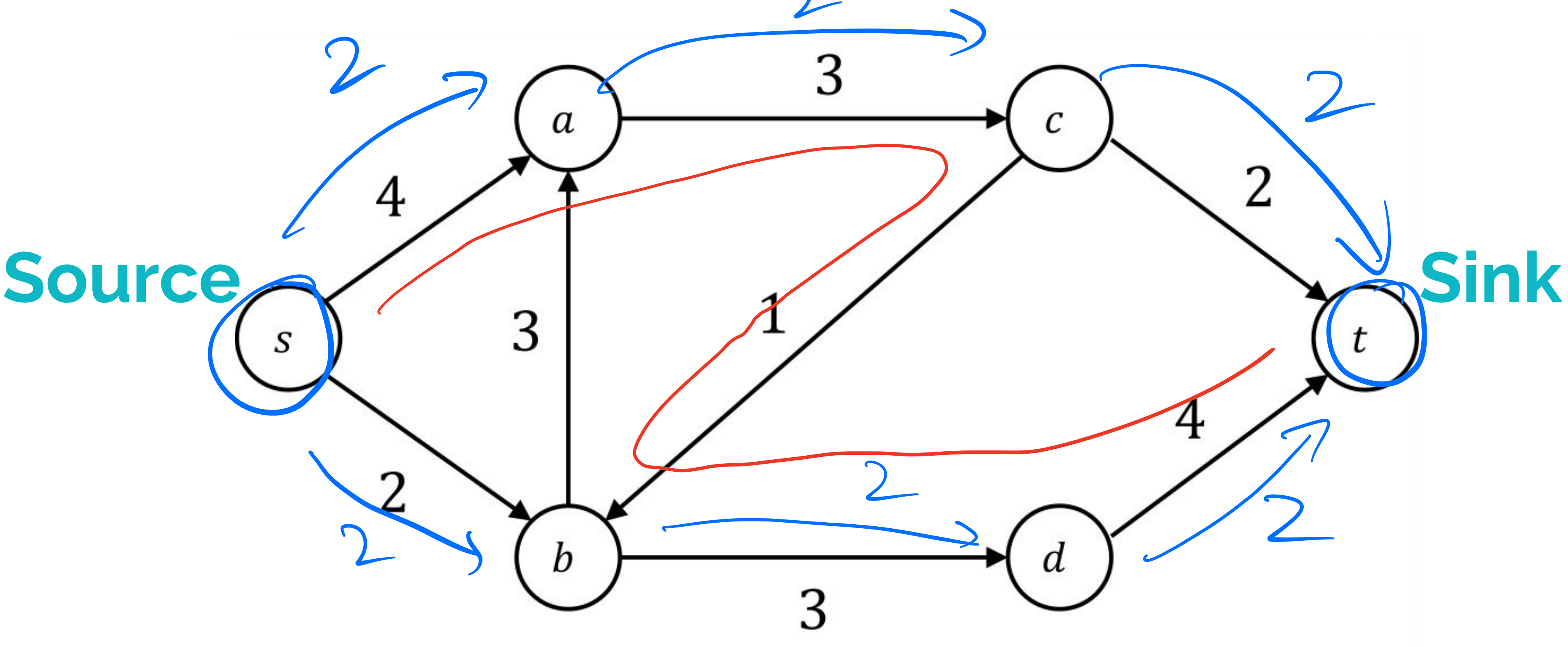
15451 Spring 2023

Max Flow, Min Cut, Ford-Fulkerson

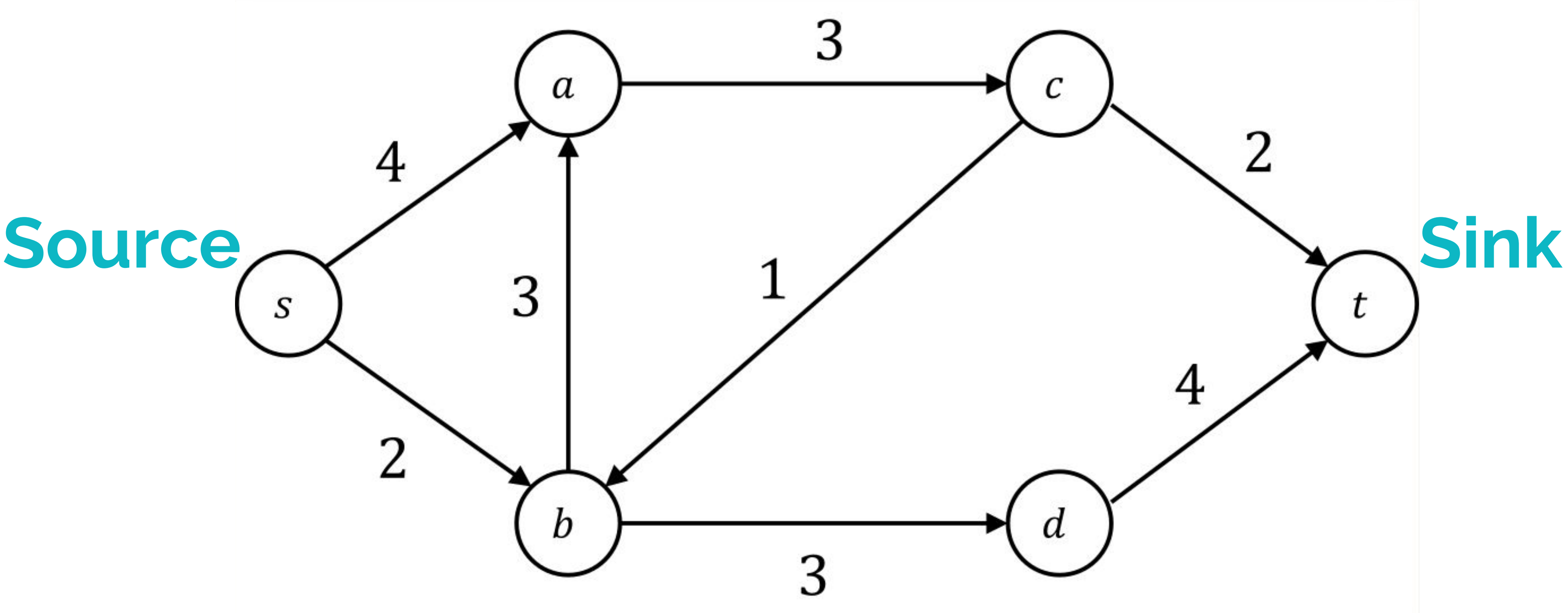
Elaine Shi

The max flow problem



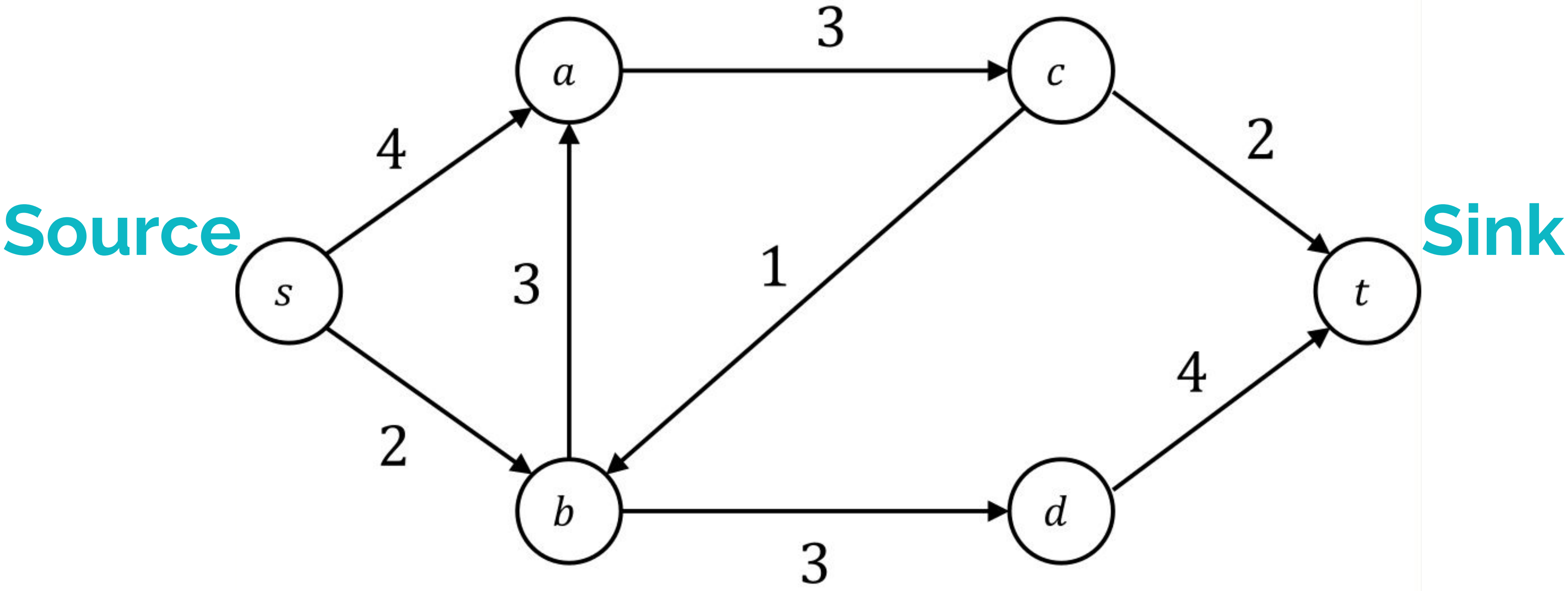
Directed graph G =(V, E), each edge **e** has a **capacity** c(e) ≥ 0

Goal: push maximum amount of flow through



Directed graph, each edge **e** has a **capacity** **c(e)**

