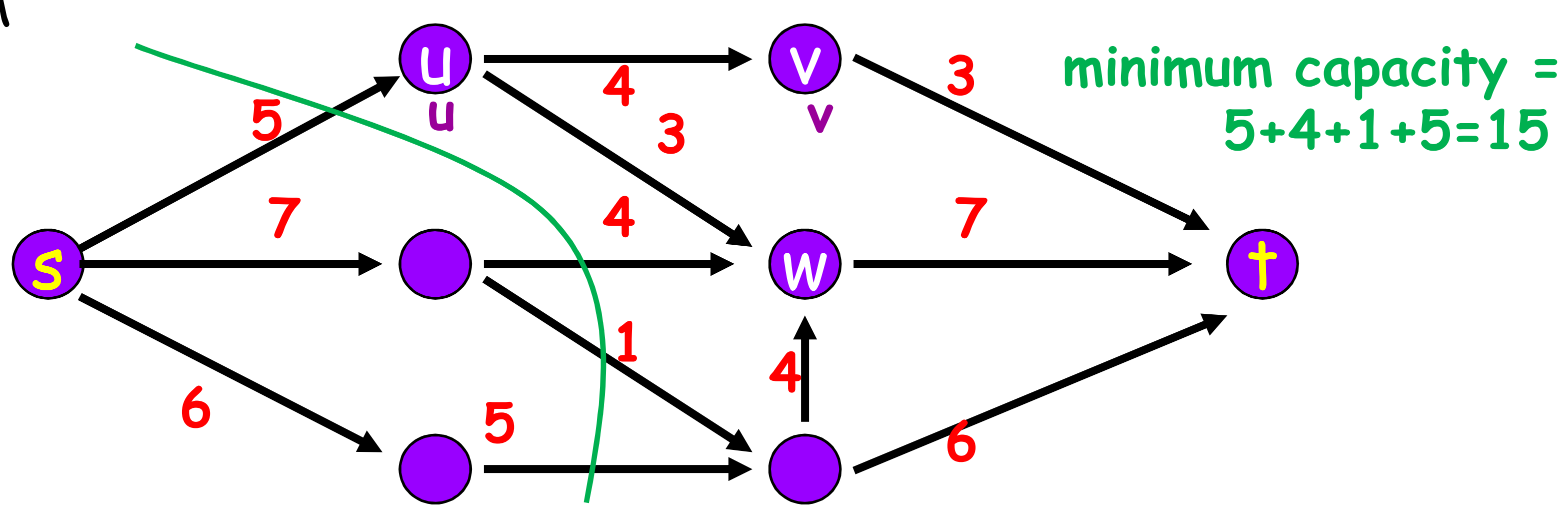
a flow network

- is a directed graph $G=(V, E)$ where each edge $(u,v) \in E$ has a non-negative capacity $c(u,v)$.
- also is specified a source node s and sink node t.
- for every vertex $v \in$ E
  there is a path from s through v to the sink node t.
  - this implies that the graph is connected.
- an s-t cut is a partitioning of V in two blocks, A and B, with $s \in$ A and $t \in$ B
- the capacity of an s-t cut is the sum of edge capacities out of A



a flow network

- is a directed graph $G=(V, E)$ where each edge $(u,v) \in E$ has a non-negative capacity $c(u,v)$.
- also is specified a source node s and sink node t.
- for every vertex $v \in$ V there is a path from s through v to the sink node t.
  - this implies that the graph is connected.
- an s-t cut is a partitioning of V in two blocks, A and B, with $s \in$ A and $t \in$ B
- the capacity of an s-t cut is the sum of edge capacities out of A



minimum cut problem: find an s-t cut of minimum capacity !

flows

do not overload the capacity of each edge

- the flow $f : V \times V \rightarrow R$ satisfies the following:
  - capacity constraints: for all u,v , we require $f(u,v) \leq c(u,v)$
  - skew symmetry: for all u,v , we require $f(u,v) = - f(v,u)$

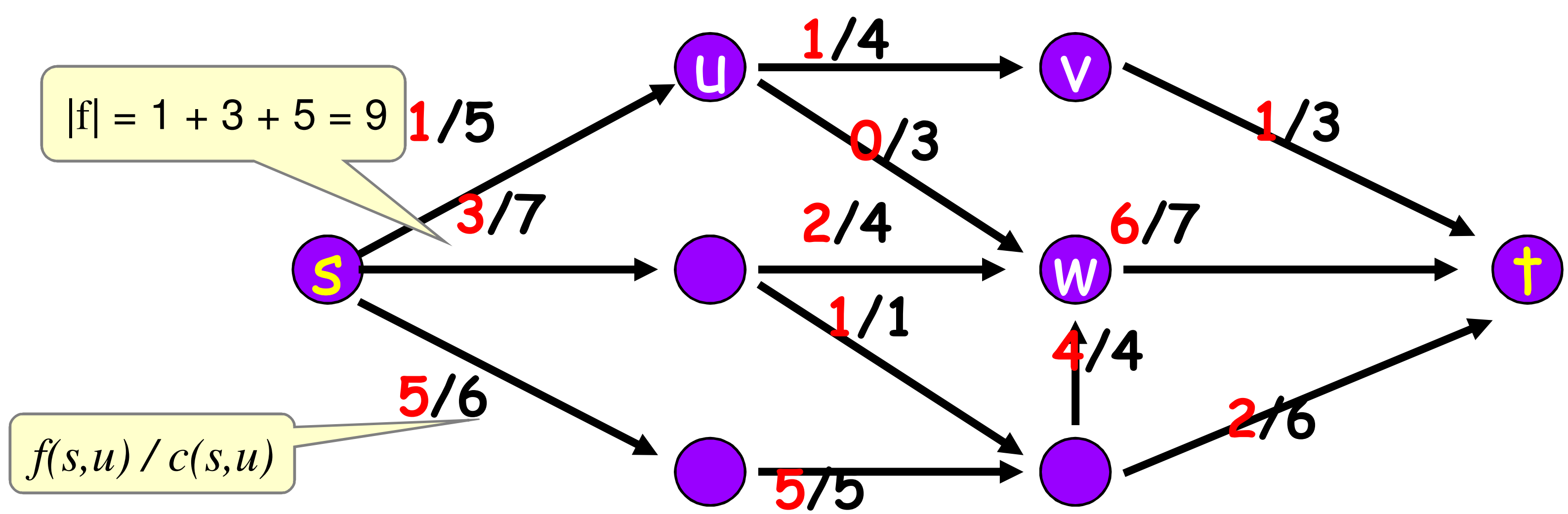
    just math
  - flow conservation: for all $u \in V \setminus \{s,t\}$

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

    what flows into a vertex, must also flow out!
  - net flow across a cut: $\sum_{e \text{ out of } A} f(e) - \sum_{e \text{ into } A} f(e)$
  - flow value: $|f| = \sum_{v \in V} f(s,v)$

    the total flow out of the source
- maximum flow problem:

  find the maximum flow value



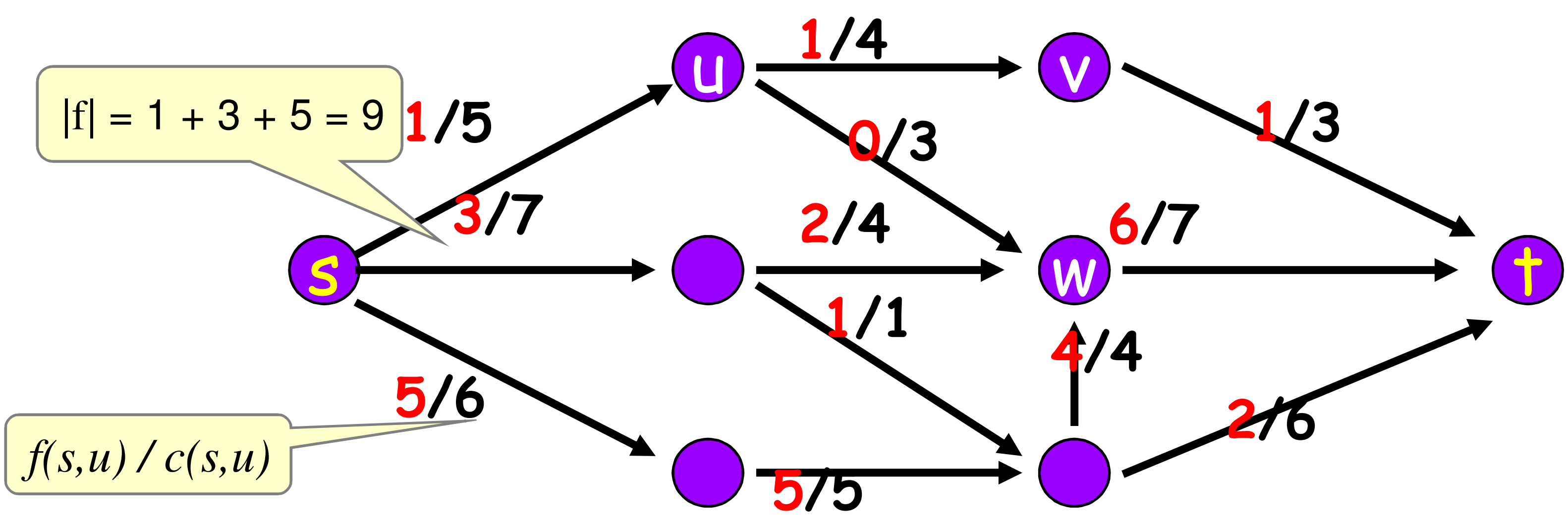
flows and cuts

flow value lemma: let f be any flow and (A,B) any s-t cut ,
then the net flow across the cut is equal to the flow value:

$$\sum_{e \text{ out of } A} f(e) - \sum_{e \text{ into } A} f(e) = \sum_{v \in V} f(s,v) = |f|$$

