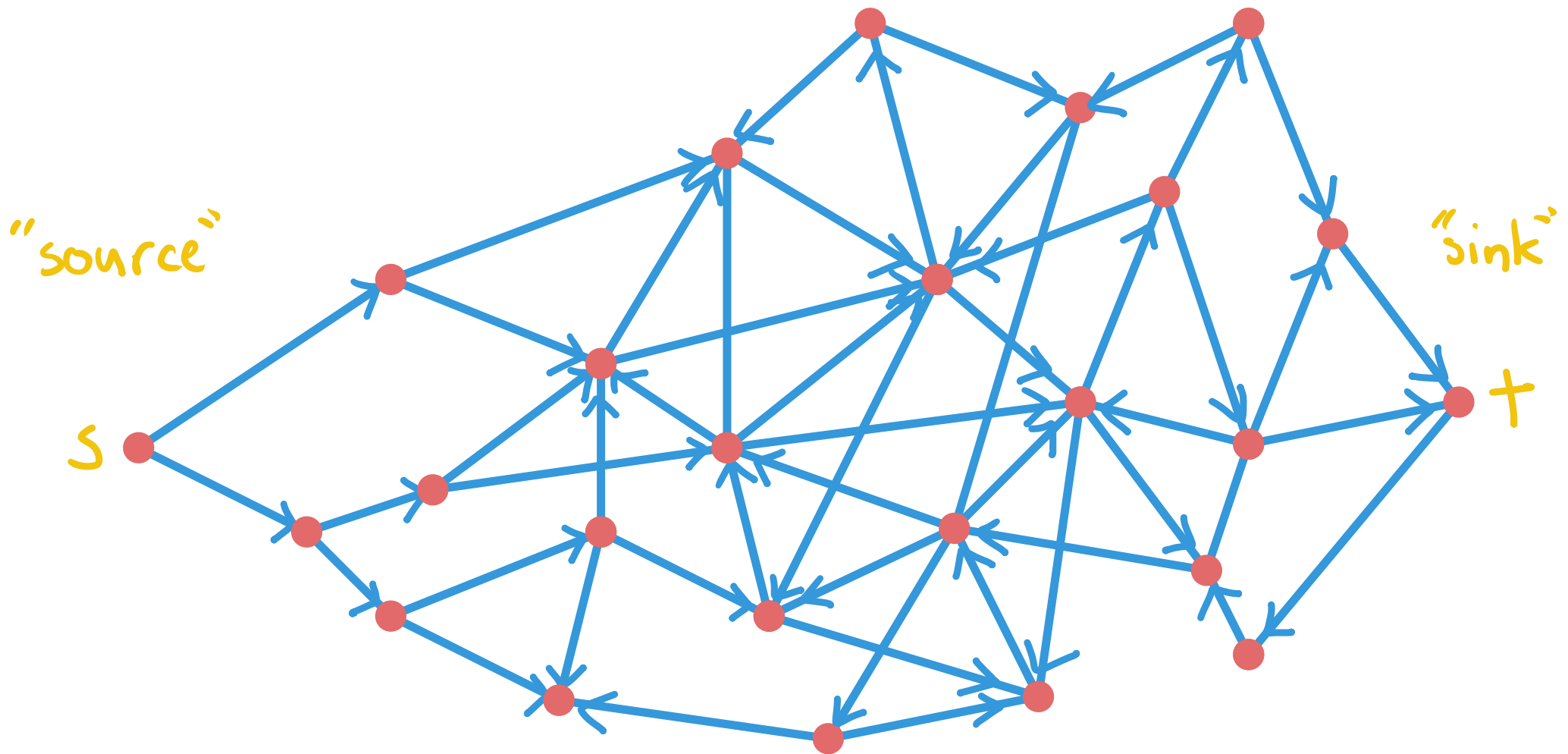


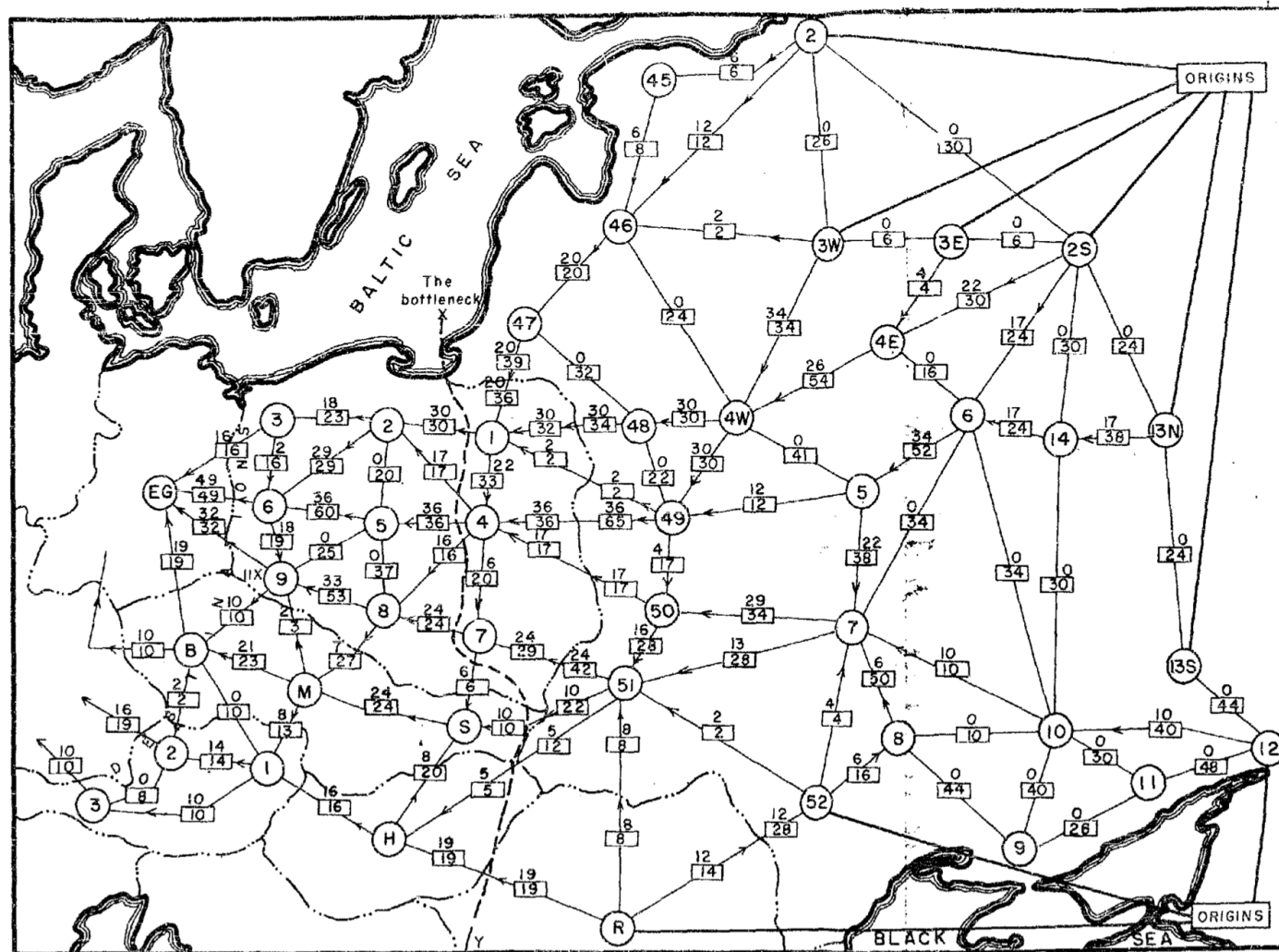
Network Flow

Measuring connectivity



how well connected are s and t?

Physical infrastructure



SECRET
RM-1373
10-24-55
-33-

Fig. 7 — Traffic pattern: entire network available

Legend:

— International boundary

(B) Railway operating division

← 9/12 → Capacity: 12 each way per day. Required flow of 9 per day toward destinations (in direction of arrow) with equivalent number of returning trains in opposite direction

All capacities in $\sqrt{1000}$'s of tons } each way per day

Origins: Divisions 2, 3W, 3E, 2S, 13N, 13S, 12, 52 (USSR), and Roumania

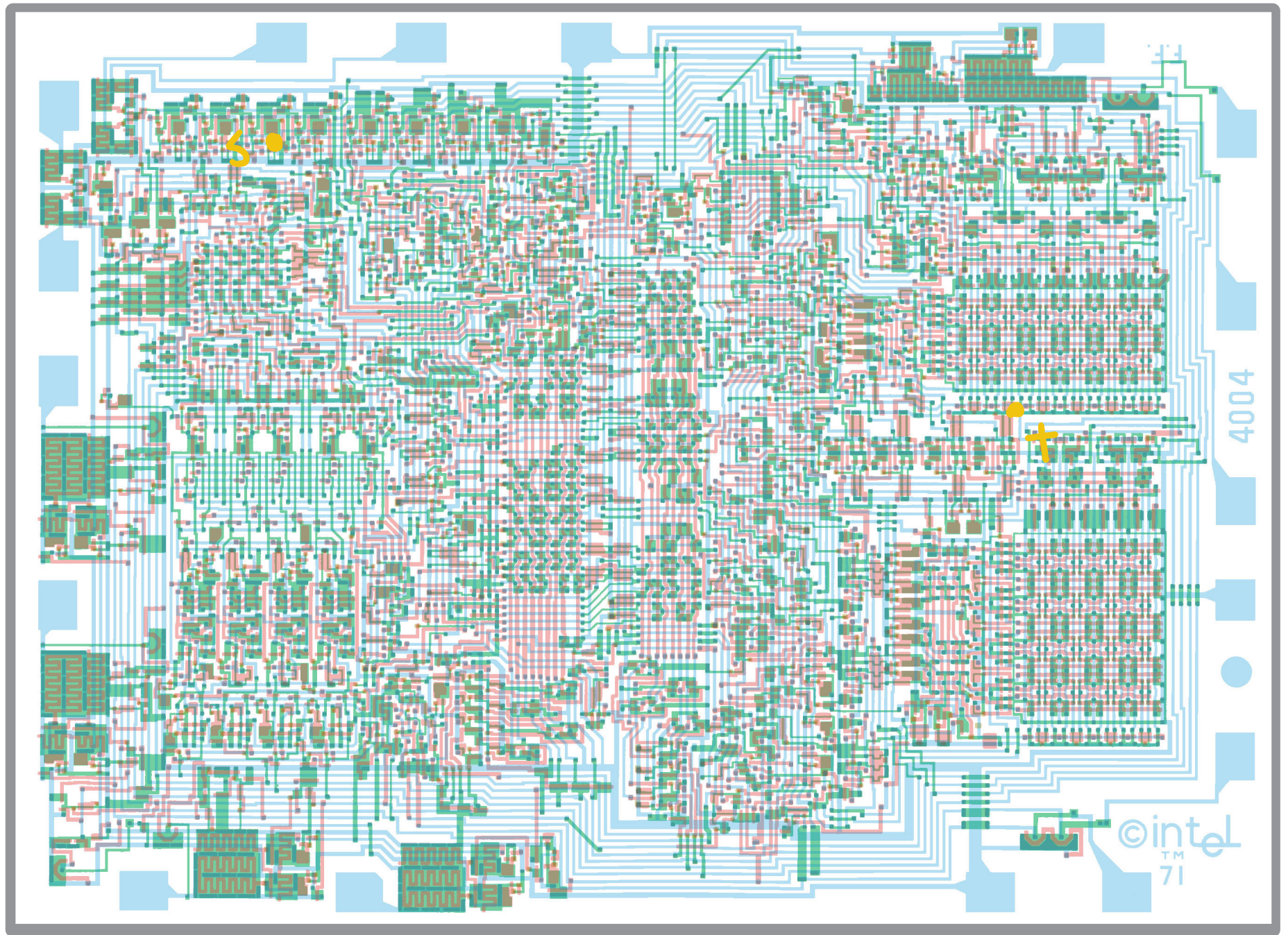
Destinations: Divisions 3, 6, 9 (Poland); B (Czechoslovakia); and 2, 3 (Austria)

Alternative destinations: Germany or East Germany

Note IIX of Division 9, Poland

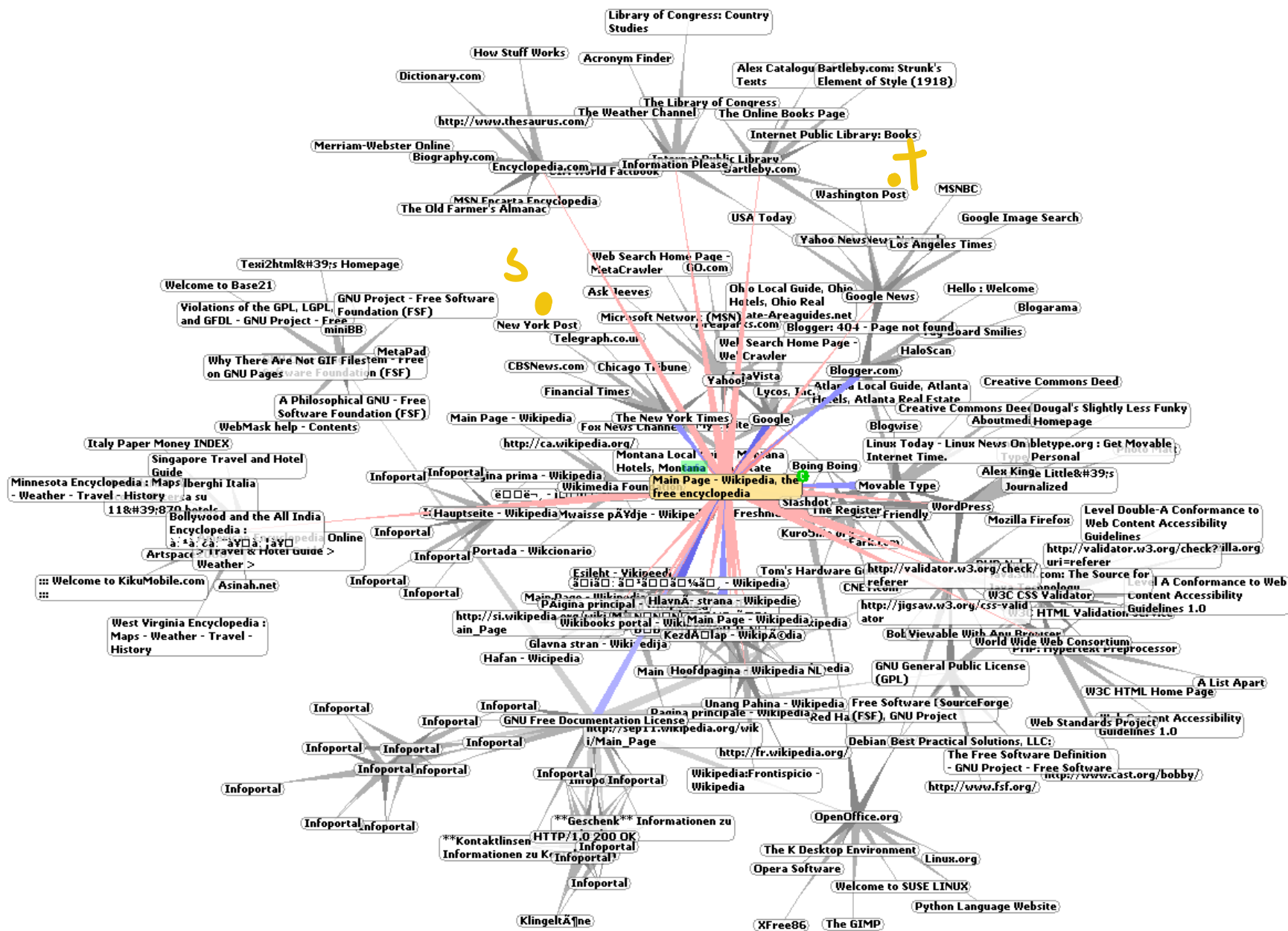
(from Jeff's notes)