Flow example



Definition: flow constraints

$f : \underline{E} \to \underline{\mathbb{R}}$

$\forall e$

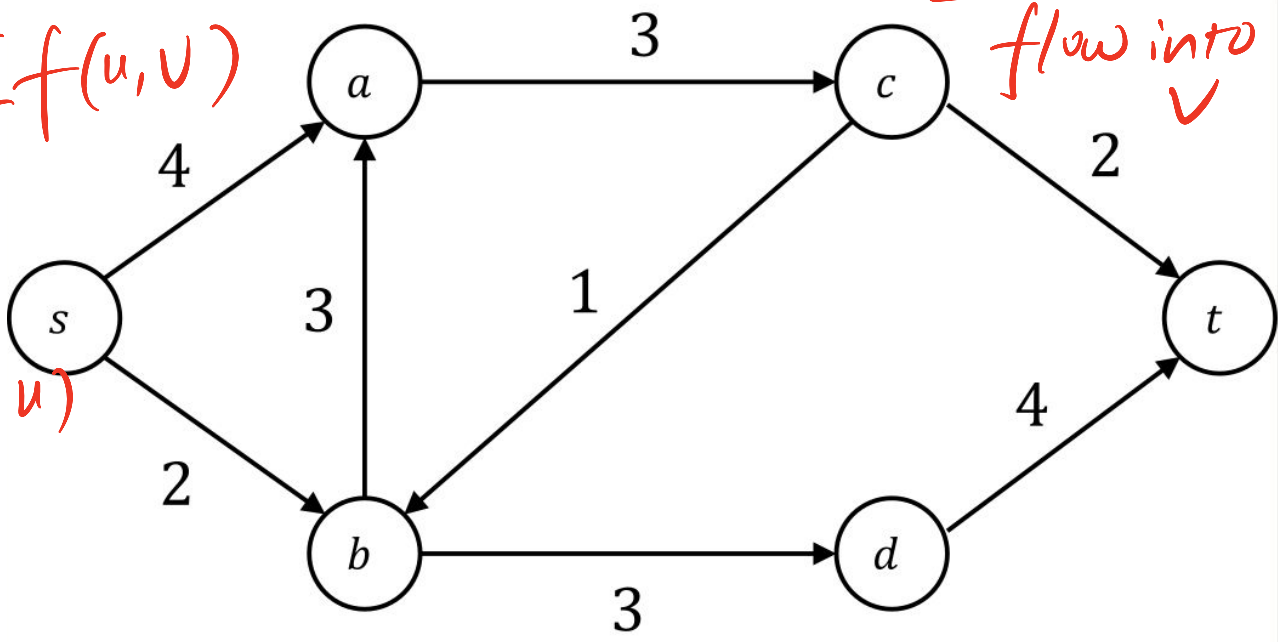
$f(\underline{u,v})$

Capacity constraints: $0 \le f(e) \le c(e)$

Flow conservation: $\forall v \notin \{\underline{s}, \underline{t}\}, \underbrace{\sum_u f(u,v)}_{\text{flow into } v} = \underbrace{\sum_u f(v,u)}_{\text{flow out of } v}$