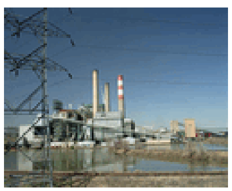
weak duality: let f be any flow and (A,B) any s-t cut ,
then the flow value is at most the capacity of the cut

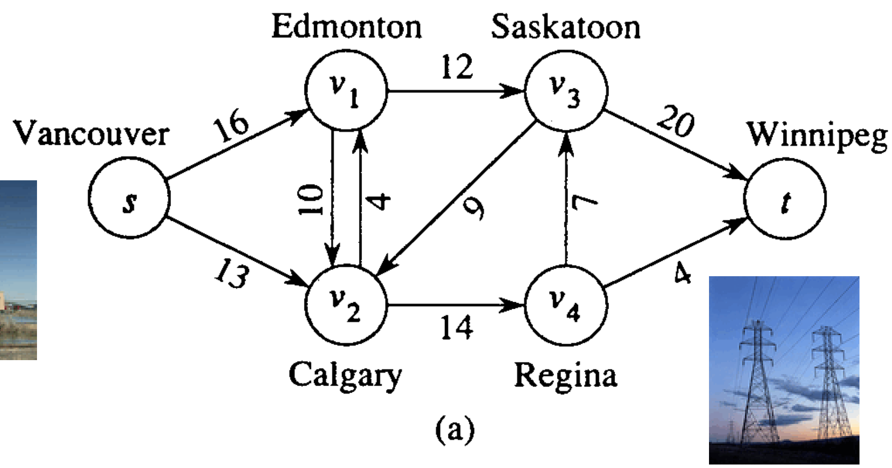
optimality certificate: let f be any flow and (A,B) any s-t cut :
if the flow value is equal to the capacity of the cut,
then the flow is maximum and the capacity is minimum



an example of a flow network

- suppose we operate a electrical power distribution company in Canada.
- the electrical power is generated in Vancouver, and as much as possible power is to be transported to Winnipeg:

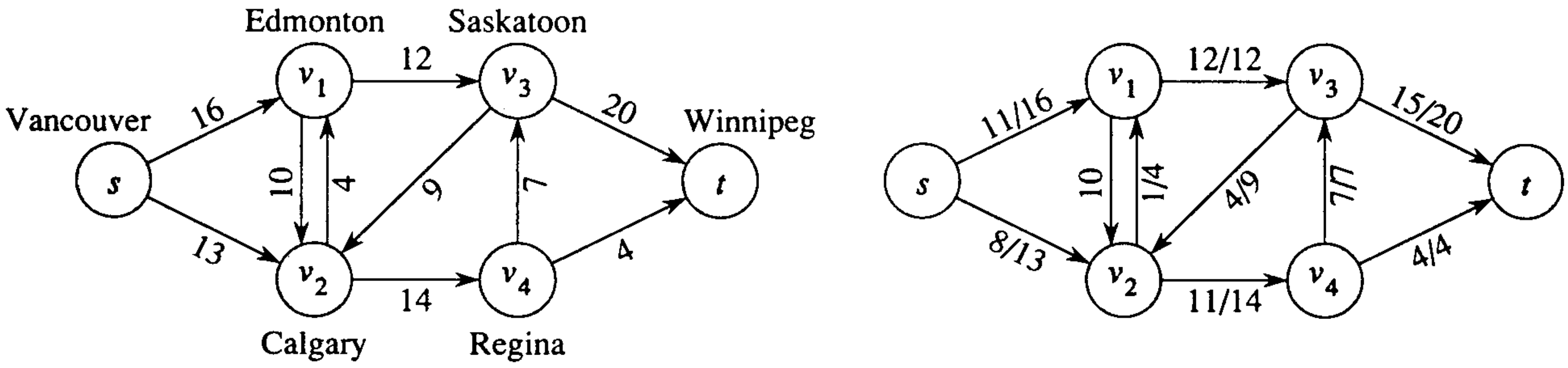




- high-voltage cables connect the cities, each of the cables has a maximum transport capacity in megawatt
- note that between Edmonton and Calgary two cables exist:
  - $c(v1, v2) = 10$ that can transport 10 megawatts from Edmonton to Calgary
  - $c(v2,v1) = 4$ with a capacity of 4 megawatt in the opposite direction.

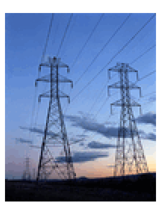
a possible flow assignment

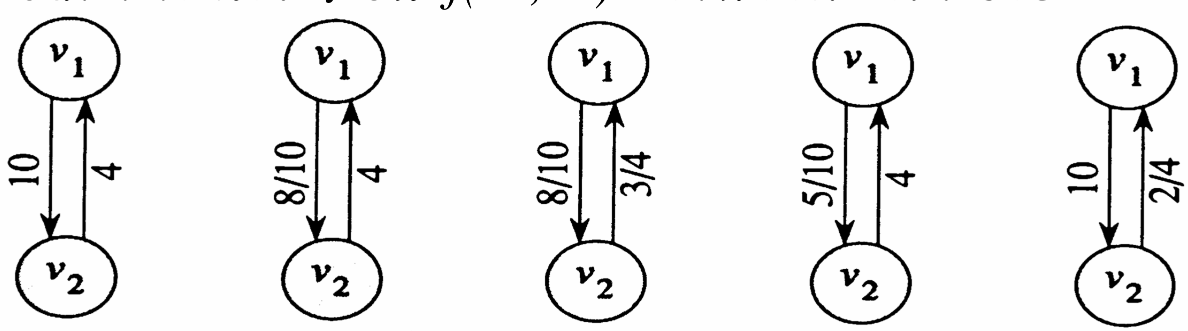
- the figure on the right shows a possible flow f where $|f| = 19$, meaning 19 megawatts are transported from Vancouver to Winnipeg:



(a) (b)

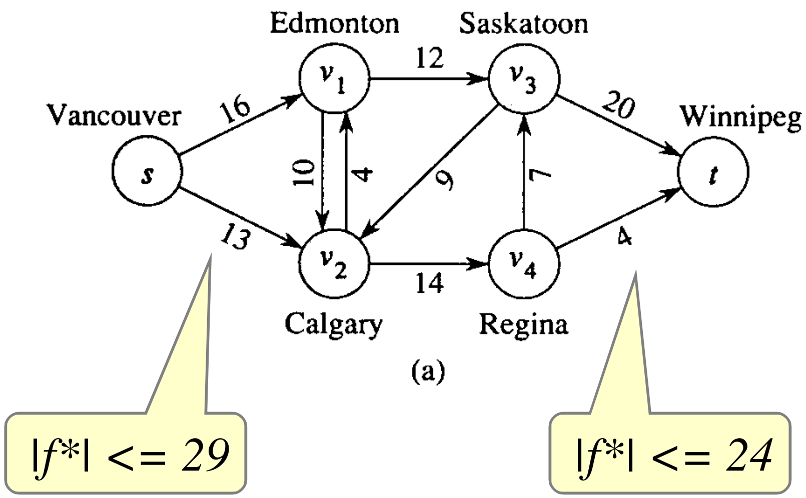
- if there is more than one edge between two vertices, certain flows in the edges can cancel out: we care about the ***net flow*** $f(v1,v2)$ between vertices:

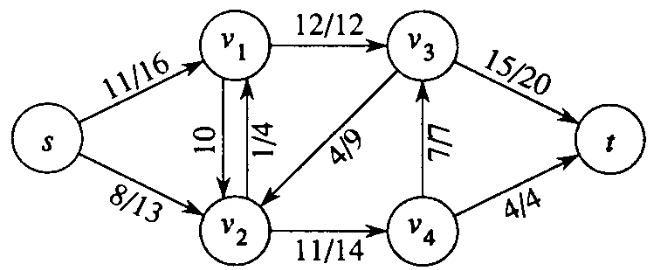




$f(v1,v2) = 0$ | $f(v1,v2) = 8$ | $f(v1,v2) = 5$ | $f(v1,v2) = 5$ | $f(v1,v2) = -2$

now find the maximum flow $|f^*|$





A possible flow with $|f| = 19$

- if we look at the edges connected to the source and sink:
  - the flow out of Vancouver is never > 16 + 13 = 29
  - the flow into Winnipeg is never > 20 + 4 = 24

- so… is the
  maximum flow through the network then 24 MW????

the ford-fulkerson method (1962)

- is based on 3 bright ideas:
  - **residual flow networks**
    the graph that shows where extra capacity might be found
  - **augmenting paths**
    paths along which extra capacity is possible
  - **cuts**
    used to characterize the flow upper bounds in a network
- the basic method is iterative,
  starting from a network with 0 flow:

```
FORD-FULKERSON-METHOD(G, s, t)
initialize flow f to 0
while there exists an augmenting path p
      do augment flow f along p
return f
```

bright idea 1: residual networks

- the residual network is a graph with the same vertices as the flow network, yet
- its edges capture the surplus capacity, that is the amount of additional flow we can send through before saturating the edge.
- so:

capacity

flow

we maintain capacity in each direction!