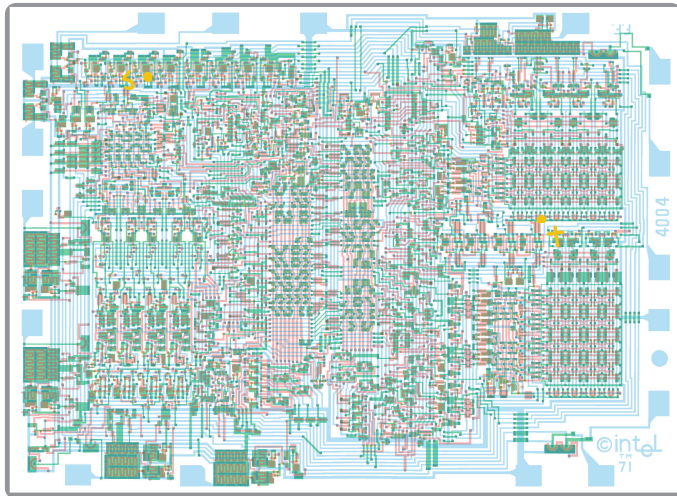
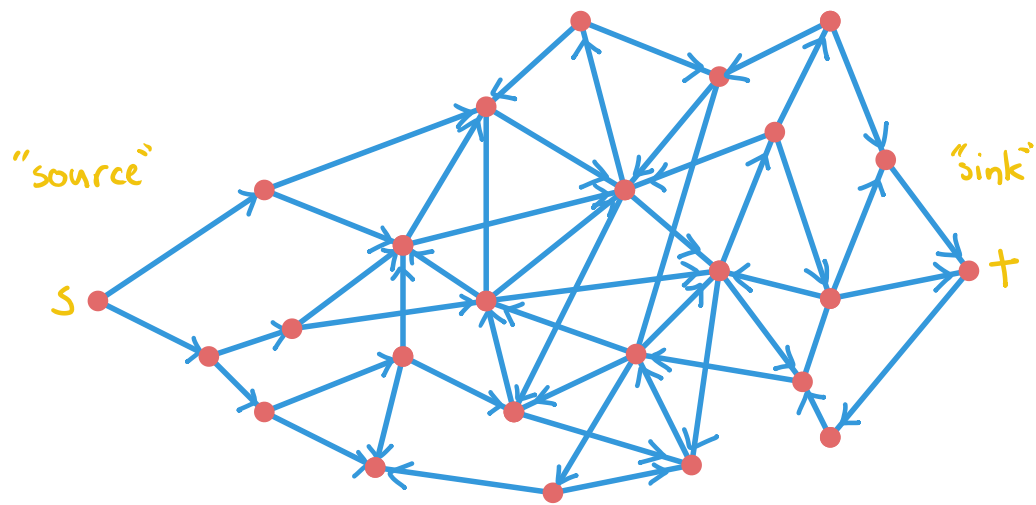
CPU's



social network

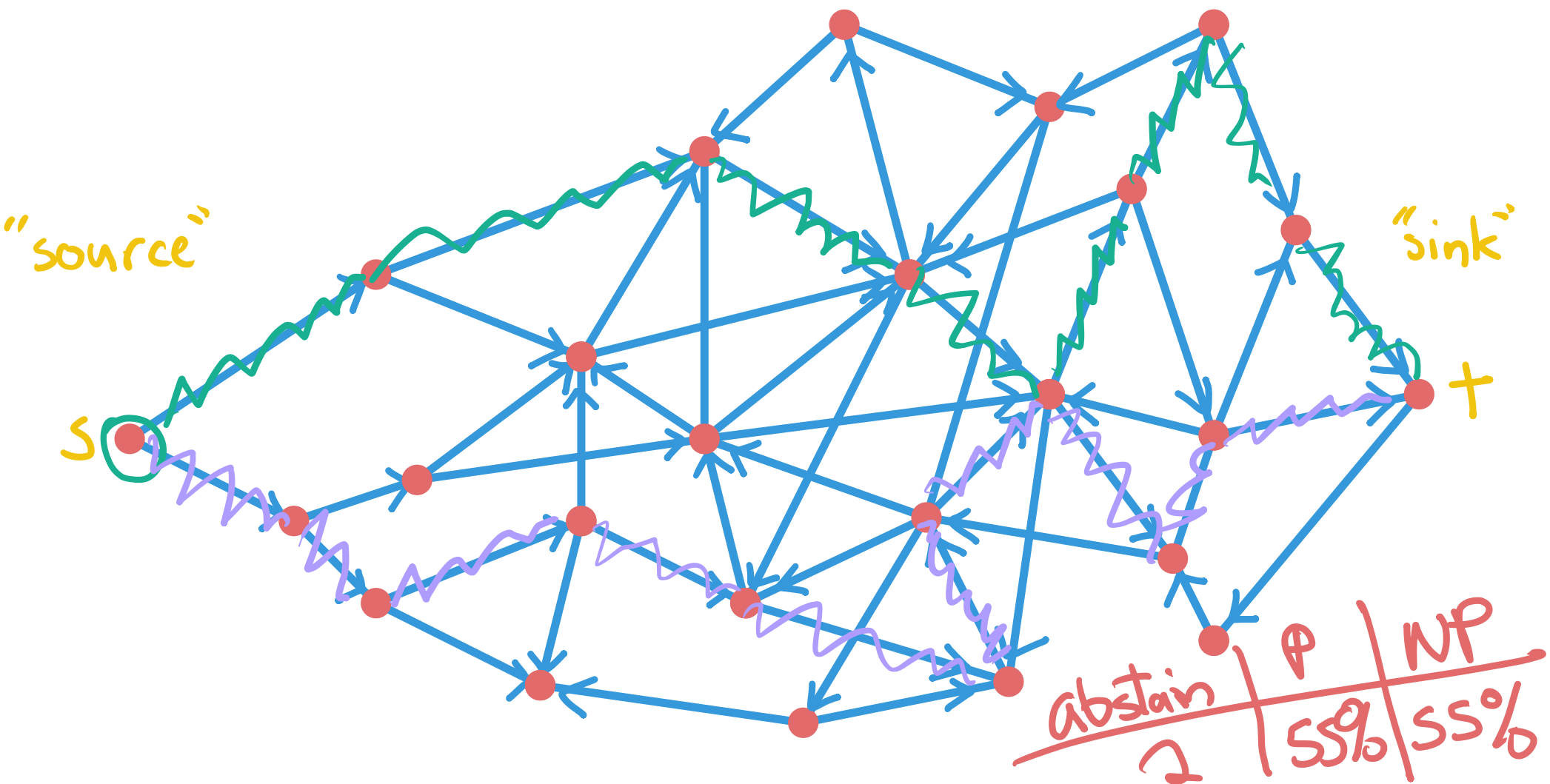






- ① Is there an $s \rightsquigarrow t$ path?
- ② How many distinct $s \rightsquigarrow t$ paths?
- ③ how many k -edge $s \rightsquigarrow t$ walks?
(k fixed)
- ④ length of shortest $s \rightsquigarrow t$ path?
- ⑤ how many shortest $s \rightsquigarrow t$ paths?
- ⑥ In a random walk starting from s , what is the expected # of steps until we reach t ?
- ⑦ max # of edge-disjoint $s \rightsquigarrow t$ paths?
- ⑧ min # of edge deletions to disconnect $s \rightsquigarrow t$?

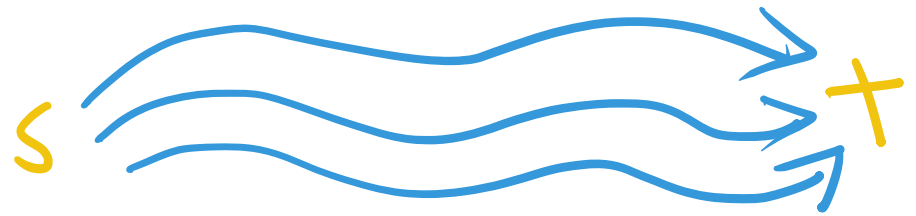
Measuring connectivity



⑦ max # of edge-disjoint $s \leadsto t$ paths?

⑦ max # of edge-disjoint $s \rightsquigarrow t$ paths?

"path packing"



collection of edge-disjoint paths

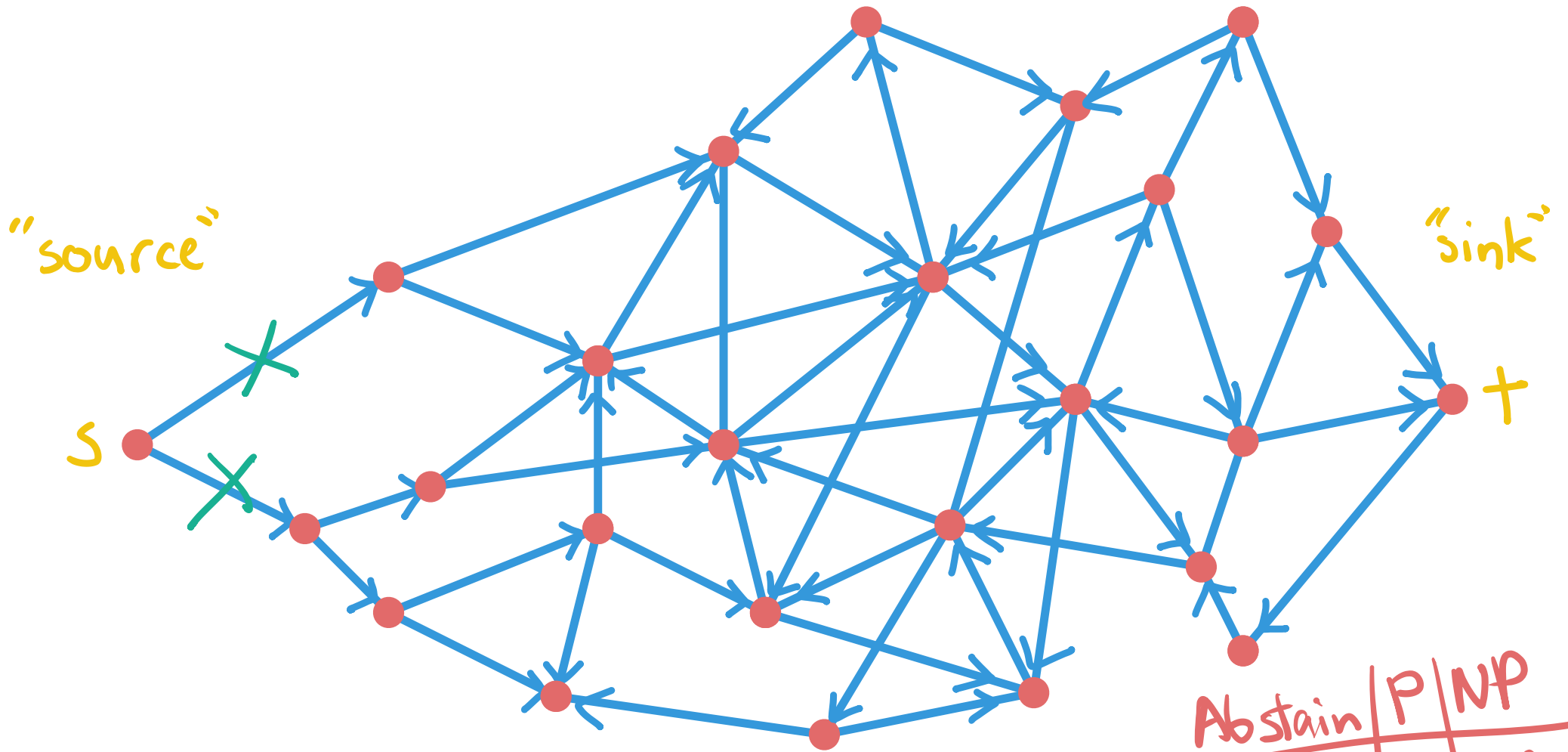
"packing paths into edges"

Question ⑦ = max cardinality path packing

1st example of "packing problem"

path p "packs into" each edge $e \in p$

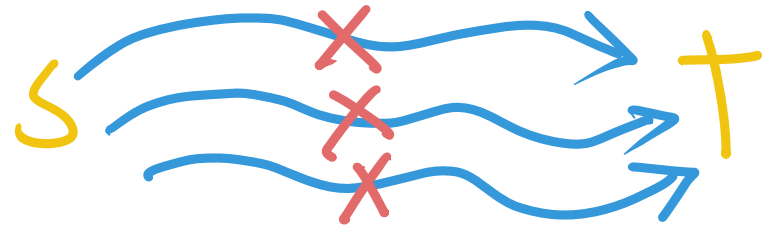
Measuring connectivity



Abstain	P	NP
2	45%	55%

⑧ min # of edge deletions to disconnect $s \rightsquigarrow t$?

⑧ min # of edge deletions to
disconnect $s \rightarrow t$?



" $s \rightarrow t$ cut"

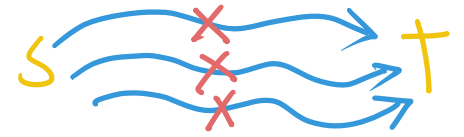
set of edges whose removal
disconnects t from s

Question ⑧ = min s - t cut

⑧ min # of edge deletions to disconnect $s \rightsquigarrow t$?