$f^{in}(v) = \sum_u f(u,v)$

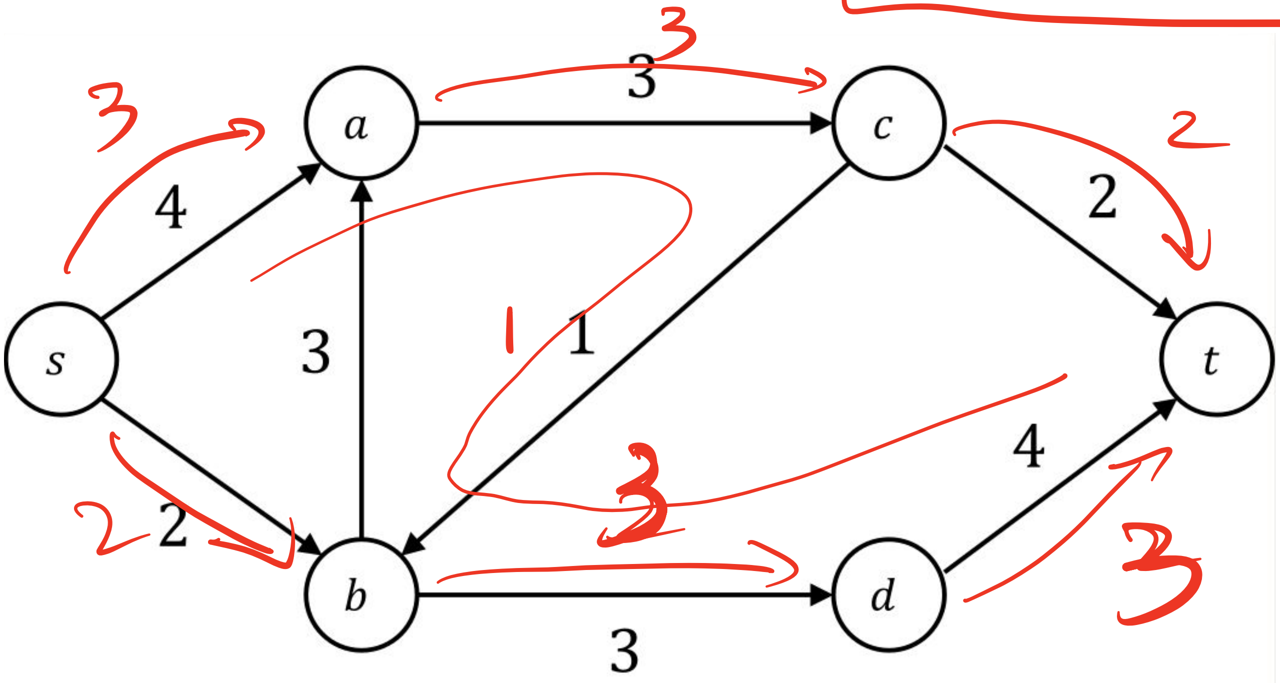
$f^{out}(v) = \sum_u f(v,u)$



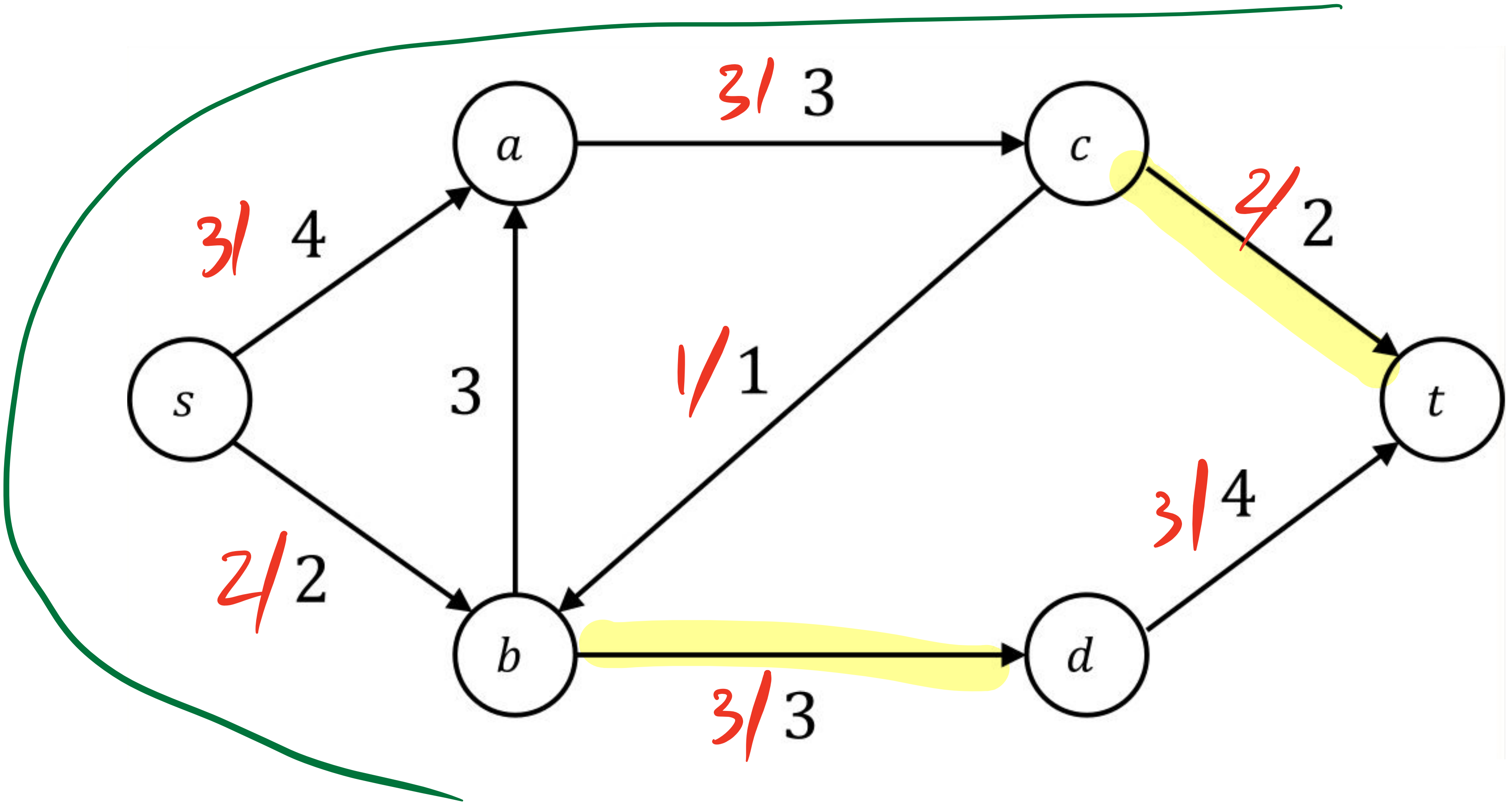
Value of a flow or "net flow from s to t"