residual capacity

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

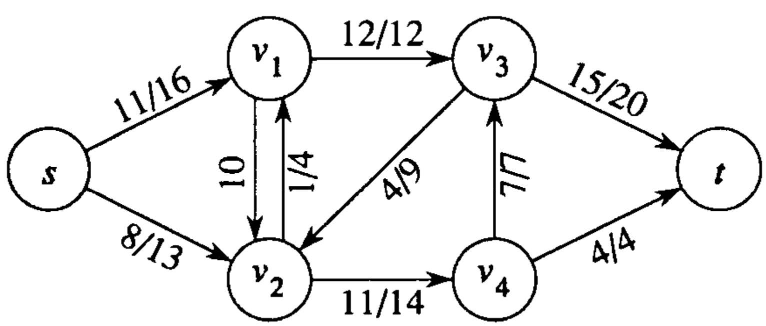
and

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

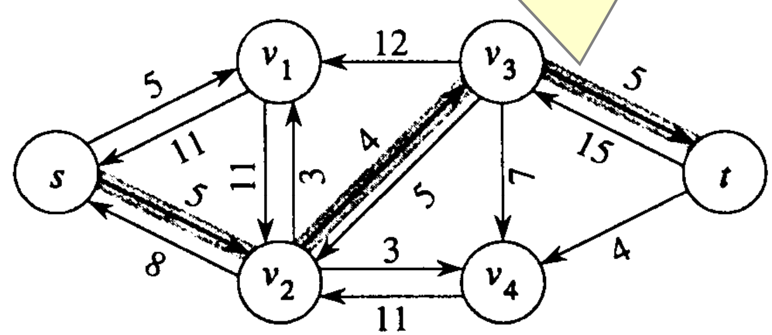
only include edges with a surplus!

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

5 more megawatts could be sent from v3 to t, and we could take away 15 before we hit 0



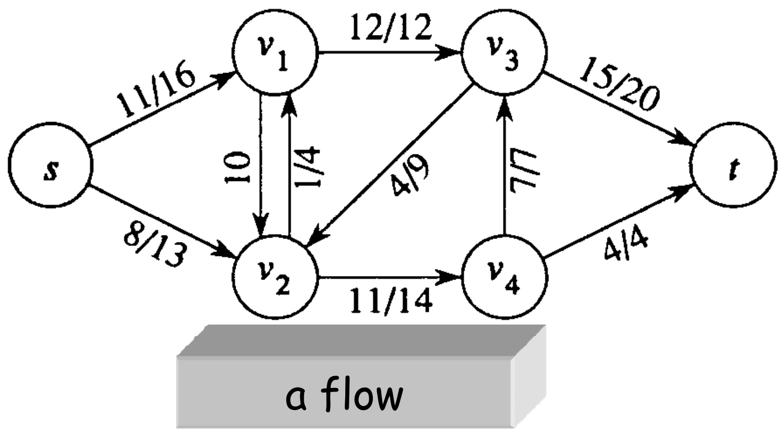
a flow

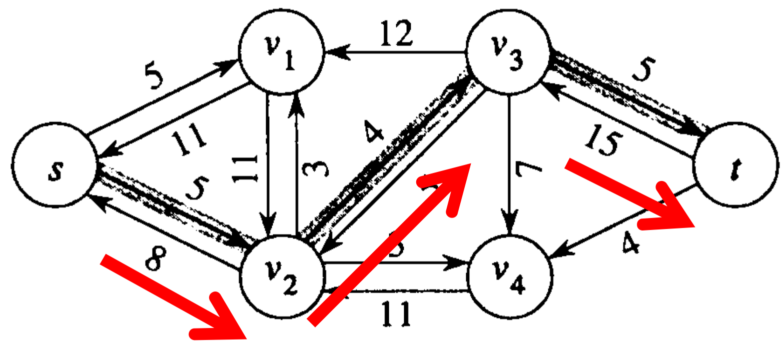


its residual network

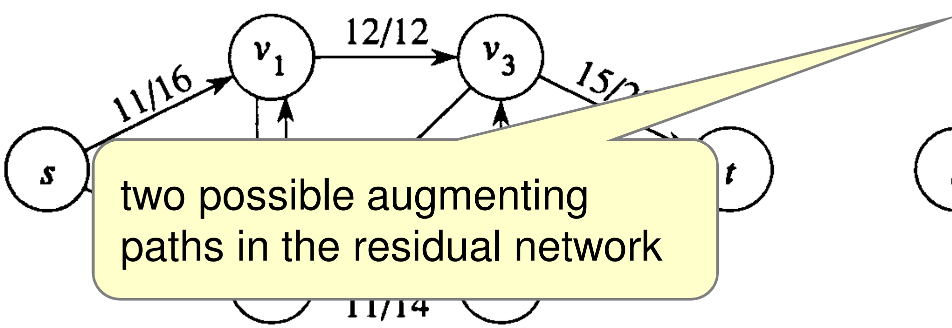
bright idea 2: augmenting paths

- any arbitrary simple path from source to sink that we can find in the residual network shows a way to increase the flow in our network!
- this augmenting path is a simple path from s to t in the residual network.

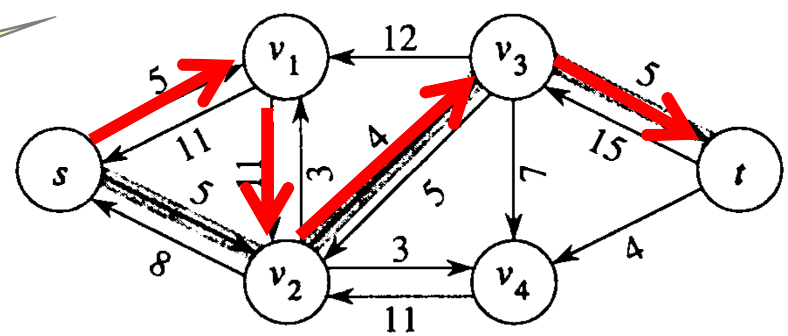




(b)



(a)



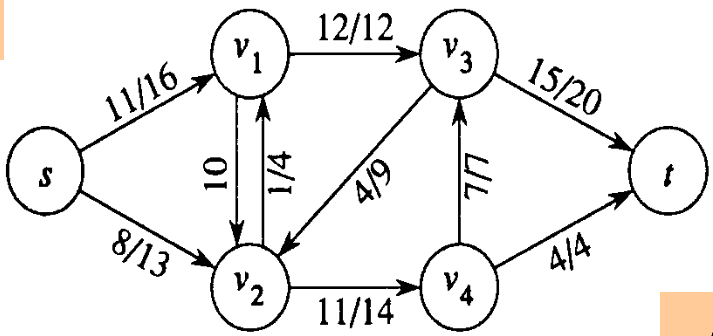
(b)

the residual capacity

- the **residual capacity** $c_f(p)$ of an augmenting path p is the maximum additional flow we can allow along the augmenting path, so:
  - $c_f(p) = min\ \{c_f(u,v) : (u,v)\ is\ on\ p\ \}$

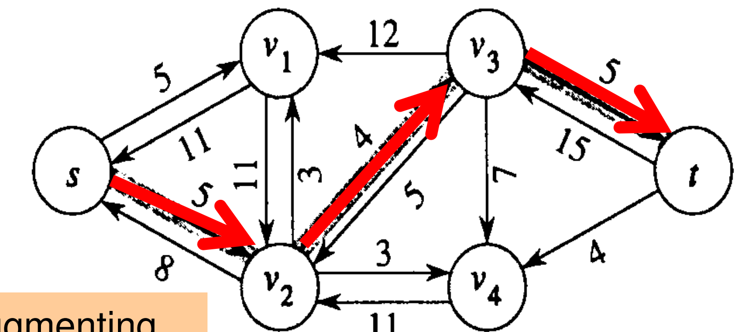
The weakest link sets the residual capacity!

A flow



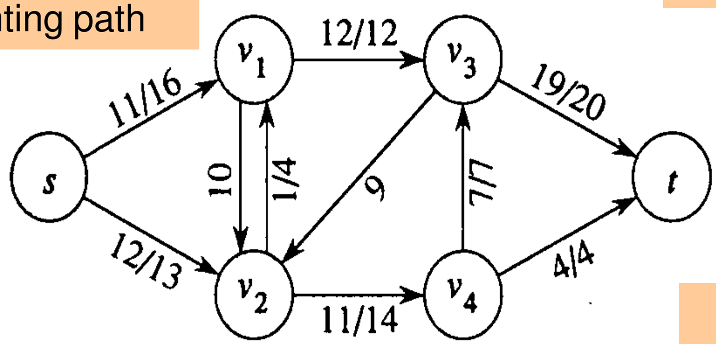
(a)

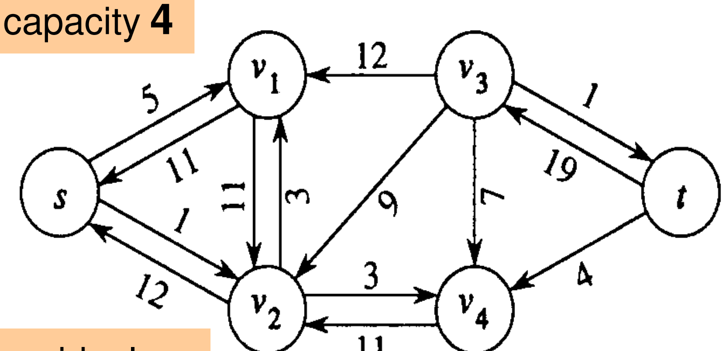
An augmenting path for this flow with residual capacity **4**