Question ⑧ = min s - t cut

" $s \rightarrow t$ cut"



set of edges whose removal disconnects t from s

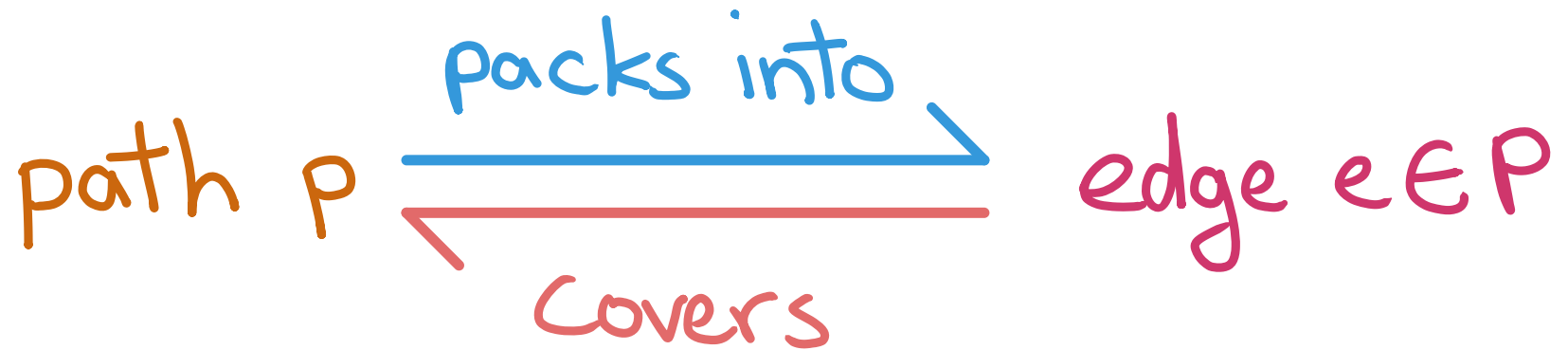
example of "covering problem"

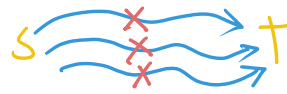
edge e "covers" path p containing e

goal:

min edge set covering all $s \rightsquigarrow t$ paths

Duality





Duality
 path p $\xrightarrow{\text{packs into}}$ edge $e \in P$
~~covers~~

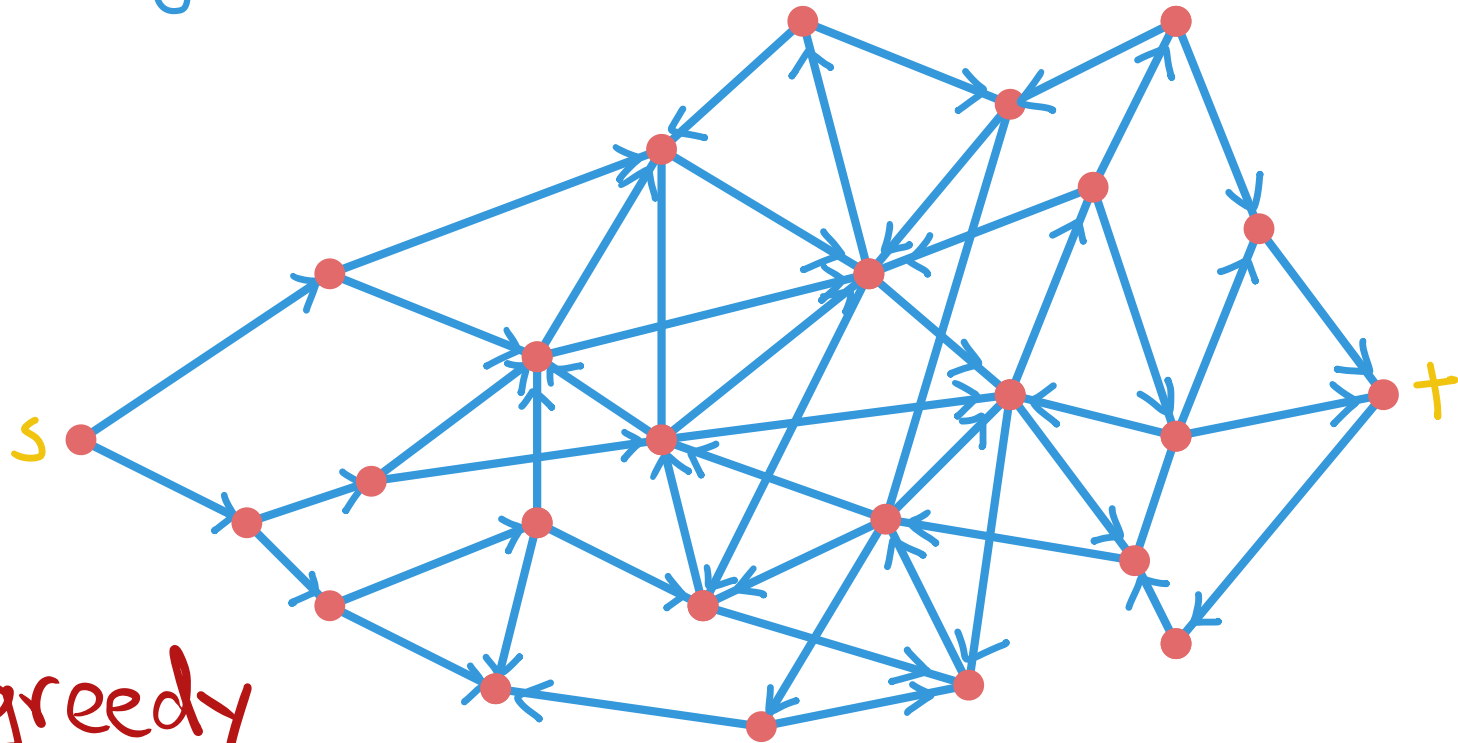
Lemma

$$\left(\begin{array}{c} \text{max \#} \\ \text{disjoint} \\ s \rightsquigarrow t \text{ paths} \end{array} \right) \leq \left(\begin{array}{c} \text{min \# edges} \\ \text{in any } s \rightsquigarrow t \text{ cut} \end{array} \right)$$

* 2nd example of
 packing $\leq$ covering

How to compute

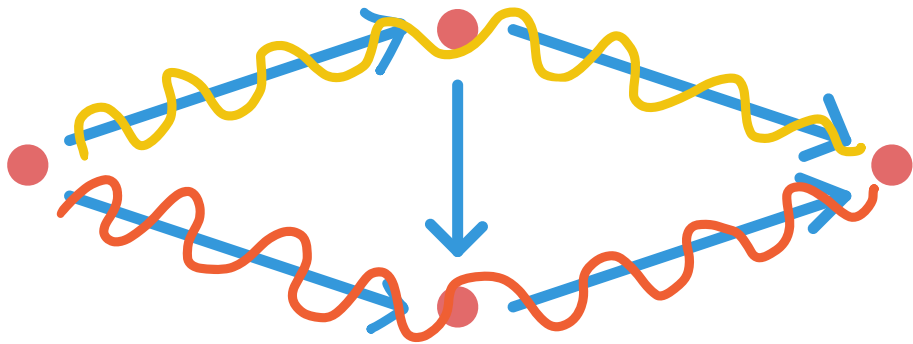
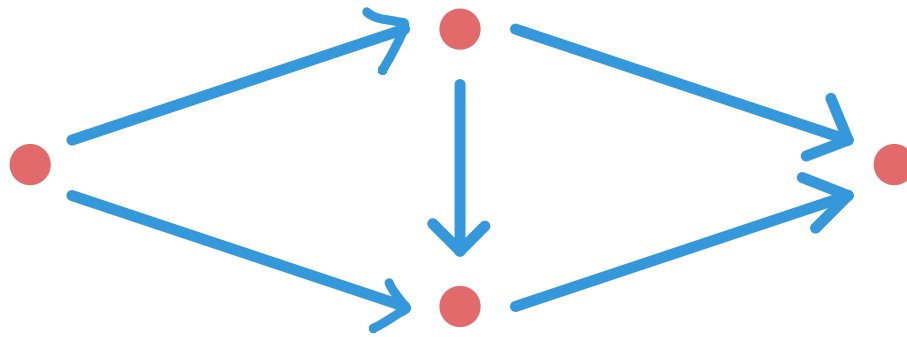
⑦ max # of edge-disjoint $s \rightsquigarrow t$ paths?



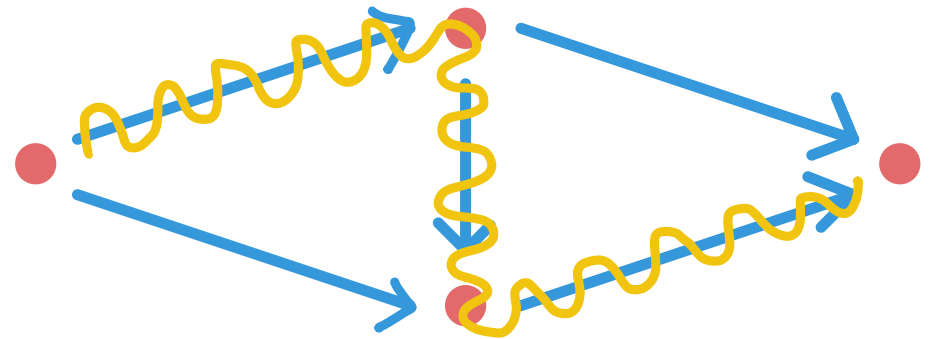
Iterative/greedy

① while there is a path $p: s \rightsquigarrow t$

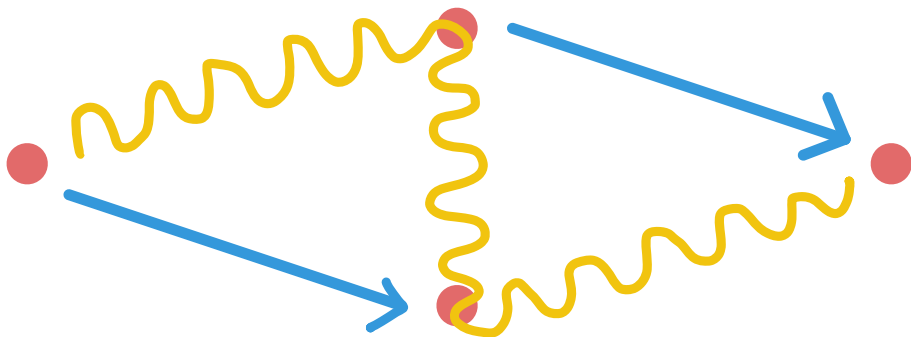
② take p and remove all its edges from the graph



(good)

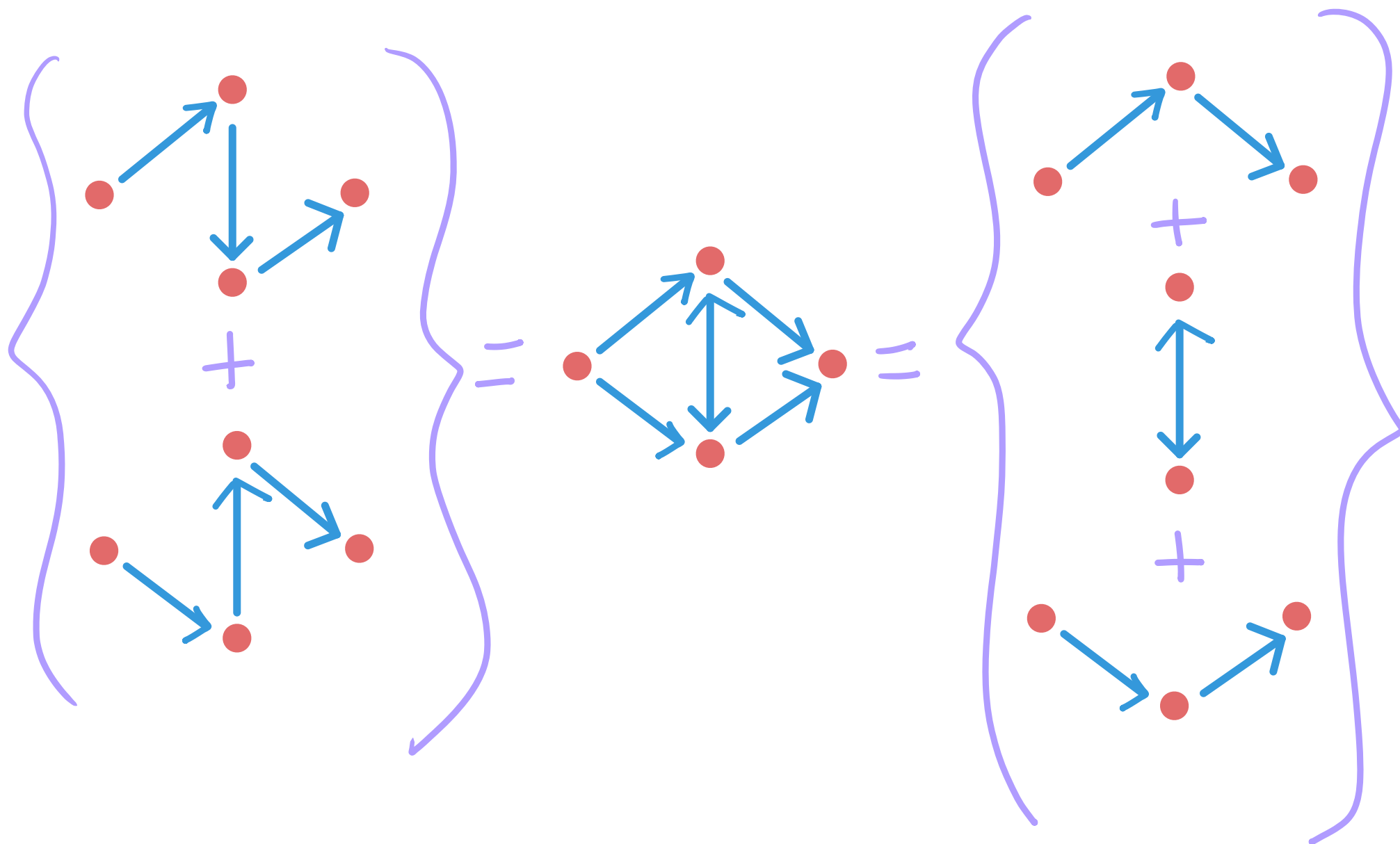


(bad)

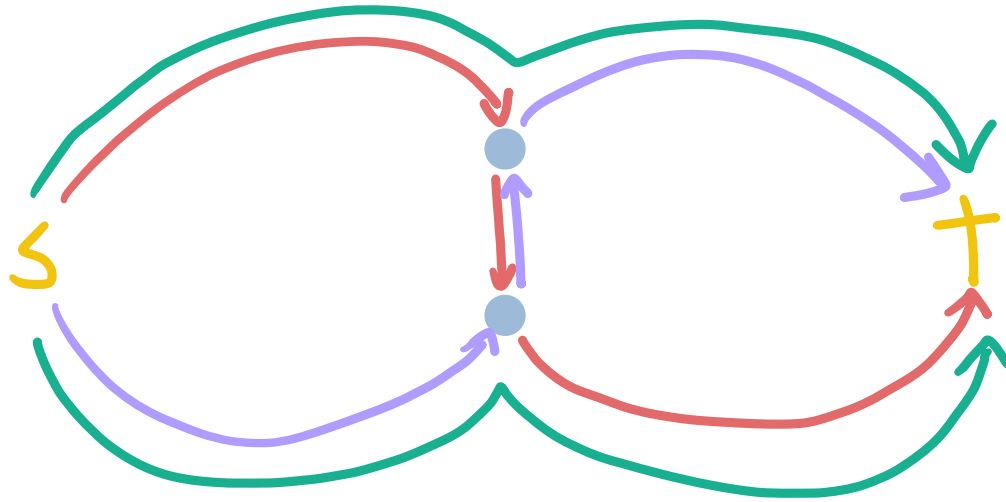


- ① while there is a path $p: s \rightsquigarrow t$
 - @ take p and remove all its edges from the graph

Consider:



Lemma let p, q be edge-disjoint $s \rightsquigarrow t$ paths



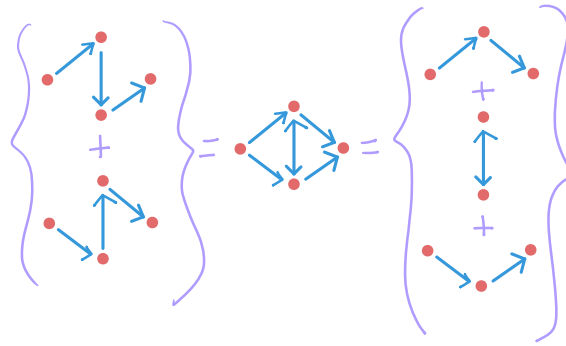
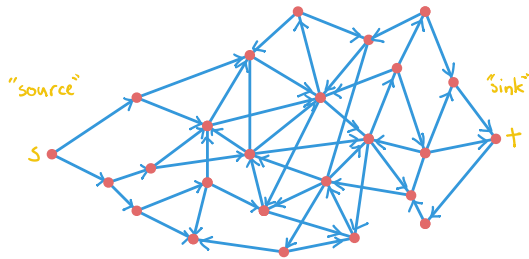
then there is p', q' : $s \rightsquigarrow t$ disjoint
without opposite pairs of edges

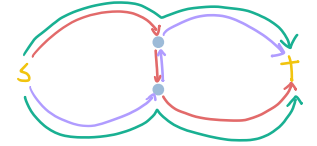
Proof: try it!

hint: by induction on # of opposite pairs

How to compute

⑦ max # of edge-disjoint $s \rightsquigarrow t$ paths?