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

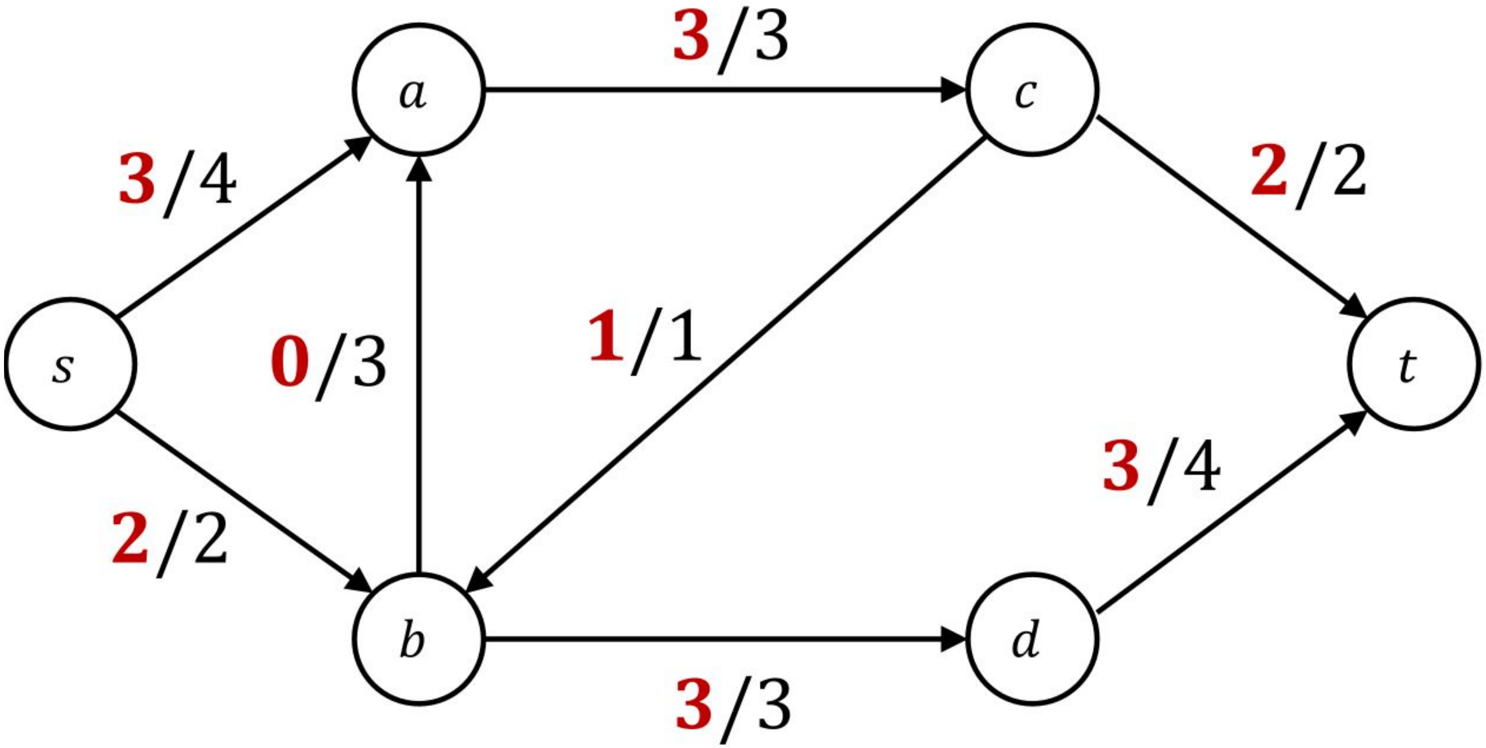
$|f| = 5$



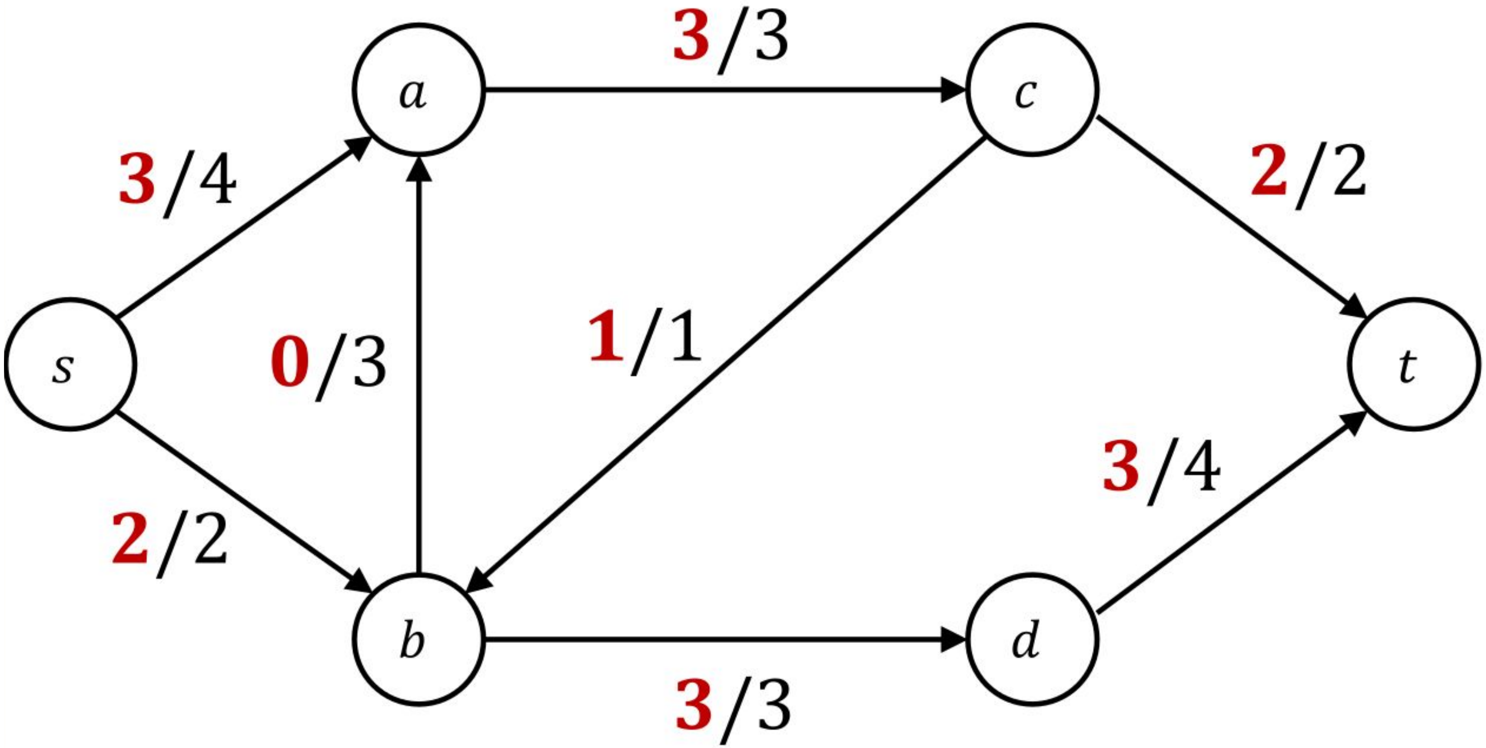
What's the max flow in this graph?