(b)

The flow after increasing the above flow by **4** over the augmenting path





(d)

The residual network after this

ford-fulkerson iterative method

FORD-FULKERSON(G, s, t)

for each edge $(u, v) \in E[G]$
    do $f[u, v] \leftarrow 0$
        $f[v, u] \leftarrow 0$
while there exists a path p from s to t in the residual network G_f
    do $c_f(p) \leftarrow \min\{c_f(u, v) : (u, v) \text{ is in } p\}$
        for each edge (u, v) in p
            do $f[u, v] \leftarrow f[u, v] + c_f(p)$
                $f[v, u] \leftarrow -f[u, v]$

Start by setting all flows to 0

Pick some augmenting path p which is a path in the residual network

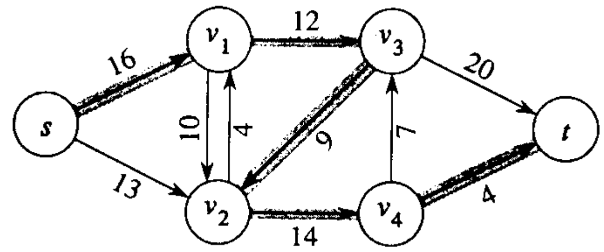
Determine its residual capacity

And increase the flow over the augmenting path by the residual capacity

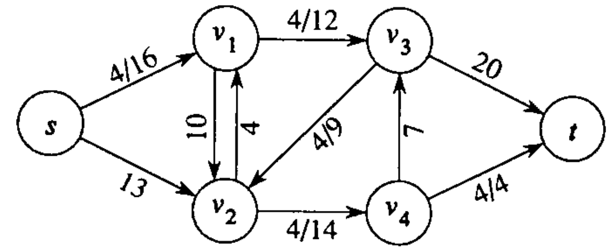
- the iteration stops when no augmenting path can be found.
- we will see that this results in a maximum flow.

Now do it yourself

(a)



Residual network of the initial flow graph with an an augmenting path

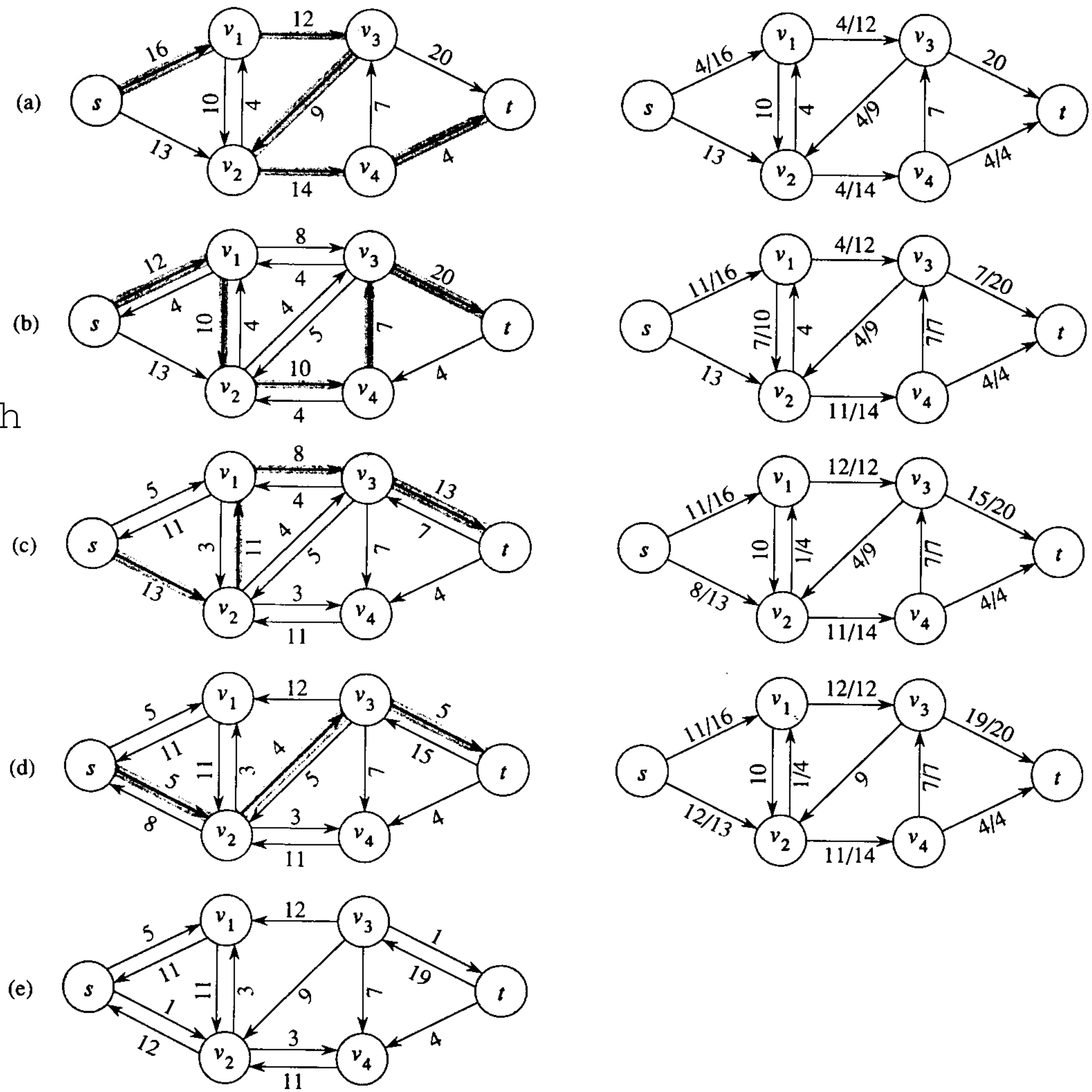


Flow graph after augmenting path is applied

- set flow to 0
- while (augmenting path in residual network) {
    - calculate residual capacity
    - Update flow over augmenting path
- }

Now do it yourself

- set flow to 0
- while (augmenting path in residual network) {
  - calculate residual capacity
  - Update flow over augmenting path
- }



max-flow min-cut theorem

augmenting path theorem: flow f is a maximum flow fif the residual network has no augmenting path

max-flow min-cut theorem: the value of the maximum flow is equal the capacity of the minimum cut

proof: consider the following statements:

1. there exists a cut (A,B) with capacity equal to value of flow f
2. the flow f is a maximum flow
3. there is no augmenting path in the residual network of f

1 -> 2: this followed from weak duality (optimality certificate)

2 -> 3: show the contrapositive: if there were an augmenting path, f can be improved by sending an additional flow along that path

3 -> 1: let f be a flow leaving no augmenting paths in the residual network
let A be the set of vertices reachable in the residual network,
then s is in A and t is not in A

$$|f| = \sum_{v \in V} f(s,v) = \sum_{e \text{ out of } A} f(e) - \sum_{e \text{ into } A} f(e) = \sum_{e \text{ out of } A} c(e) - 0$$

running time for integer capacities

assumption: all capacities are integers c with $1 \leq c \leq C$

invariant: every flow value and every residual capacity
remains an integer throughout the algorithm

termination: the algorithm terminates in at most $|f^*|$ steps
where f^* is an optimal flow

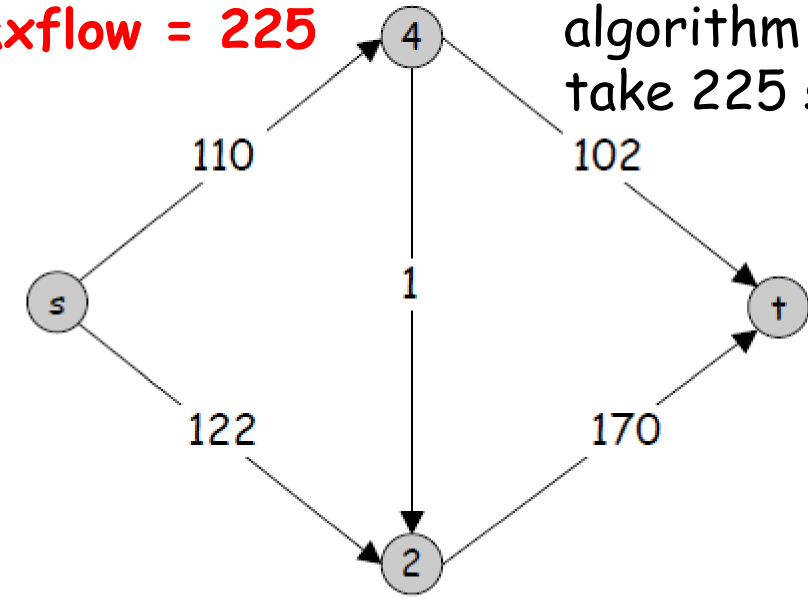
integrality theorem: if all capacities are integers,
then there exists a maximum flow f
for which all edges have integer flows

special case: if $C=1$, then the algorithm runs in O(|V||E|) time

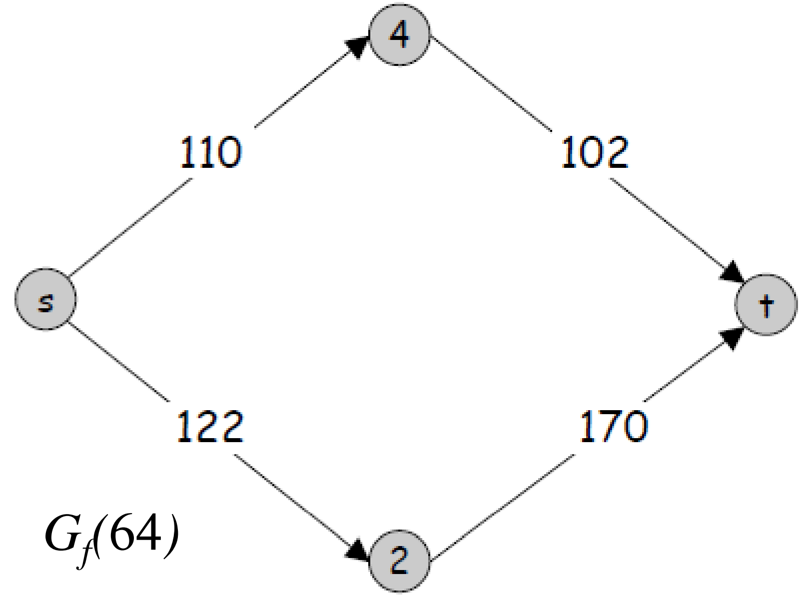
wordt case: the algorithm might take a number of augmentations
exponential in the input size !