Lemma let p, q be edge-disjoint $s \rightsquigarrow t$ paths

then there is $p', q' : s \rightsquigarrow t$ disjoint
without opposite pairs of edges

augmenting-paths($G = (V, E), s, t$)

/ Ford and Fulkerson [FF56]*

1. $P \leftarrow \emptyset$
2. While there is an (s, t) -path $p : s \rightsquigarrow t$:
 - A. Remove all of p 's edges from the graph.
 - B. Add the reverse of all of p 's edges to the graph.
 - C. Uncross p with previously selected paths, and add p to P .

"residual graph"



④ let

S = vertices reachable from s in residual graph

T = all other vertices (incl. t)

C = edges from S to T

⑤ each path decrease $|C|$ by 1.

⑥ at end, $|C| = 0$.

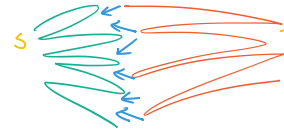
$$\Rightarrow \left(\begin{array}{c} \# \text{ paths} \\ \text{packed} \end{array} \right) = \left(\begin{array}{c} \# \text{ edges} \\ \text{initially in } C \end{array} \right)$$

Recall Lemma

$$\left(\begin{array}{c} \max \# \\ \text{disjoint} \\ s \rightsquigarrow t \text{ paths} \end{array} \right) \leq \left(\begin{array}{c} \min \# \\ \text{edges in any} \\ s \rightsquigarrow t \text{ cut} \end{array} \right)$$

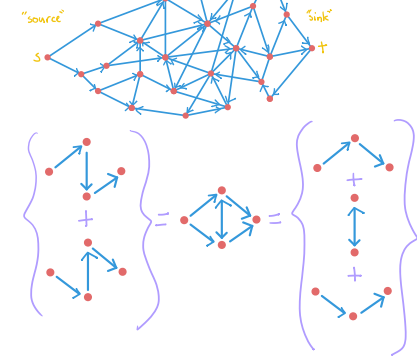
$$\left(\begin{array}{c} \# \text{ paths} \\ \text{packed} \end{array} \right) \leq \left(\begin{array}{c} \max \# \\ \text{disjoint} \\ s \rightsquigarrow t \text{ paths} \end{array} \right) \leq \left(\begin{array}{c} \min \# \\ \text{edges in any} \\ s \rightsquigarrow t \text{ cut} \end{array} \right) \leq \left(\begin{array}{c} \# \text{ edges} \\ \text{initially in } C \end{array} \right)$$

equal

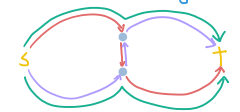


How to compute

⑦ max # of edge-disjoint $s \rightsquigarrow t$ paths?



Lemma let p, q be edge-disjoint $s \rightsquigarrow t$ paths



then there is p', q' $s \rightsquigarrow t$ disjoint without opposite pairs of edges

augmenting-paths($G = (V, E), s, t$)

/* Ford and Fulkerson [FF56]

1. $P \leftarrow \emptyset$

2. While there is an (s, t) -path $p : s \rightsquigarrow t$:

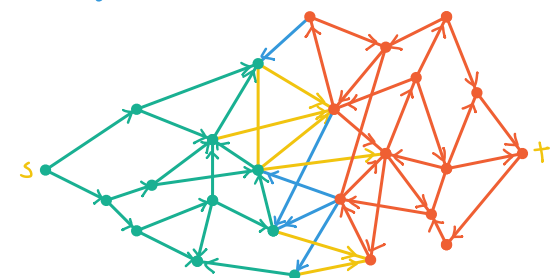
A. Remove all of p 's edges from the graph.

B. Add the reverse of all of p 's edges to the graph.

C. Uncross p with previously selected paths, and add p to P .

let (S, I) partition V st. $s \in S, t \in I$

C = edges from S to I (is on (s, t) -cut)



$$(\# \text{ paths packed}) \stackrel{?}{=} (\max \# \text{ disjoint } s \rightarrow t \text{ paths}) \stackrel{?}{=} (\min \# \text{ edges in any } s \rightarrow t \text{ cut}) \stackrel{?}{=} (\# \text{ edges initially in } C)$$

Theorems

$$\textcircled{1} \quad (\max \# \text{ disjoint } s \rightarrow t \text{ paths}) = (\min \# \text{ edges in any } s \rightarrow t \text{ cut})$$

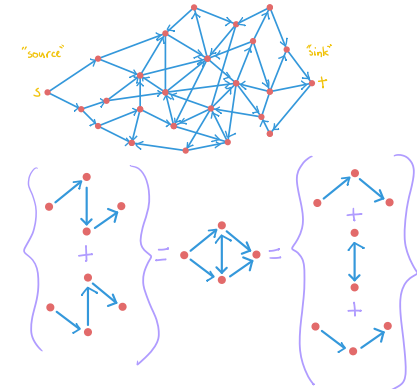
$$\textcircled{2} \quad (\# \text{ paths packed}) = (\max \# \text{ disjoint } s \rightarrow t \text{ paths})$$

$$\textcircled{3} \quad (\min \# \text{ edges in any } s \rightarrow t \text{ cut}) = (\# \text{ edges initially in } C)$$

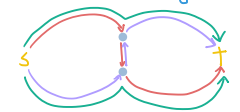
(where $(\# \text{ paths packed})$ and $(\# \text{ edges initially in } C)$ refers to the outcome of iterative finding paths in the residual graph)

How to compute

⑦ max # of edge-disjoint $s \rightarrow t$ paths?



Lemma let p, q be edge-disjoint $s \rightarrow t$ paths



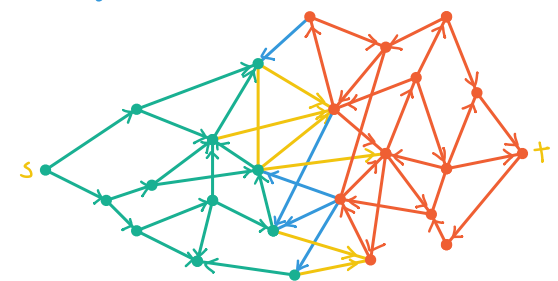
then there is p', q' : $s \rightarrow t$ disjoint without opposite pairs of edges

augmenting-paths($G = (V, E), s, t$)

/* Ford and Fulkerson [FF56]

1. $P \leftarrow \emptyset$
2. While there is an (s, t) -path $p: s \rightsquigarrow t$:
 - A. Remove all of p 's edges from the graph.
 - B. Add the reverse of all of p 's edges to the graph.
 - C. Uncross p with previously selected paths, and add p to P .

let (S, I) partition V st. $s \in S, t \in I$
 $C =$ edges from S to I (is an (s, t) -cut)



augmenting-paths($G = (V, E), s, t$)

/* Ford and Fulkerson [FF56]

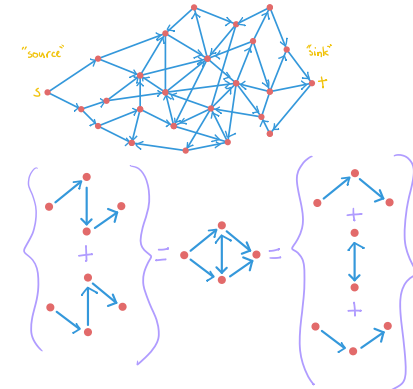
*/

1. $P \leftarrow \emptyset$
2. While there is an (s, t) -path $p : s \rightsquigarrow t$:
 - A. Remove all of p 's edges from the graph.
 - B. Add the reverse of all of p 's edges to the graph.
 - C. Uncross p with previously selected paths, and add p to P .³
3. $S \leftarrow \{\text{vertices still reachable from } s\}$.
4. Return the (s, t) -path packing P and the (s, t) -cut $\partial^+(S)$.

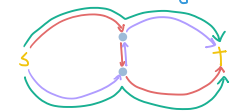
Theorem 14.5 ([FF56]). Let $G = (V, E)$ be a directed graph with m edges, and let $s, t \in V$ be distinct vertices. Let λ be the size of the max-flow/min-cut from s to t . Then augmenting-paths(G, s, t) produces both a maximum packing and a minimum cut in $O(m\lambda)$ time.

How to compute

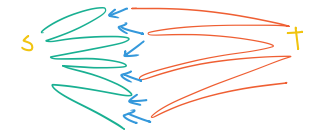
⑦ max # of edge-disjoint $s \rightsquigarrow t$ paths?



Lemma let p, q be edge-disjoint $s \rightsquigarrow t$ paths



then there is $p', q' : s \rightsquigarrow t$ disjoint without opposite pairs of edges



$$\textcircled{1} \left(\begin{array}{c} \text{max \#} \\ \text{disjoint} \\ s \rightsquigarrow t \text{ paths} \end{array} \right) = \left(\begin{array}{c} \text{min \#} \\ \text{edges in any} \\ s \rightsquigarrow t \text{ cut} \end{array} \right)$$

$$\textcircled{2} \left(\begin{array}{c} \# \text{ paths} \\ \text{packed} \end{array} \right) = \left(\begin{array}{c} \text{max \#} \\ \text{disjoint} \\ s \rightsquigarrow t \text{ paths} \end{array} \right)$$

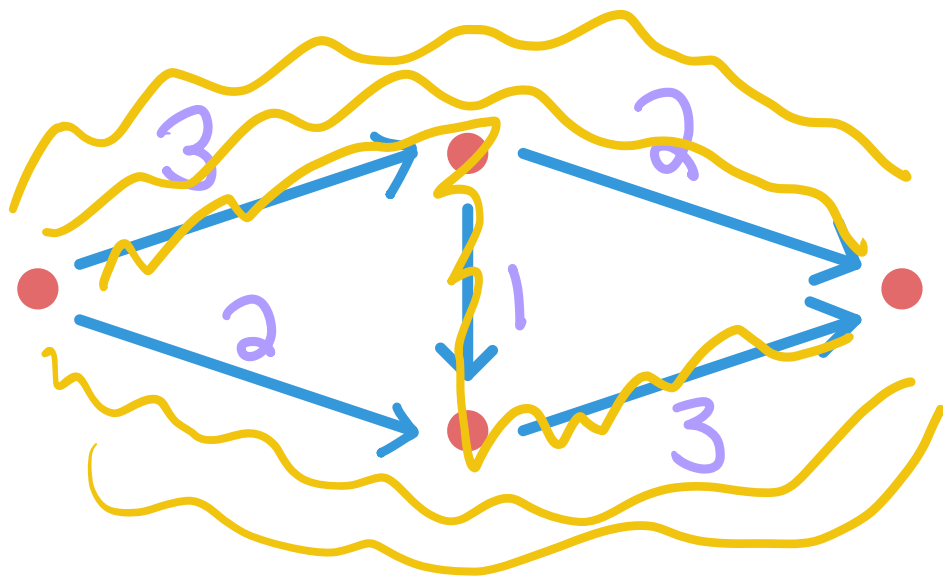
$$\textcircled{3} \left(\begin{array}{c} \text{min \#} \\ \text{edges in any} \\ s \rightsquigarrow t \text{ cut} \end{array} \right) = \left(\begin{array}{c} \# \text{ edges} \\ \text{initially in } C \end{array} \right)$$

1. $P \leftarrow \emptyset$
2. While there is an (s, t) -path $p : s \rightsquigarrow t$:
 - A. Remove all of p 's edges from the graph.
 - B. Add the reverse of all of p 's edges to the graph.
 - C. Uncross p with previously selected paths, and add p to P .³
3. $S \leftarrow \{\text{vertices still reachable from } s\}$.
4. Return the (s, t) -path packing P and the (s, t) -cut $\partial^+(S)$.

*/

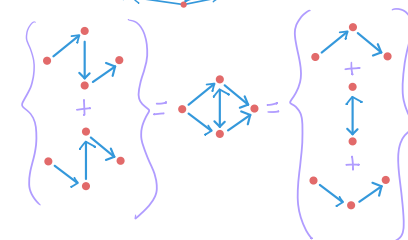
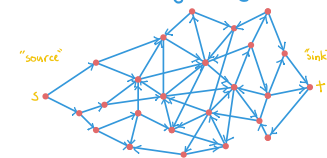
Theorem 14.5 ([FF56]). Let $G = (V, E)$ be a directed graph with m edges, and let $s, t \in V$ be distinct vertices. Let λ be the size of the max-flow/min-cut from s to t . Then $\text{augmenting-paths}(G, s, t)$ produces both a maximum packing and a minimum cut in $O(m\lambda)$ time.

Next time: edge capacities,
fractional packings, & network flow



How to compute

⑦ max # of edge-disjoint $s \rightsquigarrow t$ paths?



Lemma let p, q be edge-disjoint $s \rightsquigarrow t$ paths



then there is $p', q' : s \rightsquigarrow t$ disjoint without opposite pairs of edges



$$\textcircled{1} \left(\begin{array}{c} \text{max \#} \\ \text{disjoint} \\ s \rightsquigarrow t \text{ paths} \end{array} \right) = \left(\begin{array}{c} \text{min \#} \\ \text{edges in any} \\ s \rightsquigarrow t \text{ cut} \end{array} \right)$$

$$\textcircled{2} \left(\begin{array}{c} \# \text{ paths} \\ \text{packed} \end{array} \right) = \left(\begin{array}{c} \text{max \#} \\ \text{disjoint} \\ s \rightsquigarrow t \text{ paths} \end{array} \right)$$

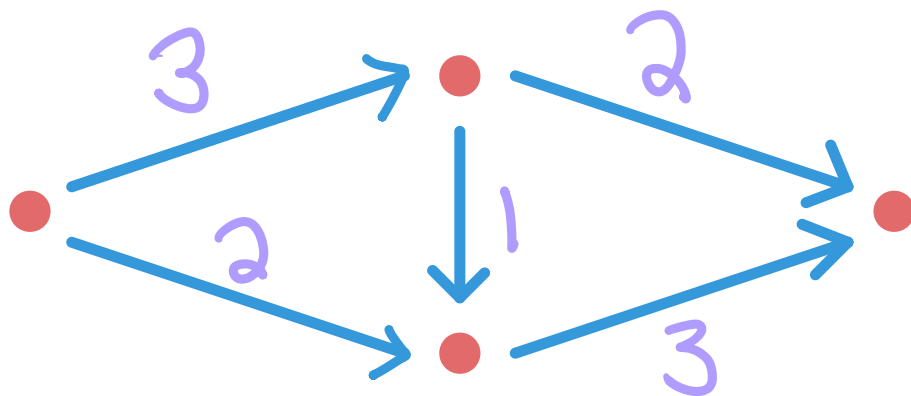
$$\textcircled{3} \left(\begin{array}{c} \text{min \#} \\ \text{edges in any} \\ s \rightsquigarrow t \text{ cut} \end{array} \right) = \left(\begin{array}{c} \# \text{ edges} \\ \text{initially in } C \end{array} \right)$$

capacity $c(e) > 0$: we can reuse
edge e up to $c(e)$ times

1. $P \leftarrow \emptyset$
2. While there is an (s, t) -path $p: s \rightsquigarrow t$:
 - A. Remove all of p 's edges from the graph.
 - B. Add the reverse of all of p 's edges to the graph.
 - C. Uncross p with previously selected paths, and add p to P .³
3. $S \leftarrow \{\text{vertices still reachable from } s\}$.
4. Return the (s, t) -path packing P and the (s, t) -cut $\partial^+(S)$.