Is this the max flow?



Certifying optimality: s-t cuts



Certifying optimality: s-t cuts

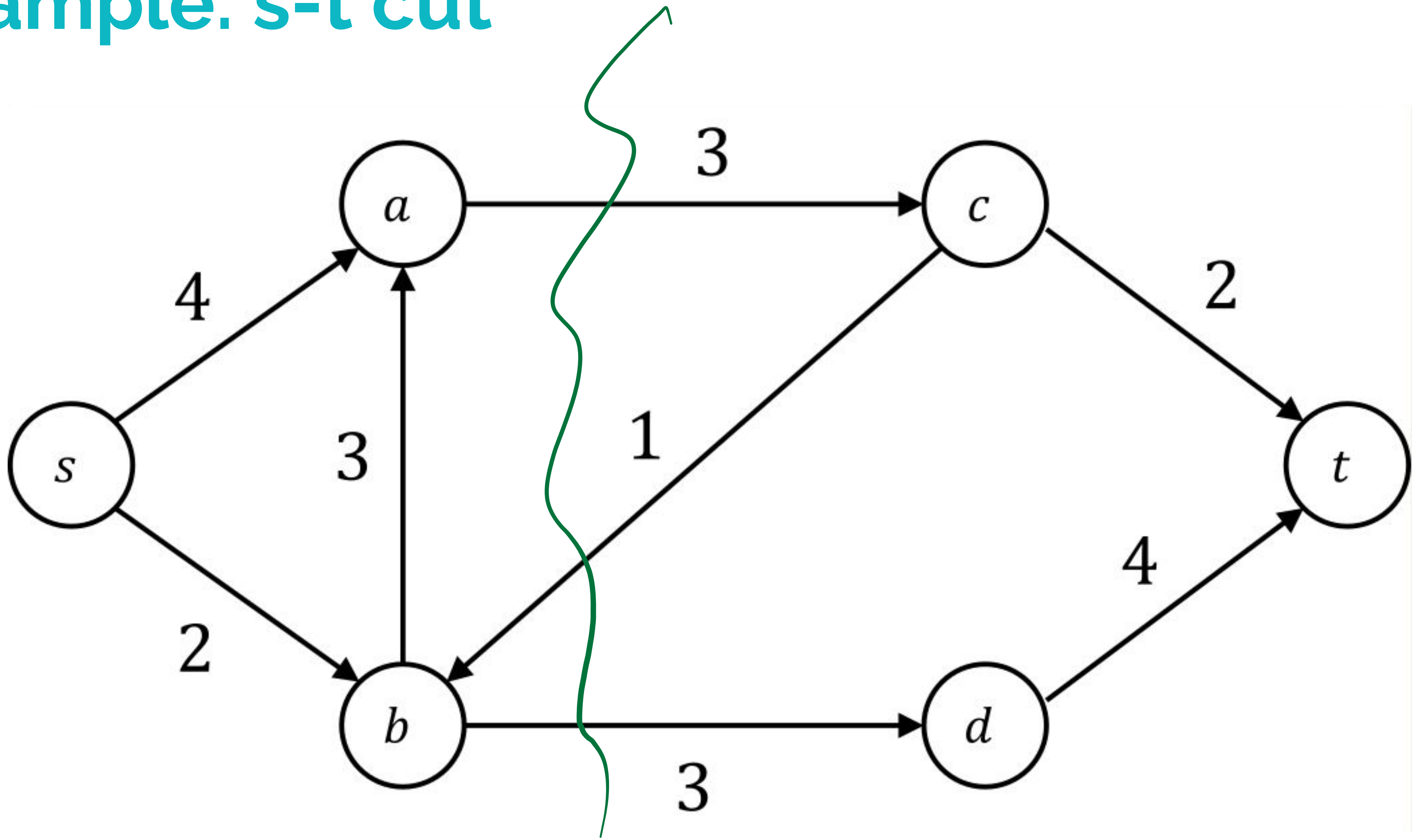
s-t cut: (S, T)

- S, T is a partition of V
- s ∈ S, t ∈ T

Capacity of the cut (S, T)

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

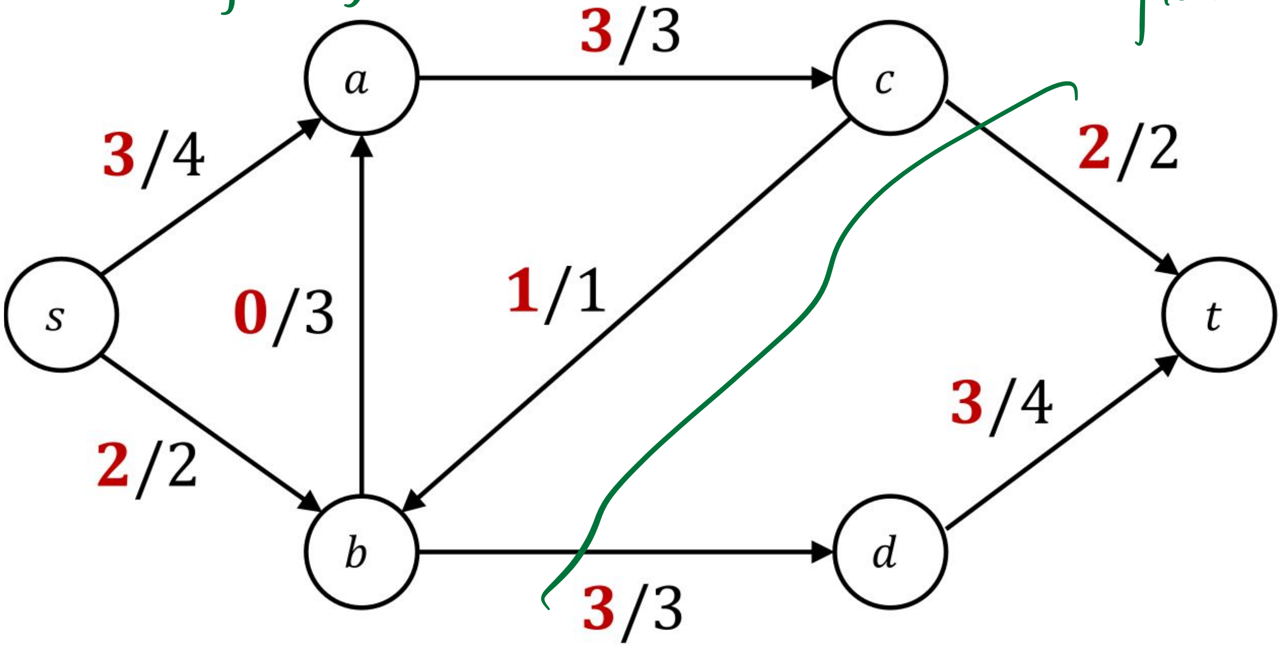
Example: s-t cut



Net flow across a cut (S, T)