avoiding piecemeal augmenting

maxflow = 225



algorithm might take 225 steps



intuition: we would like large residual capacities on augmenting paths

- maintain a scaling parameter Δ
- initially Δ is high, e.g. the largest power of 2 smaller than C
- search for augmenting path in reduced residual graph $G_f(\Delta)$ with only the arcs with capacity larger than Δ
- repeat with $\Delta/2$ until $\Delta=1$

the scaling max-flow algorithm finds the maximum flow in $O(|E|\ \mathrm{ld}(C))$ steps

it can be implemented in $O(|E|^2\ \mathrm{ld}(C))$ time

polynomial number of augmentations

- lemma 1: the scaling is done $\lfloor ld(C) \rfloor$ time

  proof: initially $C/2 < \Delta \leq C$. C decreases with a factor 2 until $C=1$

- lemma 2: let f be the flow after the phase with scaling Δ;
  then f^* is at most $|f|$ + |E| Δ .

  proof: almost identical to proving the max-flow min-cut theorem

- lemma 3: there are at most 2|E| augmentations per phase

  proof: let f be the flow at the end of the previous scaling phase:
  according to lemma 2 we have $|f^*| \leq |f|$ + |E|2Δ
  and each augmentation in a Δ phase increases $|f|$
  by at least Δ.

the scaling max-flow algorithm finds the maximum flow in O(|E| ld(C)) steps

it can be implemented in O(|E|^2 ld(C)) time

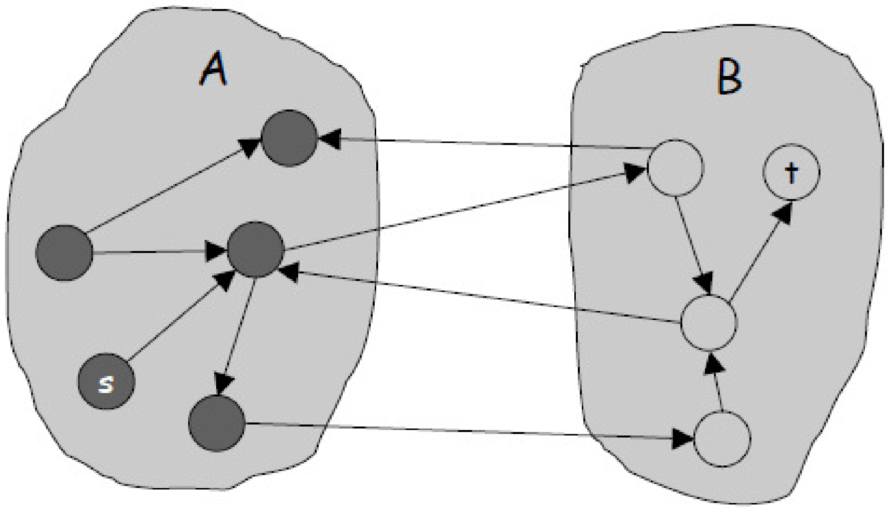
proof of lemma 2

Lemma 2. Let f be the flow at the end of a Δ-scaling phase. Then value of the maximum flow is at most $|f| + |E|\,\Delta$

Pf. (almost identical to proof of max-flow min-cut theorem)

- We show that at the end of a Δ-phase, there exists a cut (A, B) such that cap(A, B) $\leq |f| + |E|\,\Delta$
- Choose A to be the set of nodes reachable from s in $G_f(\Delta)$.
- By definition of A, $s \in A$.
- By definition of f, $t \notin A$.

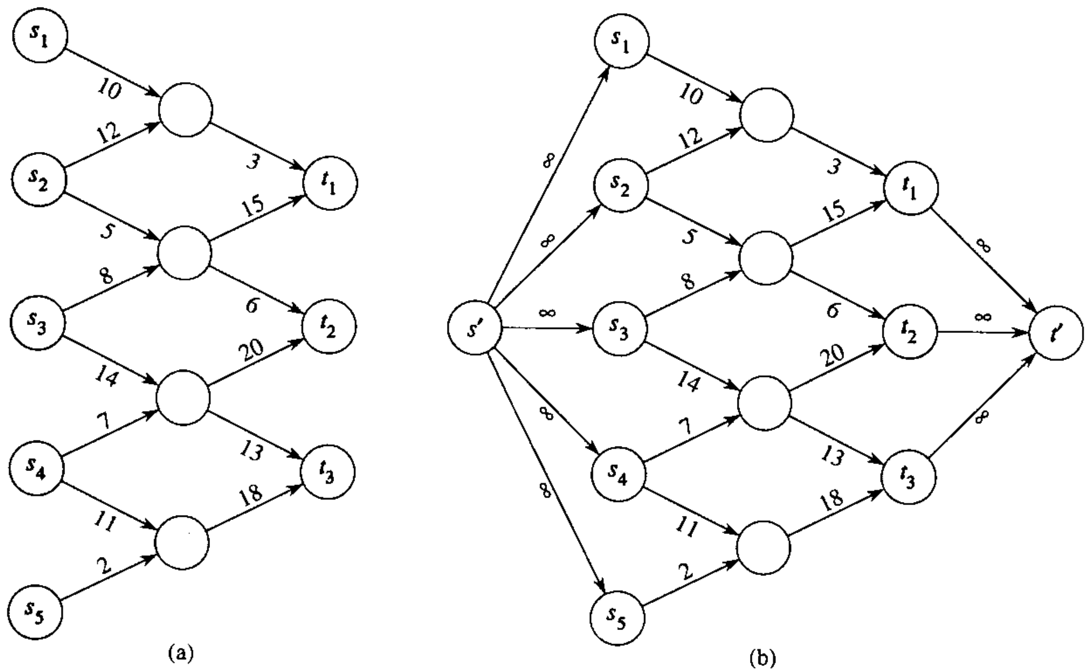
$$
\begin{aligned}
|f| &= \sum_{e \text{ out of } A} f(e) - \sum_{e \text{ in to } A} f(e) \\
&\geq \sum_{e \text{ out of } A} (c(e) - \Delta) - \sum_{e \text{ in to } A} \Delta \\
&= \sum_{e \text{ out of } A} c(e) - \sum_{e \text{ out of } A} \Delta - \sum_{e \text{ in to } A} \Delta \\
&\geq cap(A, B) - |E|\,\Delta \quad \blacksquare
\end{aligned}
$$



original network

multiple sources and sinks

- problems with multiple sources and sinks can be reduced to the single source/sink case



- a 'supersource 'with ∞ outgoing capacities to the multiple sources is added

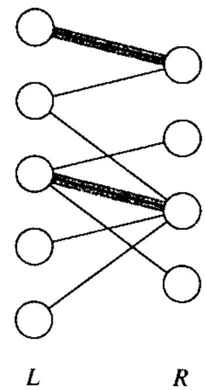
- a supersink with ∞ incoming capacities from the multiple sinks is added

network flow

- modelled after traffic in a network
- however, the power is
  in efficient solutions to combinatorial problems
  - bipartite matching
  - edge-disjoint paths
  - vertex-disjoint paths
  - scheduling
  - image segmentation
  - weighted bipartite matching
  - assignment problems
- many extensions that increase that power
  - circulations (with and without lower bounds)
  - multicommodity problems (chapter 11)
- several “easier” proofs in graph theory
  - theorem of Hall
  - theorem of Menger

bipartite graph matching

- a bipartite graph is an undirected graph $G = (V, E)$ in which the vertices V can be partitioned into two subsets L and R such that all edges are between the two sub sets.