*/

Theorem 14.5 ([FF56]). Let $G = (V, E)$ be a directed graph with m edges, and let $s, t \in V$ be distinct vertices. Let λ be the size of the max-flow/min-cut from s to t . Then $\text{augmenting-paths}(G, s, t)$ produces both a maximum packing and a minimum cut in $O(m\lambda)$ time.



capacity $c(e) > 0$: reuse edge e up to $c(e)$ times

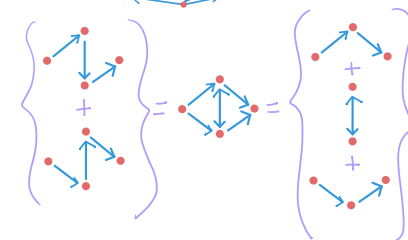
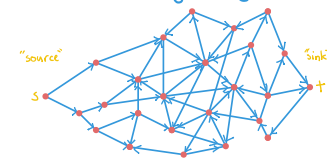
• if $c(e)$ large, Augmenting-Paths not polynomial time

• if $c(e)$ not integral (and just real-valued)

what about max fractional path packing

How to compute

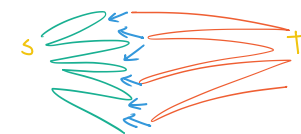
⑦ max # of edge-disjoint $s \rightsquigarrow t$ paths?



Lemma let p, q be edge-disjoint $s \rightsquigarrow t$ paths



then there is $p', q': s \rightsquigarrow t$ disjoint without opposite pairs of edges



$$\textcircled{1} \left(\begin{array}{c} \text{max \#} \\ \text{disjoint } s \rightsquigarrow t \text{ paths} \end{array} \right) = \left(\begin{array}{c} \text{min \#} \\ \text{edges in any } s \rightsquigarrow t \text{ cut} \end{array} \right)$$

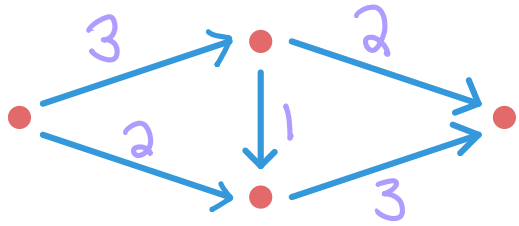
$$\textcircled{2} \left(\begin{array}{c} \# \text{ paths} \\ \text{packed} \end{array} \right) = \left(\begin{array}{c} \text{max \#} \\ \text{disjoint } s \rightsquigarrow t \text{ paths} \end{array} \right)$$

$$\textcircled{3} \left(\begin{array}{c} \text{min \#} \\ \text{edges in any } s \rightsquigarrow t \text{ cut} \end{array} \right) = \left(\begin{array}{c} \# \text{ edges} \\ \text{initially in } C \end{array} \right)$$

1. $P \leftarrow \emptyset$
2. While there is an (s, t) -path $p: s \rightsquigarrow t$:
 - A. Remove all of p 's edges from the graph.
 - B. Add the reverse of all of p 's edges to the graph.
 - C. Uncross p with previously selected paths, and add p to P .³
3. $S \leftarrow \{\text{vertices still reachable from } s\}$.
4. Return the (s, t) -path packing P and the (s, t) -cut $\partial^+(S)$.

*/

Theorem 14.5 ([FF56]). Let $G = (V, E)$ be a directed graph with m edges, and let $s, t \in V$ be distinct vertices. Let λ be the size of the max-flow/min-cut from s to t . Then $\text{augmenting-paths}(G, s, t)$ produces both a maximum packing and a minimum cut in $O(m\lambda)$ time.



capacity $c(e) > 0$:
reuse edge e up to $c(e)$ times

(Weighted) path packings

$P_{s,t} = \{\text{family of all } (s,t)\text{-paths}\}$

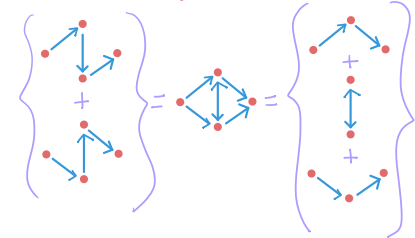
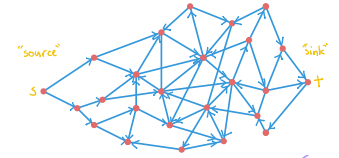
$\chi: P_{s,t} \rightarrow \mathbb{R}_{\geq 0}$

$$\sum_{\substack{p \in P_{s,t} \\ e \in p}} \chi_p \leq c(e)$$

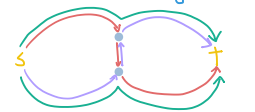
$\forall e \in E$

How to compute

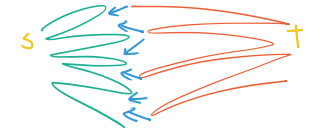
⑦ max # of edge-disjoint $s \rightsquigarrow t$ paths?



Lemma let p, q be edge-disjoint $s \rightsquigarrow t$ paths



then there is $p', q': s \rightsquigarrow t$ disjoint without opposite pairs of edges



① $\left(\begin{smallmatrix} \text{max \#} \\ \text{disjoint} \\ s \rightsquigarrow t \text{ paths} \end{smallmatrix} \right) = \left(\begin{smallmatrix} \text{min \#} \\ \text{edges in any} \\ s \rightsquigarrow t \text{ cut} \end{smallmatrix} \right)$

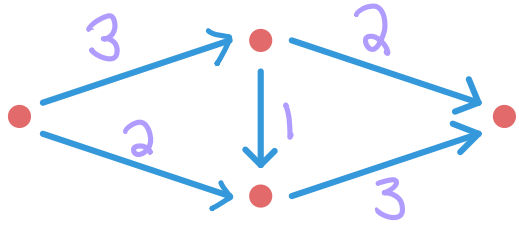
② $\left(\begin{smallmatrix} \# \text{ paths} \\ \text{packed} \end{smallmatrix} \right) = \left(\begin{smallmatrix} \text{max \#} \\ \text{disjoint} \\ s \rightsquigarrow t \text{ paths} \end{smallmatrix} \right)$

③ $\left(\begin{smallmatrix} \text{min \#} \\ \text{edges in any} \\ s \rightsquigarrow t \text{ cut} \end{smallmatrix} \right) = \left(\begin{smallmatrix} \# \text{ edges} \\ \text{initially in } C \end{smallmatrix} \right)$

1. $P \leftarrow \emptyset$
2. While there is an (s, t) -path $p: s \rightsquigarrow t$:
 - A. Remove all of p 's edges from the graph.
 - B. Add the reverse of all of p 's edges to the graph.
 - C. Uncross p with previously selected paths, and add p to P .³
3. $S \leftarrow \{\text{vertices still reachable from } s\}$.
4. Return the (s, t) -path packing P and the (s, t) -cut $\partial^+(S)$.

*/

Theorem 14.5 ([FF56]). Let $G = (V, E)$ be a directed graph with m edges, and let $s, t \in V$ be distinct vertices. Let λ be the size of the max-flow/min-cut from s to t . Then $\text{augmenting-paths}(G, s, t)$ produces both a maximum packing and a minimum cut in $O(m\lambda)$ time.



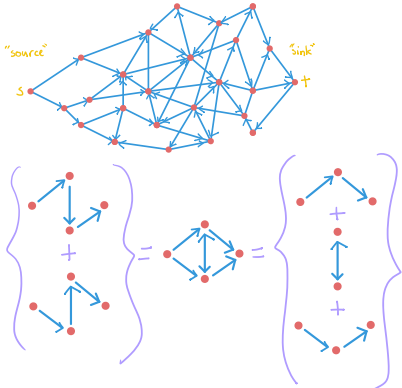
capacity $c(e) > 0$:
reuse edge e up to $c(e)$ times

Capacitated (s, t) -cuts

$$C \subseteq E \Rightarrow \sum_{e \in C} c(e)$$

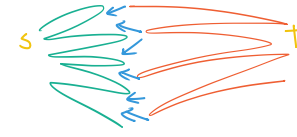
How to compute

⑦ max # of edge-disjoint $s \rightsquigarrow t$ paths?



Lemma let p, q be edge-disjoint $s \rightsquigarrow t$ paths

then there is p', q' $s \rightsquigarrow t$ disjoint without opposite pairs of edges

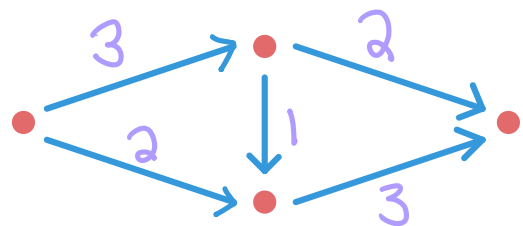


- ① $\left(\begin{smallmatrix} \text{max \#} \\ \text{disjoint} \\ s \rightsquigarrow t \text{ paths} \end{smallmatrix} \right) = \left(\begin{smallmatrix} \text{min \#} \\ \text{edges in any} \\ s \rightsquigarrow t \text{ cut} \end{smallmatrix} \right)$
- ② $\left(\begin{smallmatrix} \# \text{ paths} \\ \text{packed} \end{smallmatrix} \right) = \left(\begin{smallmatrix} \text{max \#} \\ \text{disjoint} \\ s \rightsquigarrow t \text{ paths} \end{smallmatrix} \right)$
- ③ $\left(\begin{smallmatrix} \text{min \#} \\ \text{edges in any} \\ s \rightsquigarrow t \text{ cut} \end{smallmatrix} \right) = \left(\begin{smallmatrix} \# \text{ edges} \\ \text{initially in } C \end{smallmatrix} \right)$

1. $P \leftarrow \emptyset$
2. While there is an (s, t) -path $p: s \rightsquigarrow t$:
 - A. Remove all of p 's edges from the graph.
 - B. Add the reverse of all of p 's edges to the graph.
 - C. Uncross p with previously selected paths, and add p to P .³
3. $S \leftarrow \{\text{vertices still reachable from } s\}$.
4. Return the (s, t) -path packing P and the (s, t) -cut $\partial^+(S)$.

*/

Theorem 14.5 ([FF56]). Let $G = (V, E)$ be a directed graph with m edges, and let $s, t \in V$ be distinct vertices. Let λ be the size of the max-flow/min-cut from s to t . Then augmenting-paths(G, s, t) produces both a maximum packing and a minimum cut in $O(m\lambda)$ time.



capacity $c(e) > 0$:
reuse edge e up to $c(e)$ times

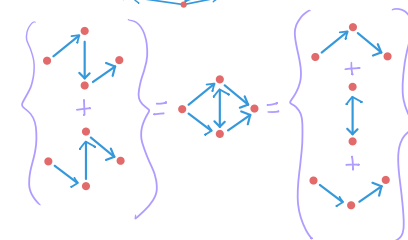
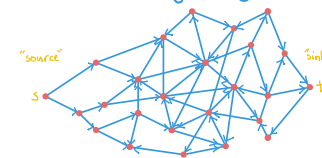
Integer capacities

reduce to "edge-disjoint"

(Max paths) = (Min cut)

How to compute

⑦ max # of edge-disjoint $s \rightsquigarrow t$ paths?



Lemma let p, q be edge-disjoint $s \rightsquigarrow t$ paths



then there is $p', q': s \rightsquigarrow t$ disjoint without opposite pairs of edges



$$\textcircled{1} \left(\begin{array}{c} \text{max \#} \\ \text{disjoint} \\ s \rightsquigarrow t \text{ paths} \end{array} \right) = \left(\begin{array}{c} \text{min \#} \\ \text{edges in any} \\ s \rightsquigarrow t \text{ cut} \end{array} \right)$$

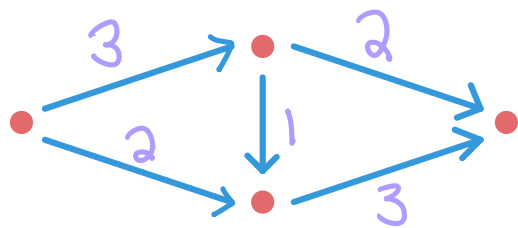
$$\textcircled{2} \left(\begin{array}{c} \# \text{ paths} \\ \text{packed} \end{array} \right) = \left(\begin{array}{c} \text{max \#} \\ \text{disjoint} \\ s \rightsquigarrow t \text{ paths} \end{array} \right)$$

$$\textcircled{3} \left(\begin{array}{c} \text{min \#} \\ \text{edges in any} \\ s \rightsquigarrow t \text{ cut} \end{array} \right) = \left(\begin{array}{c} \# \text{ edges} \\ \text{initially in } C \end{array} \right)$$

1. $P \leftarrow \emptyset$
2. While there is an (s, t) -path $p: s \rightsquigarrow t$:
 - A. Remove all of p 's edges from the graph.
 - B. Add the reverse of all of p 's edges to the graph.
 - C. Uncross p with previously selected paths, and add p to P .³
3. $S \leftarrow \{\text{vertices still reachable from } s\}$.
4. Return the (s, t) -path packing P and the (s, t) -cut $\partial^+(S)$.

*/

Theorem 14.5 ([FF56]). Let $G = (V, E)$ be a directed graph with m edges, and let $s, t \in V$ be distinct vertices. Let λ be the size of the max-flow/min-cut from s to t . Then augmenting-paths(G, s, t) produces both a maximum packing and a minimum cut in $O(m\lambda)$ time.



capacity $c(e) > 0$:
reuse edge e up to $c(e)$ times

Rational capacities

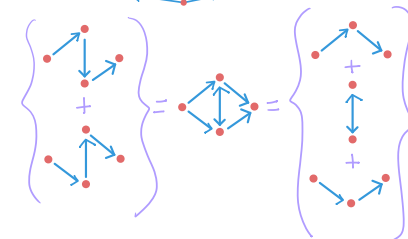
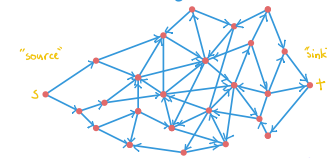
$$c(e) = \frac{33}{100}$$

(clear denominator)

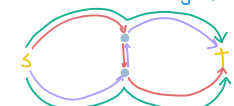
(Max paths) = (Min cut)

How to compute

⑦ max # of edge-disjoint $s \rightsquigarrow t$ paths?



Lemma let p, q be edge-disjoint $s \rightsquigarrow t$ paths



then there is $p', q': s \rightsquigarrow t$ disjoint without opposite pairs of edges



$$\textcircled{1} \left(\begin{array}{c} \text{max \#} \\ \text{disjoint} \\ s \rightsquigarrow t \text{ paths} \end{array} \right) = \left(\begin{array}{c} \text{min \#} \\ \text{edges in any} \\ s \rightsquigarrow t \text{ cut} \end{array} \right)$$

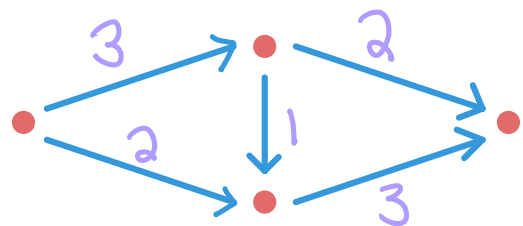
$$\textcircled{2} \left(\begin{array}{c} \# \text{ paths} \\ \text{packed} \end{array} \right) = \left(\begin{array}{c} \text{max \#} \\ \text{disjoint} \\ s \rightsquigarrow t \text{ paths} \end{array} \right)$$

$$\textcircled{3} \left(\begin{array}{c} \text{min \#} \\ \text{edges in any} \\ s \rightsquigarrow t \text{ cut} \end{array} \right) = \left(\begin{array}{c} \# \text{ edges} \\ \text{initially in } C \end{array} \right)$$

1. $P \leftarrow \emptyset$
2. While there is an (s, t) -path $p: s \rightsquigarrow t$:
 - A. Remove all of p 's edges from the graph.
 - B. Add the reverse of all of p 's edges to the graph.
 - C. Uncross p with previously selected paths, and add p to P .³
3. $S \leftarrow \{\text{vertices still reachable from } s\}$.
4. Return the (s, t) -path packing P and the (s, t) -cut $\partial^+(S)$.