Fix some flow f

$$f(\underline{S},\underline{T}) = \underbrace{\sum_{u\in S}\sum_{v\in T} f(u,v)}_{\text{flow going out of the cut from } S} - \underbrace{\sum_{u\in S}\sum_{v\in T} f(v,u)}_{\text{flow into } S}$$



Claim:

For any cut (S, T), $|f| = f(\{s\}, V \setminus \{s\}) = f(S, T)$

Claim:

For any cut (S, T), $|f| = f(\{s\}, V \setminus \{s\}) = f(S, T)$

recall $s \in S$

$$\underline{|f|} = \underline{f^{\text{out}}(s)} - \underline{f^{\text{in}}(s)} = \sum_{v \in S} (f^{\text{out}}(v) - f^{\text{in}}(v))$$

Claim:

For any cut (S, T), |f| = f({s}, V\{s}) = f(S, T)

$$|f| = f^{\text{out}}(s) - f^{\text{in}}(s) = \sum_{v \in S} (f^{\text{out}}(v) - f^{\text{in}}(v))$$

$$= \underbrace{\sum_{e \text{ out of } S} f(e) - \sum_{e \text{ into } S} f(e)}_{= f(S, T)}$$

Claim:

For any s-t cut (S, T), any flow f, $|f| \leq \text{cap}(S, T)$ $\leq f^{out}(S)$ $\leq cap(S,T)$

$|f| = f(S, T) = f^{out}(S) - f^{in}(S)$

i.e., max flow ≤ min cut

↓ we can certify optimality of a flow

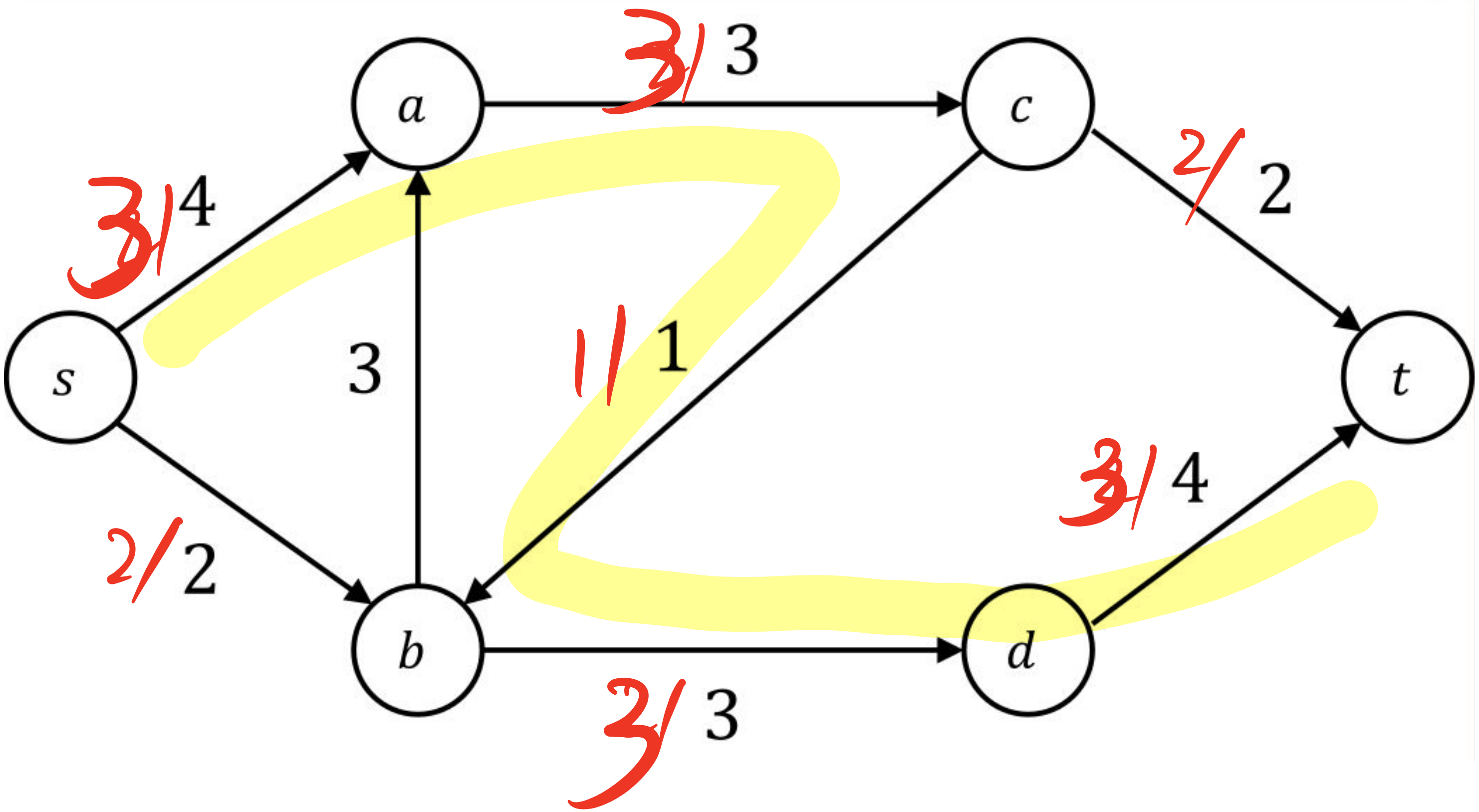
max flow = min cut ?

$\leq$ easy

Max flow = min cut?

How can we find a max flow?