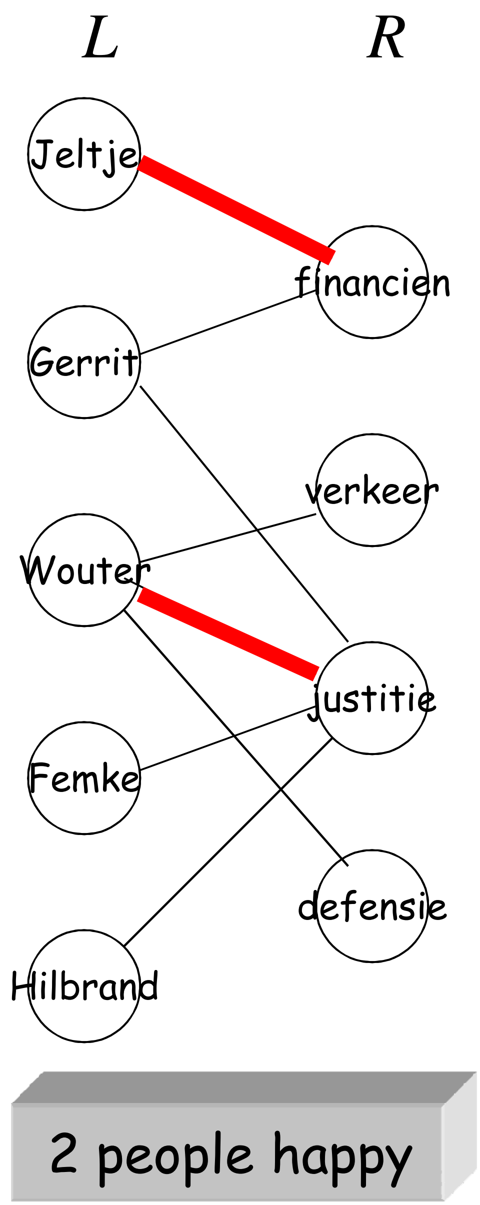


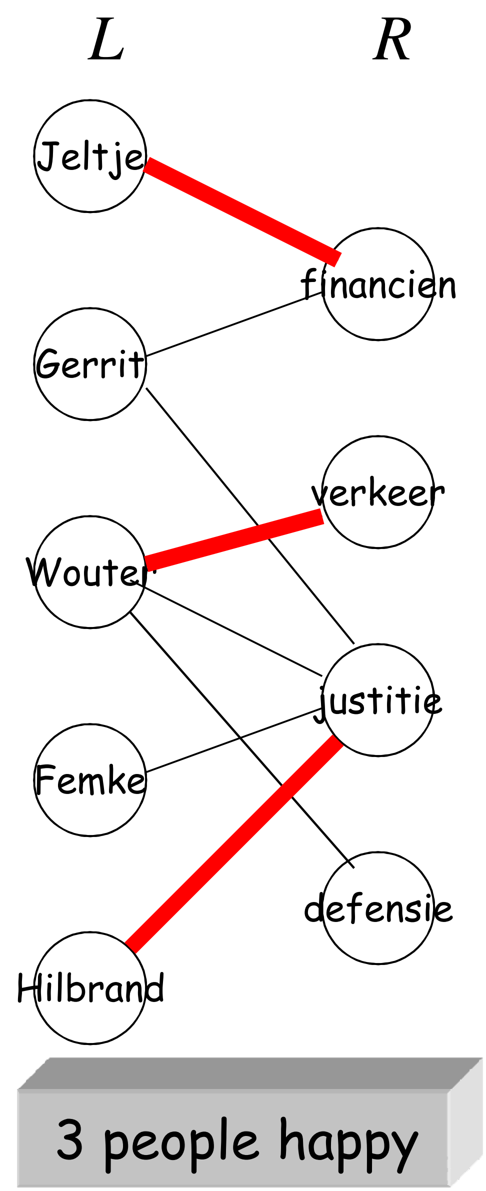
(a)

- a matching M of a graph $G = (V, E)$ is a subset of E such that for each vertex $v \in V$ at most one edge in M is incident.
- a maximum matching is a matching M that has maximum cardinality $|M|$

many practical problems can be converted into a **maximum bipartite matching** problem.