*/

Theorem 14.5 ([FF56]). Let $G = (V, E)$ be a directed graph with m edges, and let $s, t \in V$ be distinct vertices. Let λ be the size of the max-flow/min-cut from s to t . Then augmenting-paths(G, s, t) produces both a maximum packing and a minimum cut in $O(m\lambda)$ time.



capacity $c(e) > 0$:
reuse edge e up to $c(e)$ times

Real capacities

$$c(e) = \pi$$

$$\tilde{c}(e) = 3.14$$

$\Rightarrow$

$$|c(e) - \tilde{c}(e)|$$

$$\leq .159 \dots$$

choose $\varepsilon > 0$

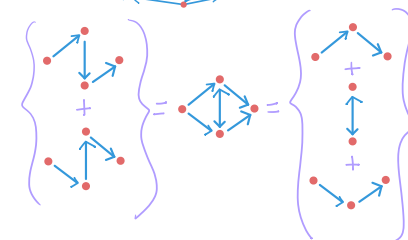
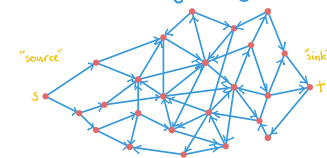
pick rational capacities

$$|\tilde{c}(e) - c(e)| \leq \varepsilon$$

$$(\widetilde{\text{Maxpaths}}) = (\widetilde{\text{Mincut}}) \Rightarrow |(\text{Maxpaths}) - \text{Mincut}| \leq \varepsilon m$$

How to compute

⑦ max # of edge-disjoint $s \rightsquigarrow t$ paths?



Lemma let p, q be edge-disjoint $s \rightsquigarrow t$ paths



then there is $p', q': s \rightsquigarrow t$ disjoint without opposite pairs of edges



$$\textcircled{1} \left(\begin{array}{c} \text{max \#} \\ \text{disjoint} \\ s \rightsquigarrow t \text{ paths} \end{array} \right) = \left(\begin{array}{c} \text{min \#} \\ \text{edges in any} \\ s \rightsquigarrow t \text{ cut} \end{array} \right)$$

$$\textcircled{2} \left(\begin{array}{c} \# \text{ paths} \\ \text{packed} \end{array} \right) = \left(\begin{array}{c} \text{max \#} \\ \text{disjoint} \\ s \rightsquigarrow t \text{ paths} \end{array} \right)$$

$$\textcircled{3} \left(\begin{array}{c} \text{min \#} \\ \text{edges in any} \\ s \rightsquigarrow t \text{ cut} \end{array} \right) = \left(\begin{array}{c} \# \text{ edges} \\ \text{initially in } C \end{array} \right)$$

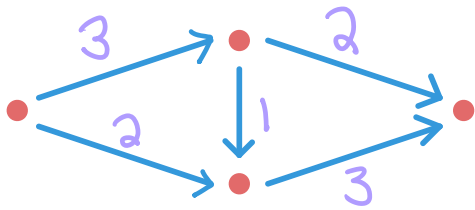
augmenting-paths($G = (V, E), s, t$)

/* Ford and Fulkerson [FF56]

1. $P \leftarrow \emptyset$
2. While there is an (s, t) -path $p : s \rightsquigarrow t$:
 - A. Remove all of p 's edges from the graph.
 - B. Add the reverse of all of p 's edges to the graph.
 - C. Uncross p with previously selected paths, and add p to P .³
3. $S \leftarrow \{\text{vertices still reachable from } s\}$.
4. Return the (s, t) -path packing P and the (s, t) -cut $\partial^+(S)$.

*/

Theorem 14.5 ([FF56]). Let $G = (V, E)$ be a directed graph with m edges, and let $s, t \in V$ be distinct vertices. Let λ be the size of the max-flow/min-cut from s to t . Then $\text{augmenting-paths}(G, s, t)$ produces both a maximum packing and a minimum cut in $O(m\lambda)$ time.



capacity $c(e) > 0$: reuse edge e
up to $c(e)$ times

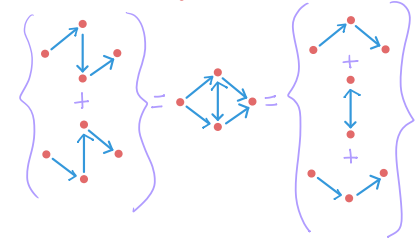
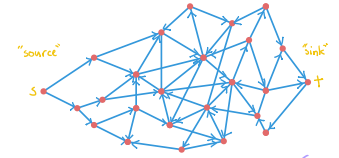
- if $c(e)$ large, Augmenting-Paths not polynomial time
- if $c(e)$ not integral (and just real-valued) what about max fractional path packing

Good news:

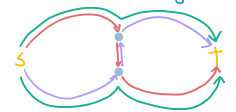
our duality theorems still hold

How to compute

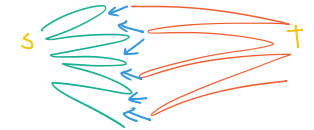
⑦ max # of edge-disjoint $s \rightsquigarrow t$ paths?



Lemma let p, q be edge-disjoint $s \rightsquigarrow t$ paths



then there is $p', q' : s \rightsquigarrow t$ disjoint
without opposite pairs of edges



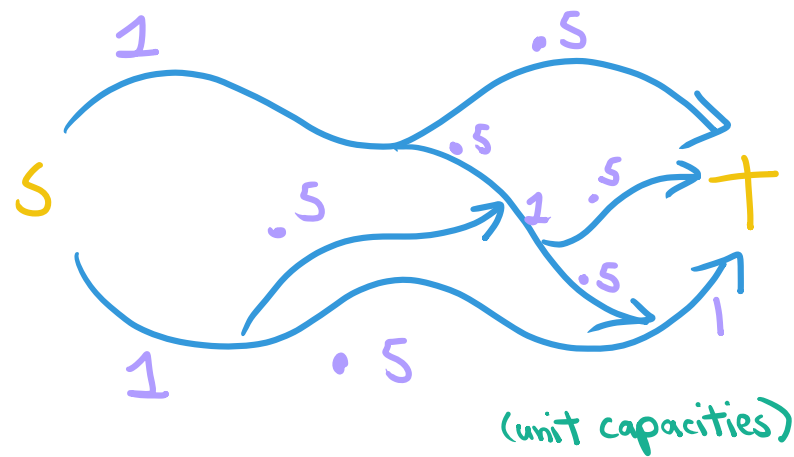
$$\textcircled{1} \left(\begin{array}{l} \text{max \#} \\ \text{disjoint} \\ s \rightsquigarrow t \text{ paths} \end{array} \right) = \left(\begin{array}{l} \text{min \#} \\ \text{edges in any} \\ s \rightsquigarrow t \text{ cut} \end{array} \right)$$

$$\textcircled{2} \left(\begin{array}{l} \# \text{ paths} \\ \text{packed} \end{array} \right) = \left(\begin{array}{l} \text{max \#} \\ \text{disjoint} \\ s \rightsquigarrow t \text{ paths} \end{array} \right)$$

$$\textcircled{3} \left(\begin{array}{l} \text{min \#} \\ \text{edges in any} \\ s \rightsquigarrow t \text{ cut} \end{array} \right) = \left(\begin{array}{l} \# \text{ edges} \\ \text{initially in } C \end{array} \right)$$

remaining issues algorithmic

Network Flow



Max Flow "liquid" version of path packing

Input $G=(V,E)$ directed, capacities $c:E \rightarrow \mathbb{R}_{>0}$

source $s \in V$, sink $t \in V$

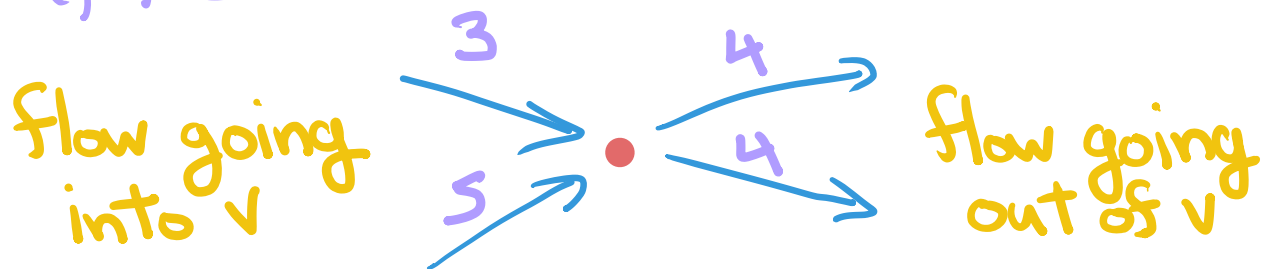
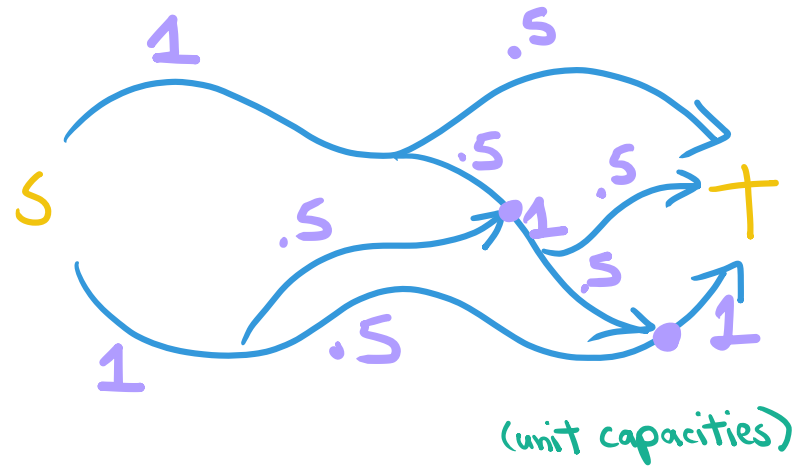
"s-t Flow" $f:E \rightarrow \mathbb{R}_{\geq 0}$ s.t.

(a) "capacity constraints"

$$0 \leq f(e) \leq c(e) \quad \forall e \in E$$

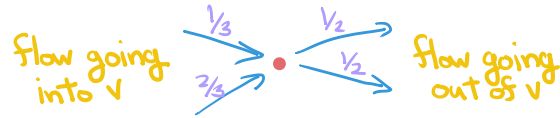
(b) "conservation of flow"

$$\sum_{(u,v) \in E} f(u,v) = \sum_{(v,w) \in E} f(v,w) \quad \forall v \notin \{s,t\}$$



(b) "Conservation of flow"

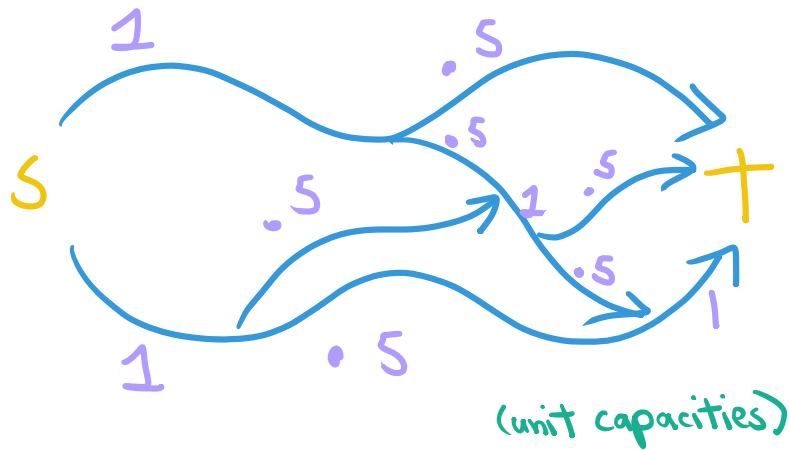
$$\sum_{(u,v) \in E} f(u,v) = \sum_{(v,w) \in E} f(v,w) \quad \forall v \notin \{s,t\}$$



net flow out of s = net flow into t

||

"net flow from s to t",
"size of flow"



Max flow problem

find $s \rightarrow t$ flow of maximum size

"linear program"
net flow out of s

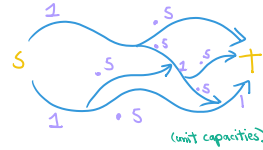
$$\max \quad \sum_{(s \rightarrow w) \in E} f(s, w) - \sum_{(u \rightarrow s) \in E} f(u, s) \quad \text{over } f \in \mathbb{R}^E$$

"capacity constraints" s.t. $0 \leq f(e) \leq c(e) \quad \forall e \in E$

"conservation of flow" $\sum_{(u, v) \in E} f(u, v) = \sum_{(v, w) \in E} f(v, w) \quad \forall v \notin \{s, t\}$

"linear program" because objective,
constraints, all linear functions of f

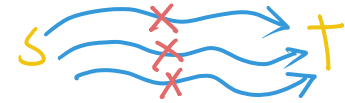
How does Max Flow



relate to

max path packings,

min cut?