Recall the greedy algorithm



Recall the greedy algorithm

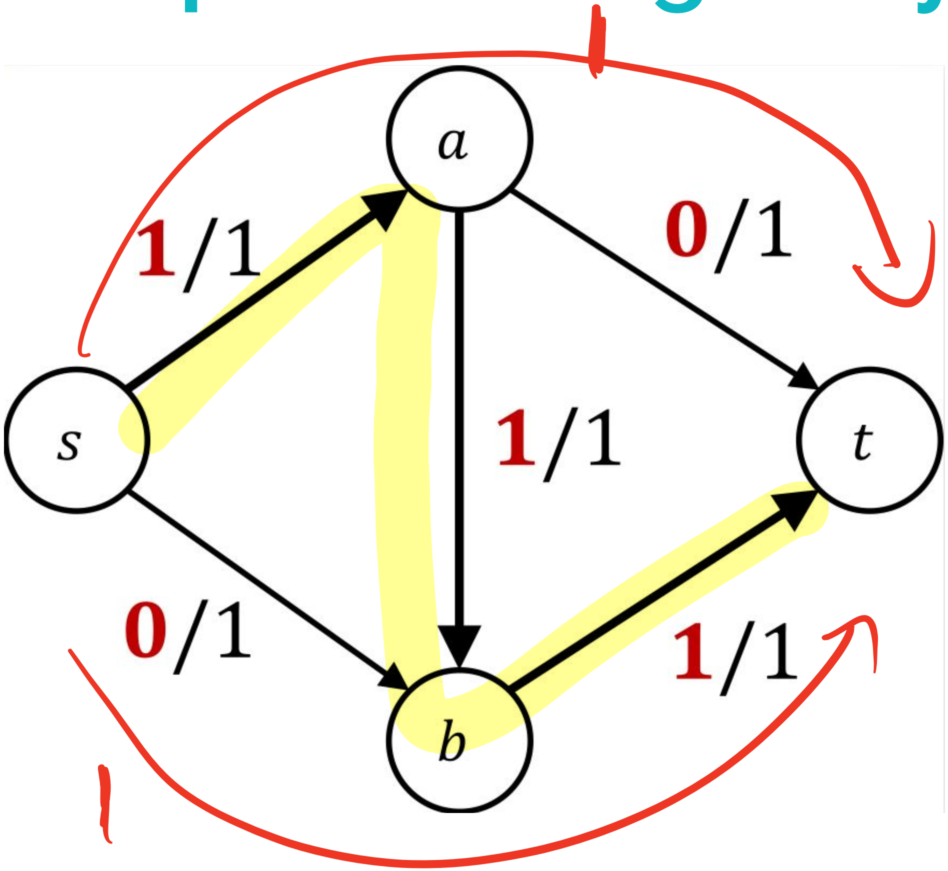
Find a path from s to t, push as much flow through the path as possible

Modify the capacities to the “leftover capacities”

Repeat until no more improvement found

Is this algorithm correct?

A counter-example for the greedy algorithm

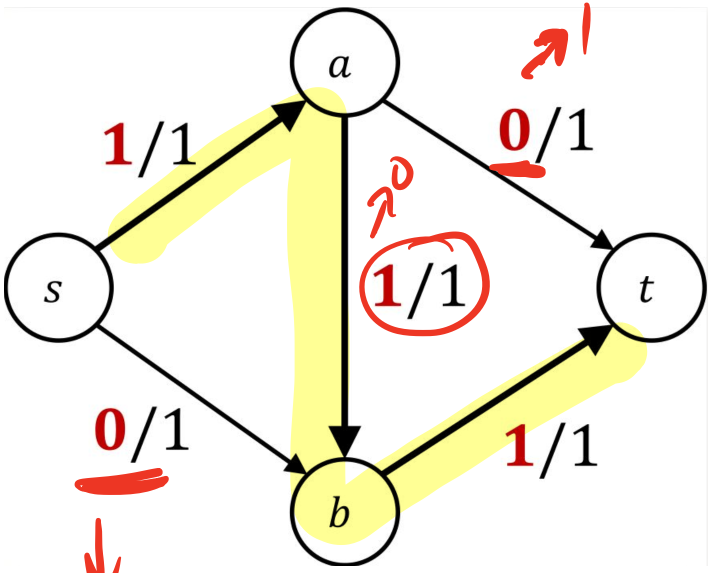


Issue: a→ b is a bad decision

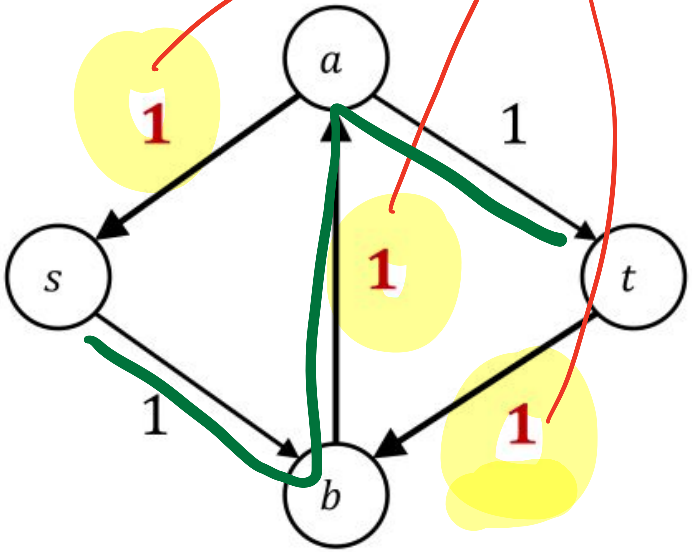
Ford-Fulkerson:

have back-edges in "residual graph"