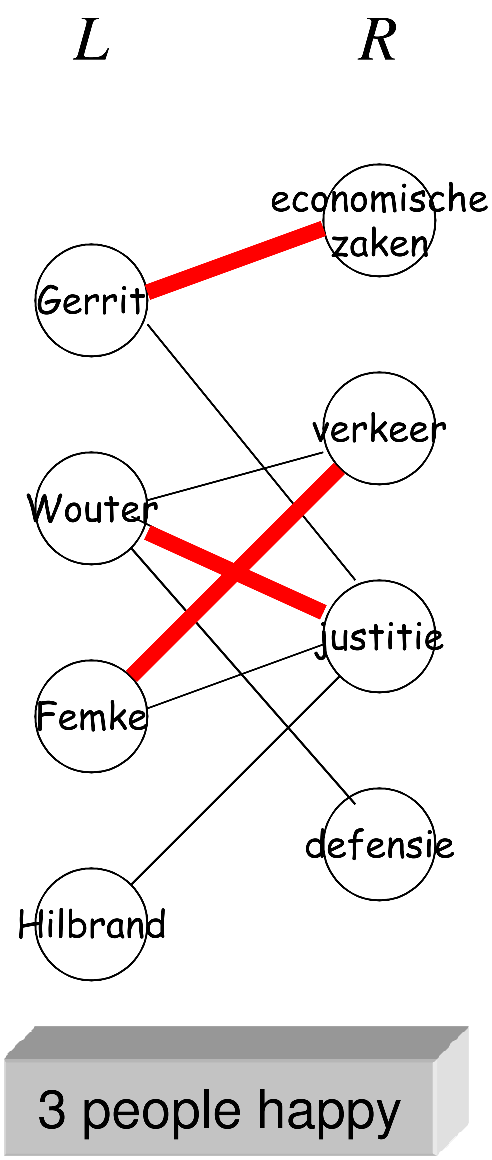
matching people to tasks

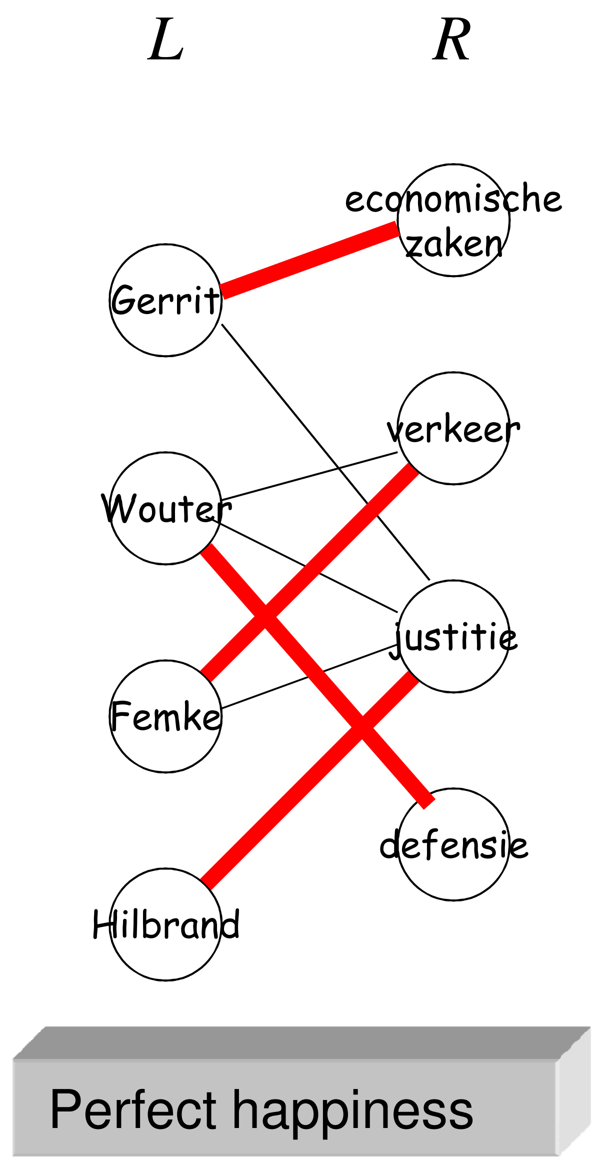




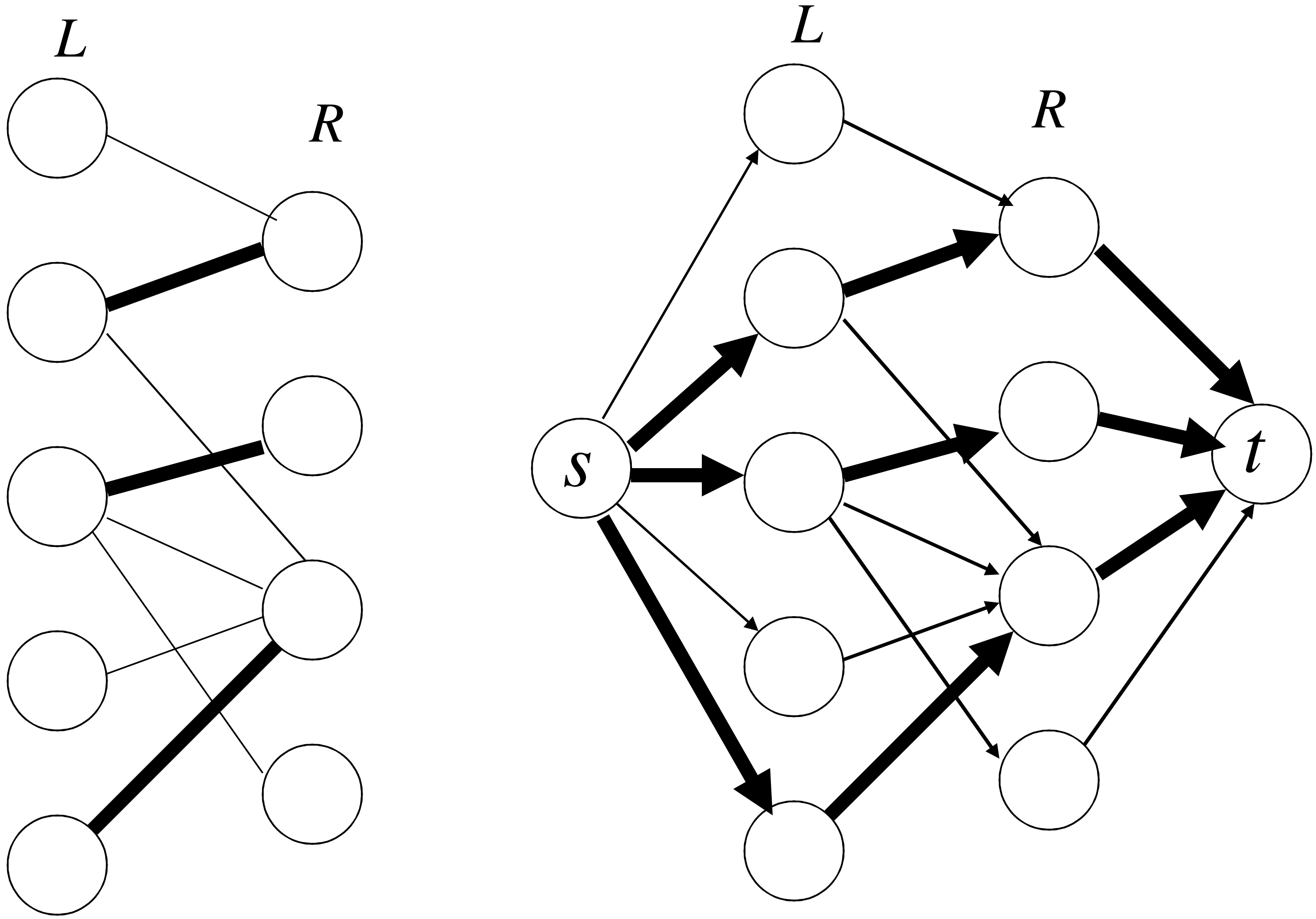
perfect matchings

- a perfect matching is a matching where every vertex is matched.

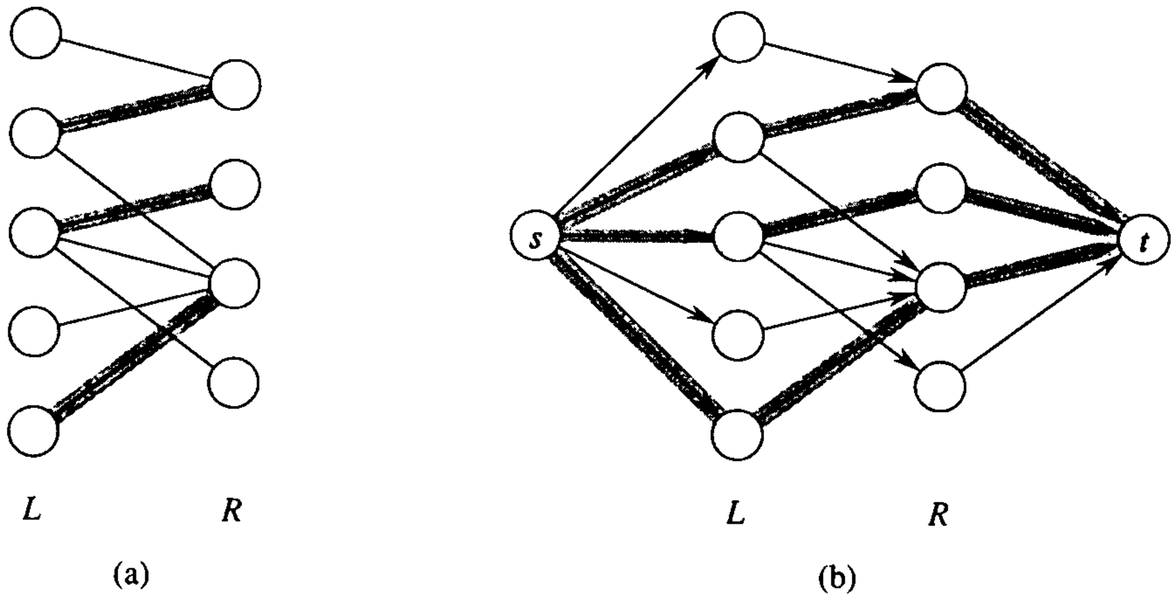




convert into a maximum flow problem



conversion to "maximum flow"



- have a single source s connected to each vertex inL and a single sink connected to each vertex in R
- assign a unit capacity to every edge in E′

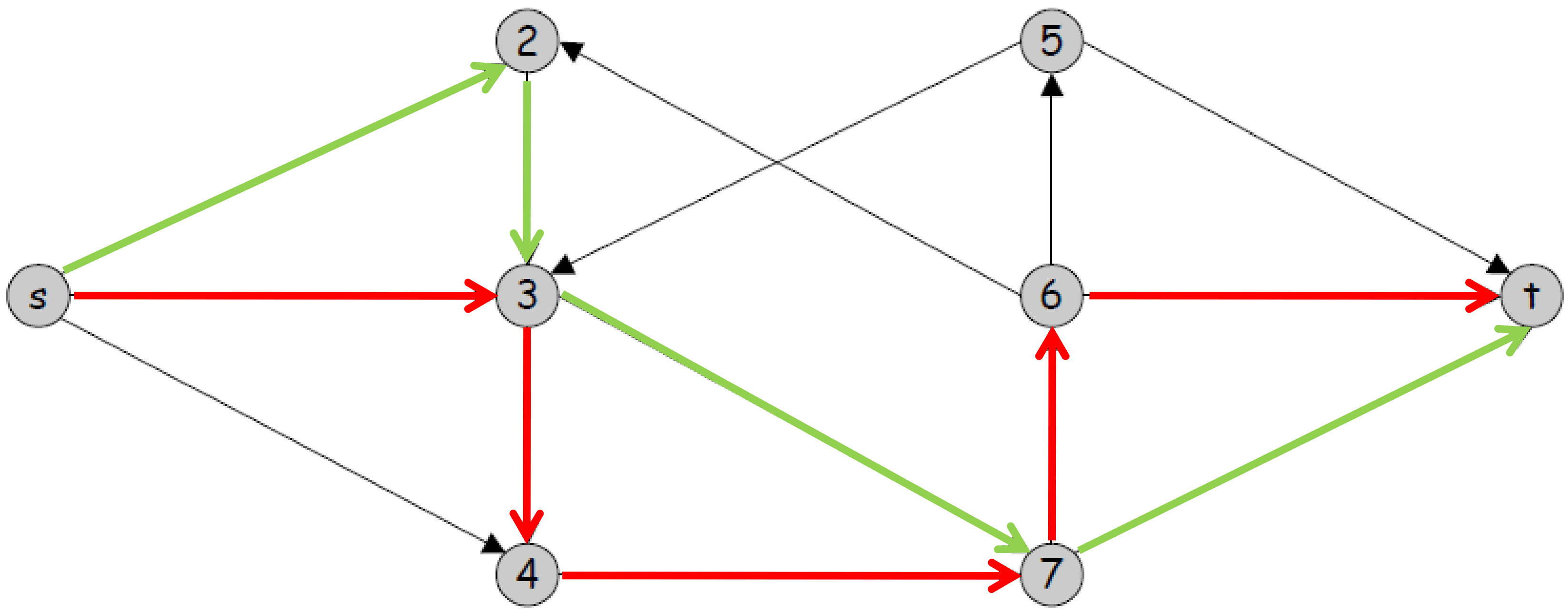
$$
\begin{aligned}
E' &= \{(s,u) : u \in L\} \\
&\cup \{(u,v) : u \in L,\ v \in R,\ \text{and } (u,v) \in E\} \\
&\cup \{(v,t) : v \in R\}\ .
\end{aligned}
$$

- flows are constrained to integer values
- the solution to the maximum flow problem gives us a solution to the maximum bipartite matching problem

edge-disjoint paths

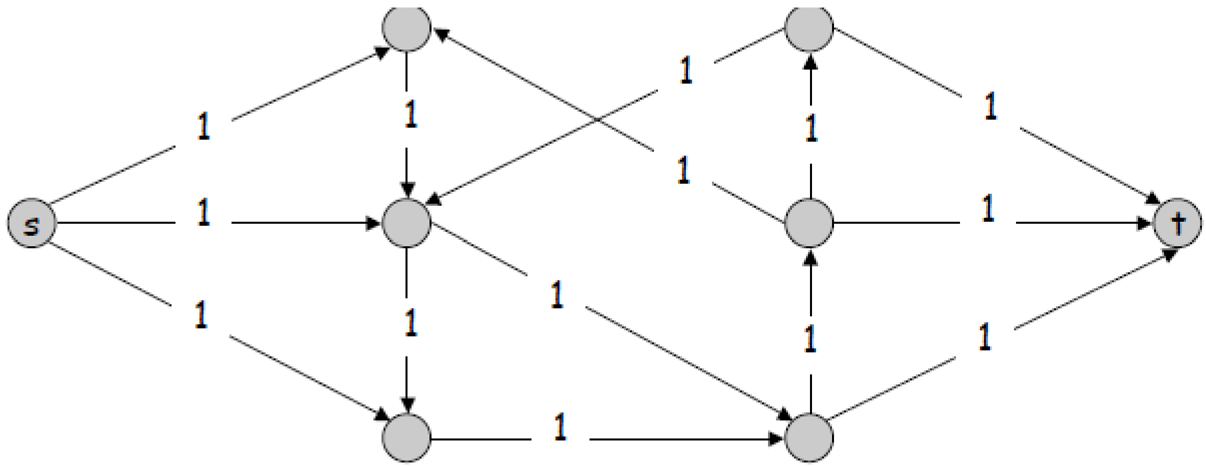
given a digraph $G=(V,E)$ and two nodes s and t ,
what is the maximum number of edge-disjoint s-t paths

two paths are edge-disjoint
if they have no edge in common



application: communication networks
where links can only be used for one packet

network flow formulation



- number of paths implies the existence of that flow value
  - suppose there are k paths
  - set $f(e) = 1$ if e is in some path, else set $f(e) = 0$
  - since the paths are edge-disjoint, f is a flow with $|f|$=k
- flow f implies $|f|$ edge-disjoint paths
  - integrality theorem implies a 0-1 flow with value $|f|$
  - consider an edge (s,u) with $f(s,u)=1$
    - by conservation there exists an edge (u,v) with $f(u,v)=1$
    - continue until t is reached, always choosing a new edge
    - remove cycles if present
  - since t is $|f|$ times reach, there are $|f|$ paths

network connectivity

given a digraph $G=(V,E)$ and two nodes and t ,
what is the minimum number of edges
whose removal disconnects s and t

a set $F \subseteq E$ disconnects s from t
if it takes at least one edge
from every s-t path