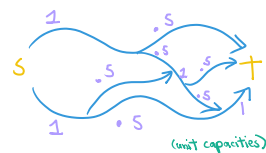
define $f: E \rightarrow \mathbb{R}_{\geq 0}$ as:

A hand-drawn diagram illustrating a stochastic process. Two red wavy lines represent paths starting from a yellow 'S' on the left and ending at a yellow 'T' on the right. A blue arrow labeled 'e' points from the lower path to the upper path. Above the paths, the text $s(x) = 2$ is written in green.

$$(\text{Max Paths}) \leq (\text{Max Flow})$$

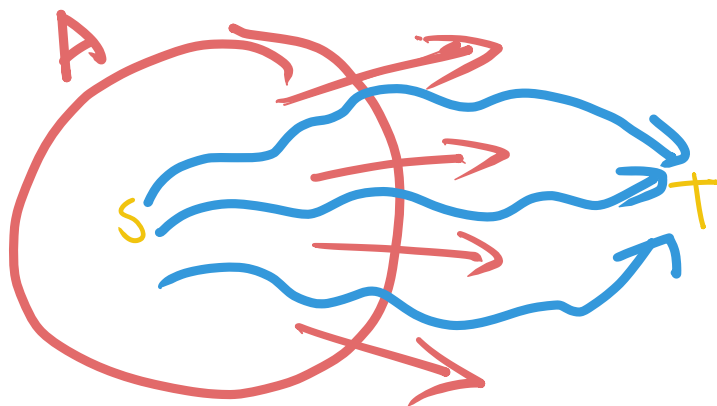


vs



Max Flow

let $C = 2^+(A)$ min-cut, $f: E \rightarrow \mathbb{R}_{\geq 0}$ max flow
 (ACV w/ $s \in A$, $t \notin A$)



(size of f) $\leq$ (net flow out of A) $\leq$ (capacity of $2^+(A)$)

(max flow) $\leq$ (min cut)

max path packings vs Max Flow

let $x: P_{s,t} \rightarrow \mathbb{R}_{\geq 0}$ encode a path packing

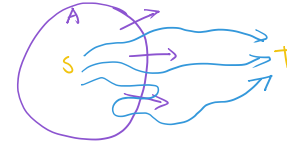
define $f: E \rightarrow \mathbb{R}_{\geq 0}$ as:

$$f(e) = \sum_{p: e \in p} x_p$$

then f is feasible flow of same size

min cut vs Max Flow

let $C = \partial^+(A)$ min-cut, $f: E \rightarrow \mathbb{R}_{\geq 0}$ max flow
($A \subset V$ w/ $s \in A, t \notin A$)

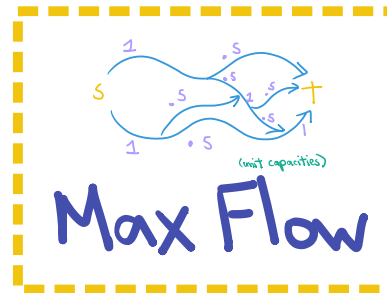


$$(\text{size of } f) = (\text{net flow out of } A) \leq (\text{capacity of } \partial^+(A))$$

we showed



$\leq$



$\leq$



But

equal

 max path packings vs  Max Flow
 let $x: P_{st} \rightarrow \mathbb{R}_{\geq 0}$ encode a path packing
 define $f: E \rightarrow \mathbb{R}_{\geq 0}$ as:

$$f(e) = \sum_{p \ni e} x_p$$

 then f is feasible flow of same size

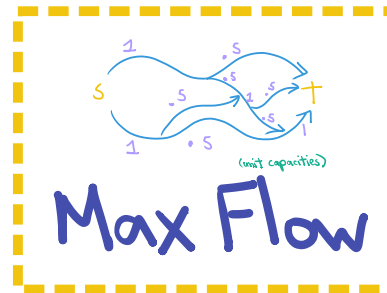
 min cut vs  Max Flow
 let $C = \mathcal{C}(A)$ min-cut, $f: E \rightarrow \mathbb{R}_{\geq 0}$ max flow
 (MFC of unit flow)

 $(\text{size of } f) = (\text{net flow out of } A) \leq (\text{capacity of } \mathcal{C}(A))$

Theorem



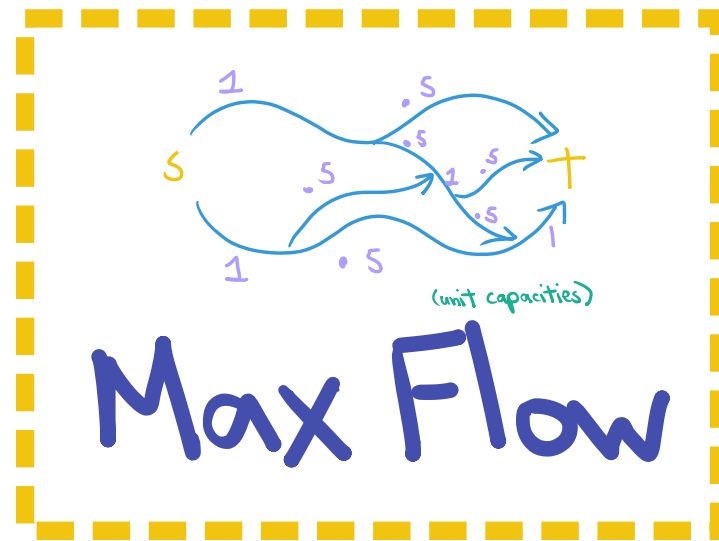
=



=



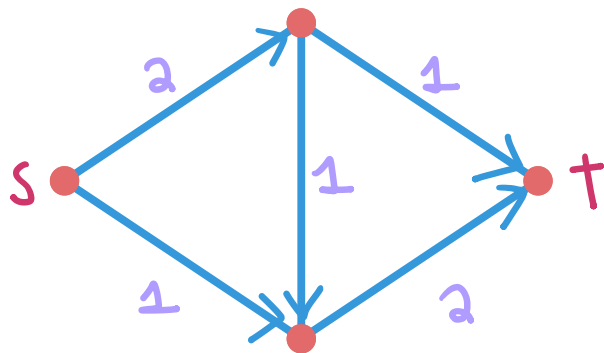
Next: strongly polynomial times



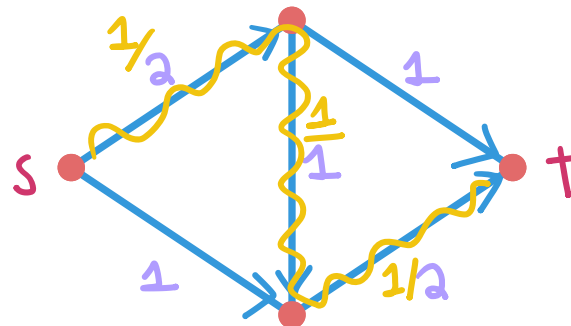
Residual graphs w/r/t flows

(exactly what you would expect)

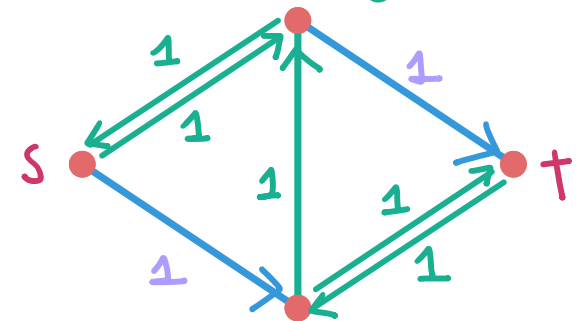
(capacities)



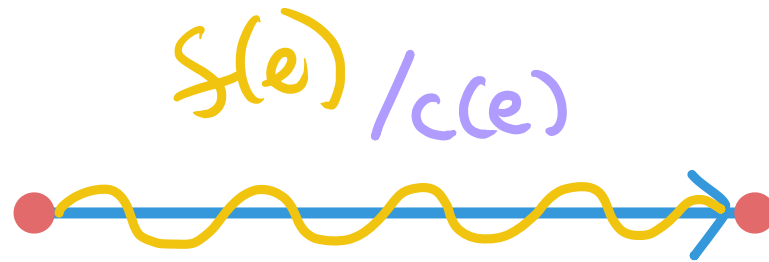
(1 unit flow)



(residual edges)



let $f: E \rightarrow \mathbb{R}_{\geq 0}$ be s-t flow
 $e \in E$ w/ capacity $c(e)$



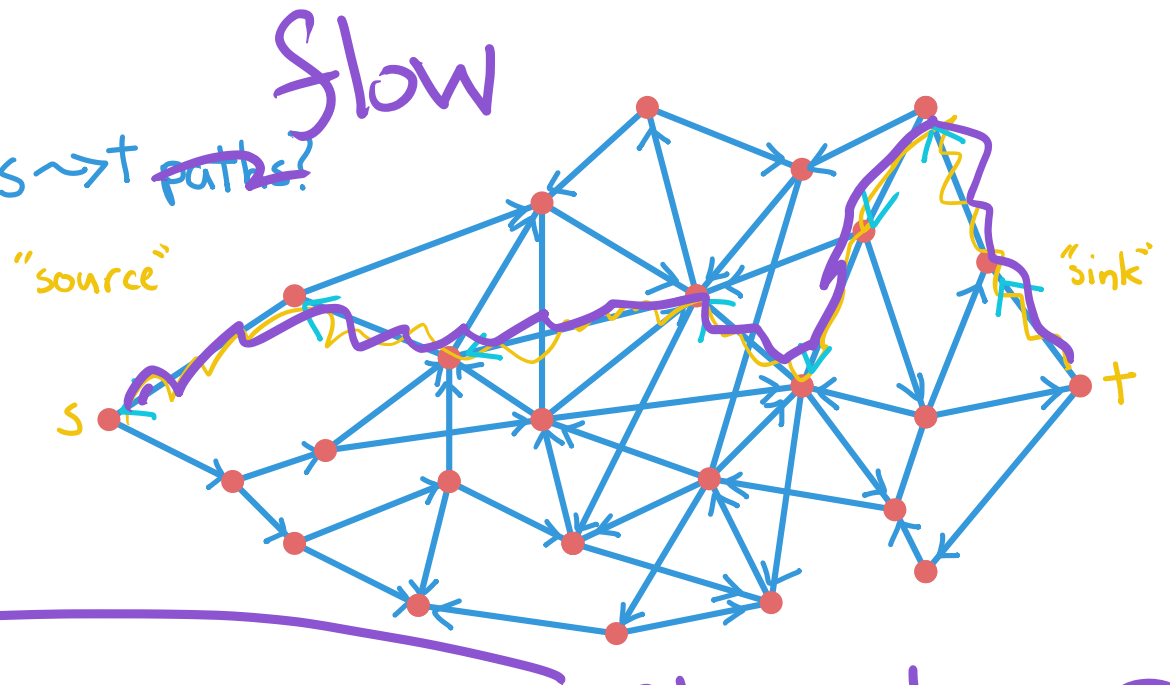
this leaves ~~$f(e)$~~ ~~$c(e)$~~ capacity in forwards direction
 and $f(e)$ capacity backwards



How to compute

⑦ max # of ~~edge-disjoint~~ $s \rightsquigarrow t$ paths?

as much $\rightsquigarrow$ possible



① while there is a path $p: s \rightsquigarrow t$

push slow along p

Ⓐ take p and remove all its edges from the graph

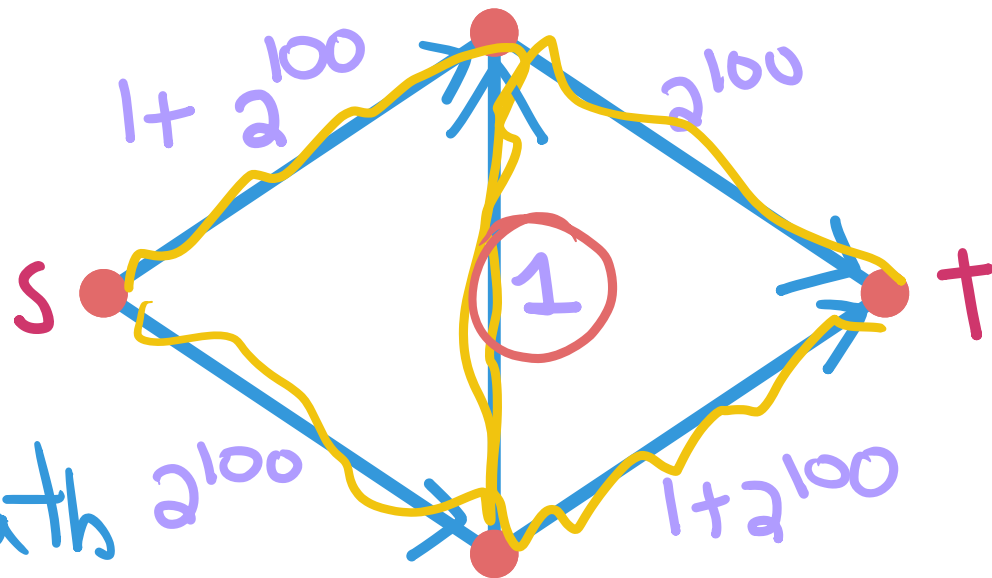
Ⓑ add the reverse of all of p 's edges in the graph

Ⓒ uncross p w/ previously selected paths

update the residual graph

↑
reasonable, but not quite polynomial time

(capacities)



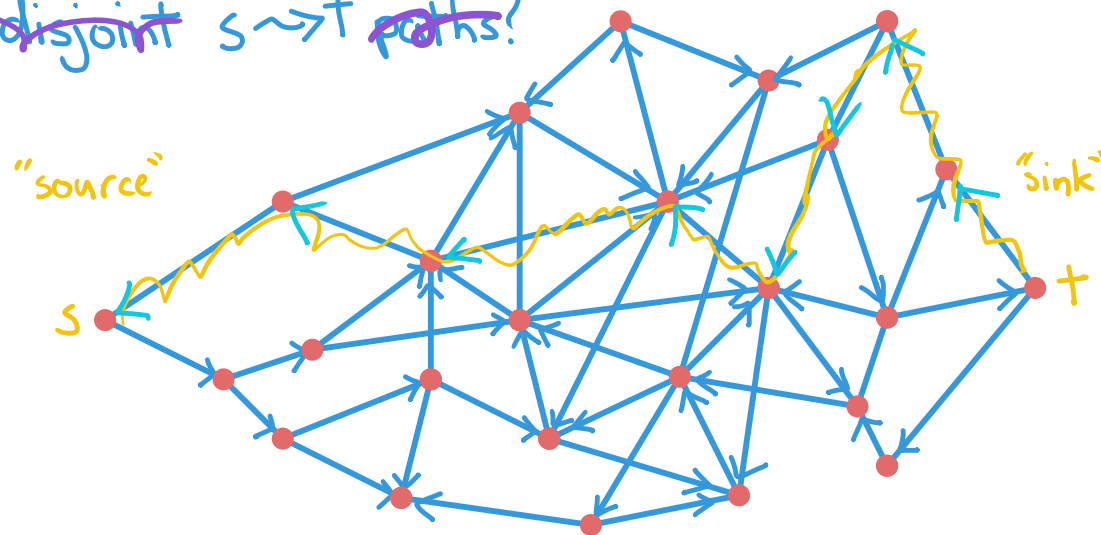
today

① shortest path

② maximize "bottleneck" edge HW

How to compute

⑦ max ~~# of edge-disjoint~~ $s \rightsquigarrow t$ ~~paths?~~ ^{flow}



(Edmonds, Karp)