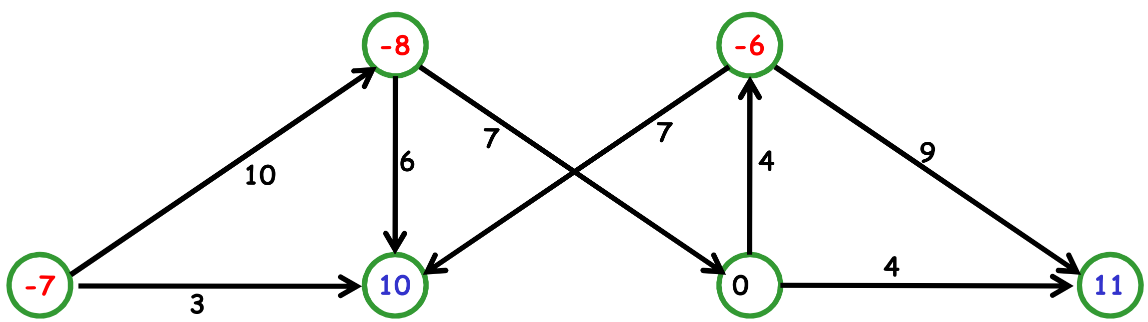
theorem of Menger:
the maximum number of edge-disjoint s-t paths
is equal to the minimum number of edges that disconnects s from t

- suppose $F \subseteq E$ disconnect s from t and $|F|=k$
  - every s-t path uses at least one edge in F
  - the paths required in the edge-didjoint path problem are disjoint
  - so, there cannot be more than k disjoint paths
- suppose maximum number of edge-disjoint paths is k
  - so the maximum flow is k
  - by the max-flow min-cut theorem there is s-t cut of capacity k
  - let F be the set of edges that go from A to B
  - F disconnects s from t

circulations

- directed graph $G=(V,E)$
- arc capacities $c(e)$, $e \in E$
- node supplies and demands $d(v)$, $v \in V$
  - when $d(v) > 0$, then v is a demand node
  - when $d(v) < 0$, then v is a supply node
  - when $d(v) = 0$, then v is a transshipment node
- a circulation is a function f such that

  - for each $e \in E$: $0 \leq f(e) \leq c(e)$
  - for each $v \in V$: $\sum_{e \text{ into } v} f(e) - \sum_{e \text{ out of } v} f(e) = d(v)$



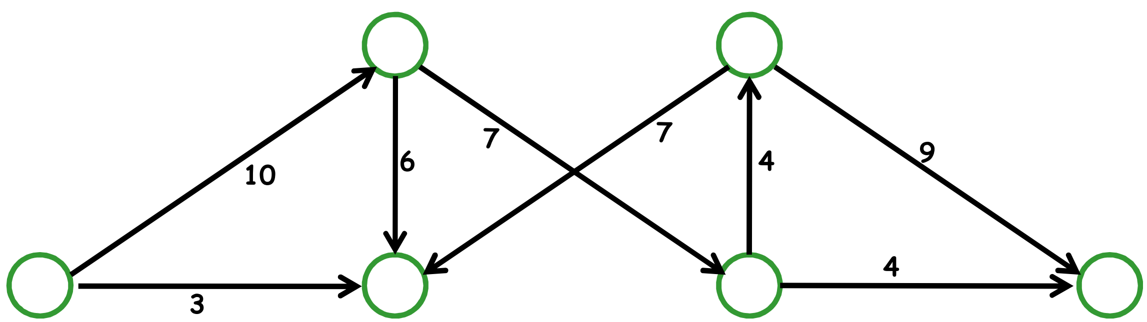
circulation problem: does such a function exist, given (V,E,c,d)

circulation problems

- a necessary condition for the existence:

$$\sum_{v=\text{demand}} d(v) = \sum_{v=\text{supply}} d(v) =: D$$

  - follows immediately from summing all conservation constraints
- network flow formulation:

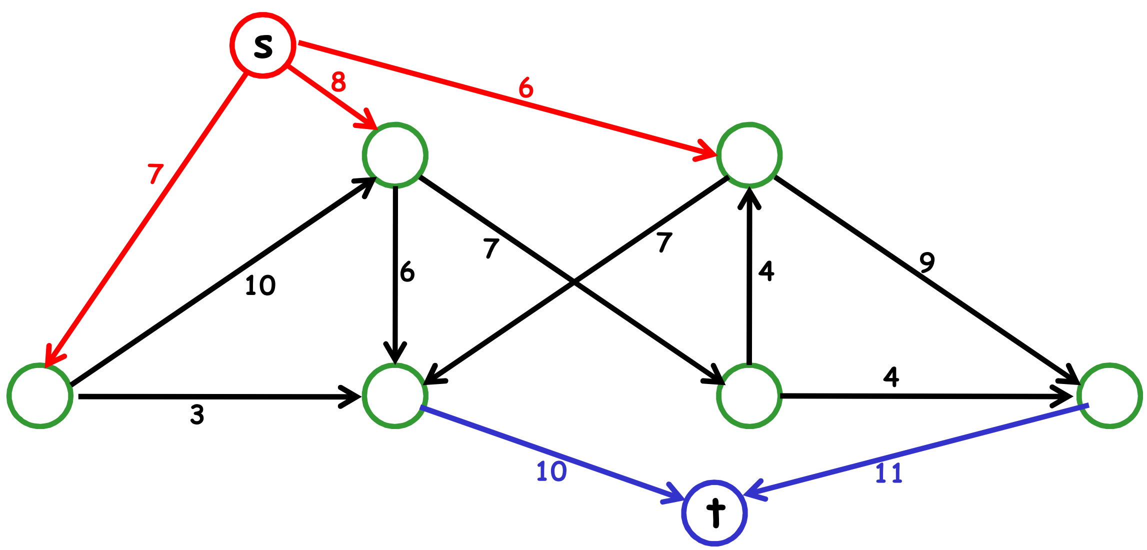


circulation problems

- a necessary condition for the existence:

$$\sum_{v=\text{demand}} d(v) = \sum_{v=\text{supply}} d(v) =: D$$

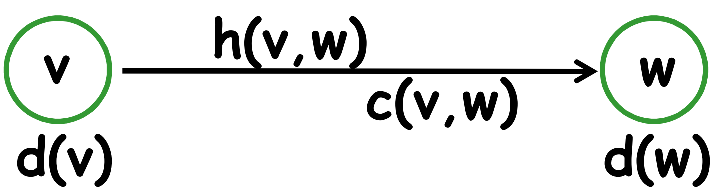
  - follows immediately from summing all conservation constraints
- network flow formulation:
  - add a source s and a sink t
  - for each supply node v, add an arc from s to v with $c(s,v)=-d(v)$
  - for each demand node v, add an arc from v to t with $c(v,t)=d(v)$
- the original circulation problem has a solution fif its network flow problem has a maximum flow value D

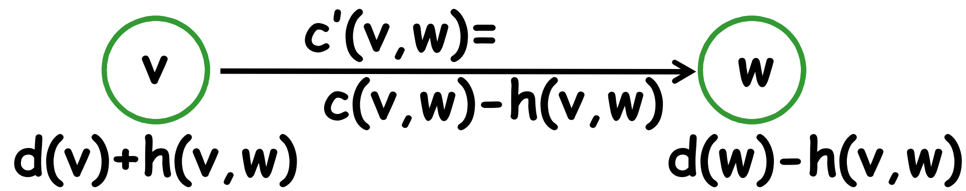


circulation problems with lower bounds

- directed graph $G=(V,E)$
- arc capacities $c(e)$ and lower bounds $h(e)$, $e \in E$
- node supplies and demands $d(v)$, $v \in V$
- a circulation bounded below is function f such that
  - for each $e \in E$: $h(e) \leq f(e) \leq c(e)$
  - for each $v \in V$: $\sum_{e \text{ into } v} f(e) - \sum_{e \text{ out of } v} f(e) = d(v)$

circulation problem with lower bounds:
does such a function f exist, given (V,E,c,d,h)





theorem: there exists a circulation bounded below in G fif
there exists a circulation in the modified graph.
When all demands, capacities and lower bounds are integers,
then there exists a circulation in G with integer flow values f