① while there is a path $p: s \rightsquigarrow t$

Ⓐ take p and ~~remove all its~~ edges ~~from the graph~~

route as much
flow as possible

Ⓑ add the reverse of all of p 's
edges in the graph

Ⓒ ~~uncross p w/ previously selected~~
~~paths~~

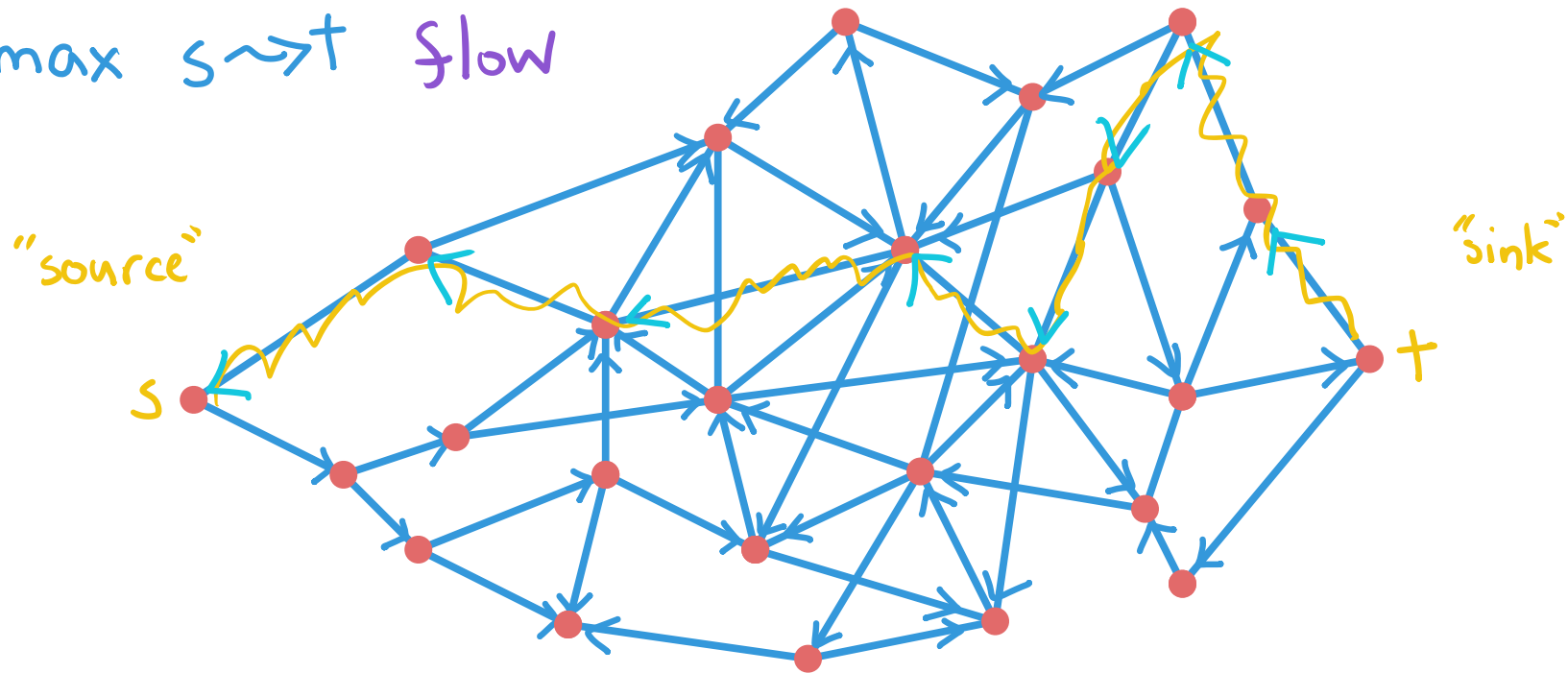
update residual
graph

(cleaned up)

(Edmonds, Karp)

How to compute

⑦ max $s \rightsquigarrow t$ flow



① while there is a path $p: s \rightsquigarrow t$

② take shortest path p and
route as much flow as possible

③ update residual graph

Analysis

① while there is a path $p: s \rightsquigarrow t$

② take shortest path p and
route as much flow as possible

③ update residual graph

Key
Lemma

$\forall v \in V$, distance from s to v

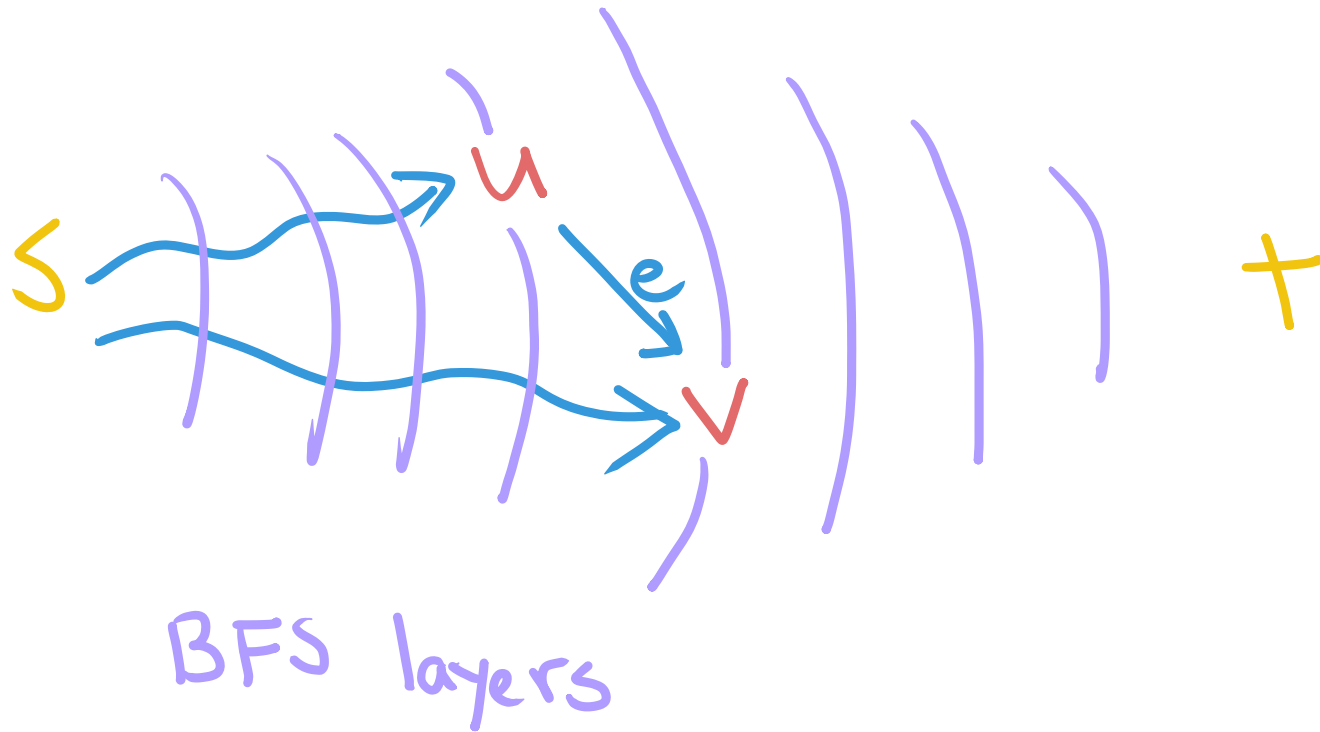
in residual graph never decreases



let us first assume true.

fix $e = (u \rightarrow v) \in E$,

consider distances to u and v

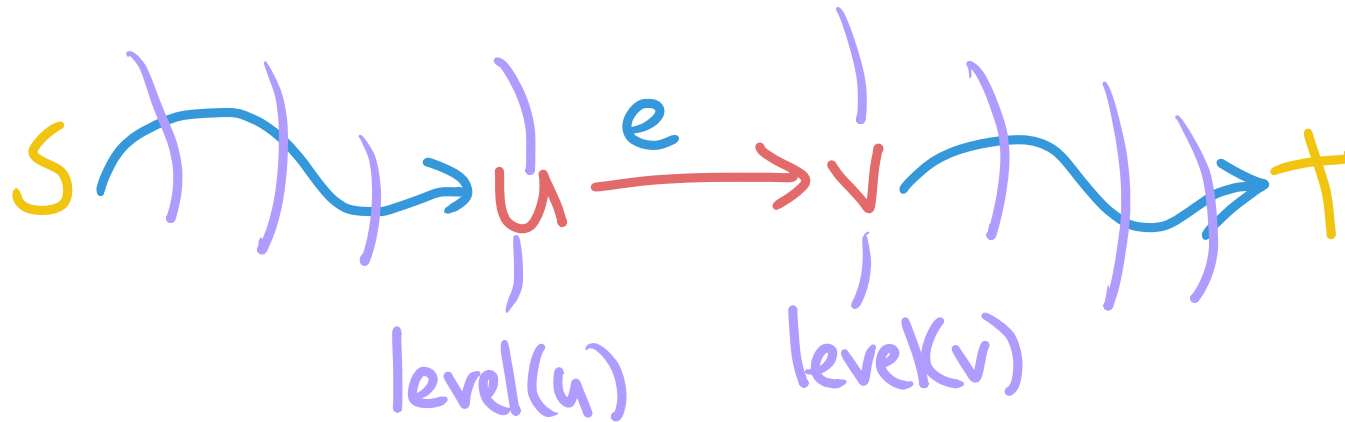


① while there is a path $p: s \rightsquigarrow t$

② take **shortest path** p and
route as much flow as possible
③ update residual graph

Key
Lemma $\forall v \in V$, distance from s to v
in residual graph never decreases

① e removed from RG ,
 $\Rightarrow e$ in shortest $s \rightsquigarrow t$ path



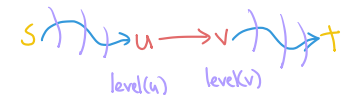
$$\Rightarrow level(v) = level(u) + 1$$

① while there is a path $p: s \rightsquigarrow t$

- ② take **shortest path** p and route as much flow as possible
- ③ update residual graph

Key Lemma $\forall v \in V$, distance from s to v in residual graph never decreases

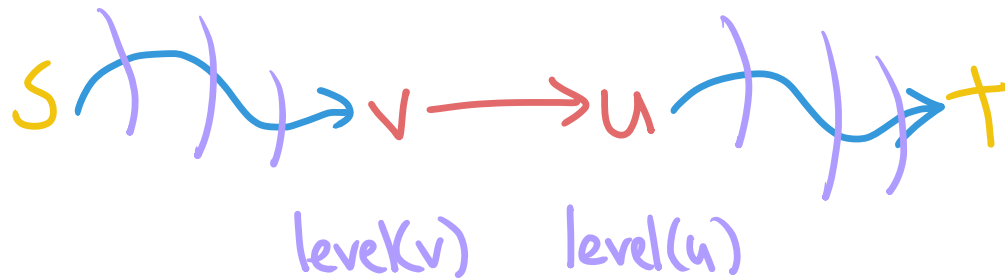
- ① e removed from ^(residual graph) RG ,
 $\Rightarrow e$ in shortest $s \rightsquigarrow t$ path



$$\Rightarrow level(v) = level(u) + 1$$

② $e = (u \rightarrow v)$ added back in to RG

$\Rightarrow (v \rightarrow u)$ was in shortest $s \rightsquigarrow t$ path



$$\Rightarrow level(u) = level(v) + 1$$

④ while there is a path $p: s \rightsquigarrow t$

⑥ take shortest path p and
route as much flow as possible

⑦ update residual graph

Key
Lemma $\forall v \in V$, distance from s to v
in residual graph never decreases

① e removed from ^(residual graph) RG ,
 $\Rightarrow e$ in shortest $s \rightsquigarrow t$ path



$$\Rightarrow level(v) = level(u) + 1$$

② $e = (u \rightarrow v)$ added back in to RG
 $\Rightarrow (v \rightarrow u)$ was in shortest $s \rightsquigarrow t$ path



$$\Rightarrow level(u) = level(v) + 1$$

each event ①/② $\Rightarrow level(u), level(v)$ have switched order.

level's nondecreasing by key lemma

$\Rightarrow e$ gets deleted/added at most $O(n)$ times

Last observation

each iteration removes (at least)
one edge from residual graph

b/c we route as much as possible

= $O(mn)$ iterations

Running time

$O(m)$ per iteration

$O(mn)$ iterations

$O(m^2n)$ time

Key
Lemma $\forall v \in V$, distance from s to v
in resid. graph never decreases

Proof

Suppose we augment along $p: s \rightsquigarrow t$
need to show no levels decrease
induction on level before augmenting

Key
Lemma $\forall v \in V$, distance from s to v
in resid. graph never decreases

Proof

Suppose we augment along $p: s \rightsquigarrow t$
need to show no levels decrease
induction on level before augmenting

Base case

if level was 0, then cannot decrease

Key
Lemma $\forall v \in V$, distance from s to v
in resid. graph never decreases

Proof

Suppose we augment along $p: s \rightsquigarrow t$
need to show no levels decrease
induction on level before augmenting
after

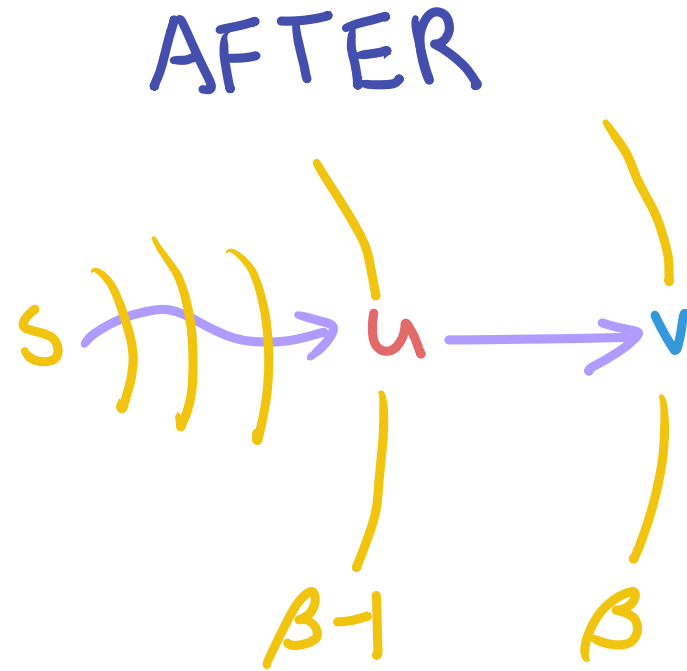
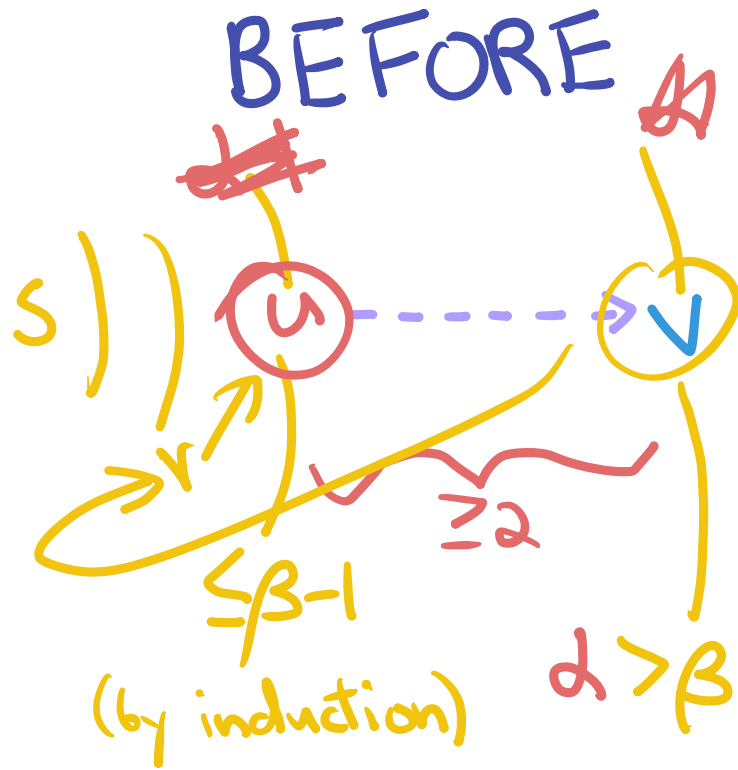
General case suppose level of $v > 0$
(after augmentation)

α = level before

β = level after

suppose by contradiction $\beta < \alpha$.

let u be previous vertex in $s \rightsquigarrow v$ path ^{shortest} after



level $u = \beta - 1$ after

by induction, level of $u \leq \beta - 1$ before

$\Rightarrow u \rightarrow v$ not in RG before graph

$\Rightarrow \underline{v \rightarrow u}$ in augmenting path

$\Rightarrow \underline{\text{level } v} < \underline{\text{level } u}$ before ~~$\Rightarrow$~~

1961?

Analysis

Edmonds, Karp

① while there is a path $p: s \rightsquigarrow t$

② take shortest path p and
route as much flow as possible

③ update residual graph

Key
Lemma $\forall v \in V$, distance from s to v
in resid. graph never decreases

Running time

$O(m)$ per iteration

$O(mn)$ iterations

$O(m^2n)$ time