back-edges



Flow



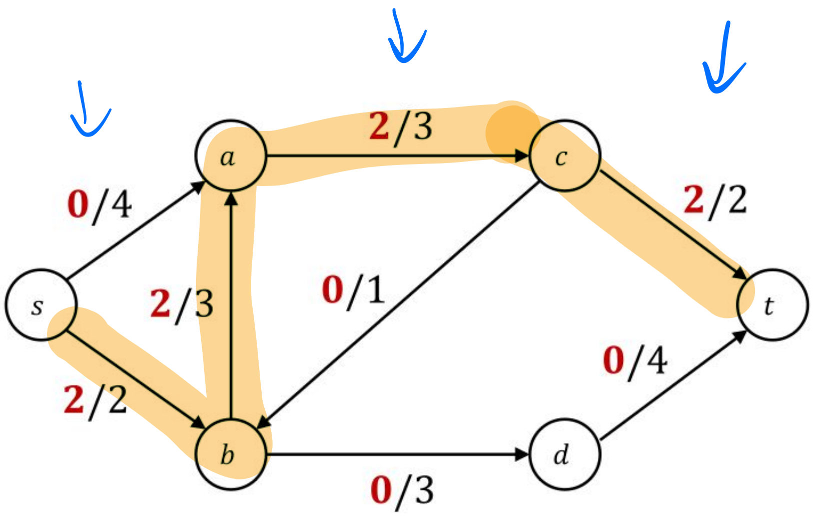
Residual graph

Ford-Fulkerson:
have back-edges in residual graph

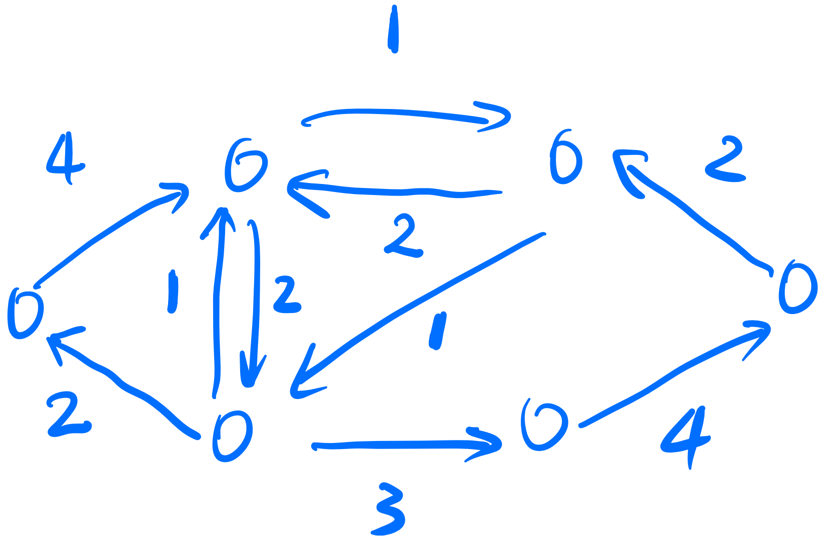
Given a flow f in G, the "residual capacity of edge (u, v)" is defined as:

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

Another example



flow



residual graph

Ford-Fulkerson:

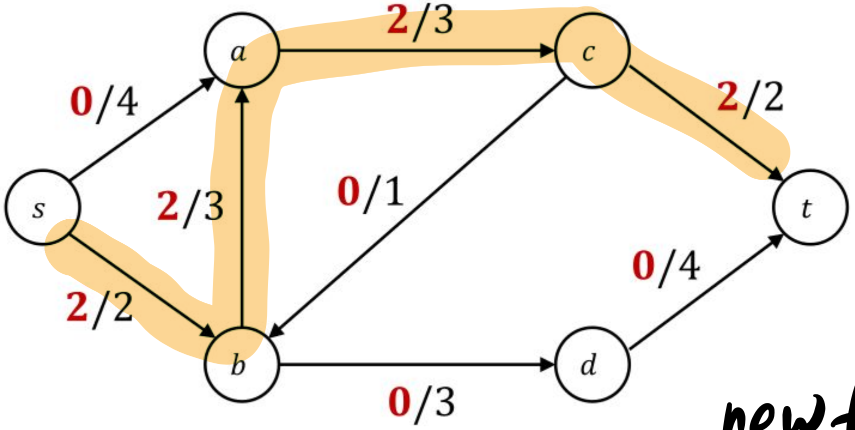
augment = improve

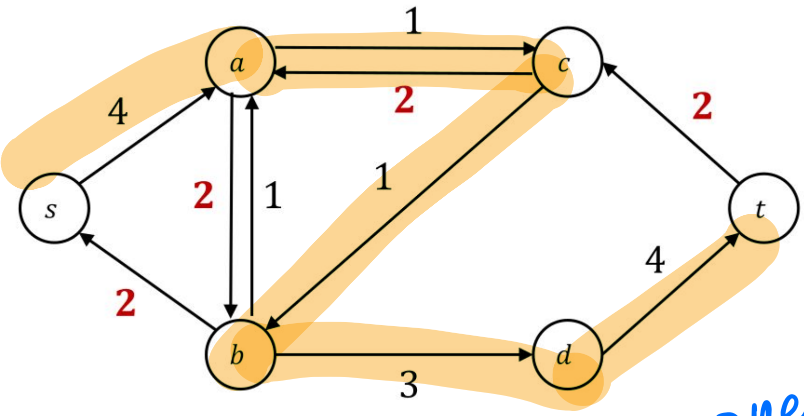
While exists s→t augmenting path P of positive residual capacity

Push the maximum possible flow along P

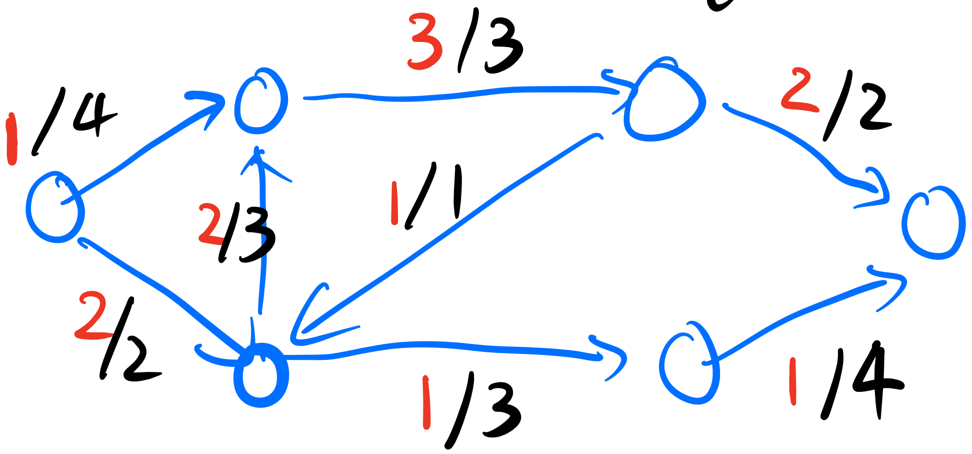
Ford-Fulkerson:

Residual graph

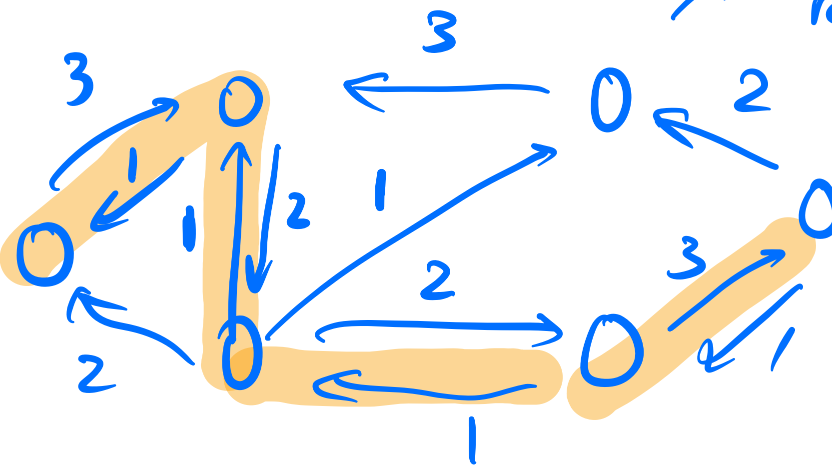




new flow



new residual graph



What’s the running time of FF?

Claim:

If G has integer capacities, FF terminates in time O(mF)

m: # edges

F: value of max flow

Claim:

When FF terminates, output flow value = min cut

Claim:

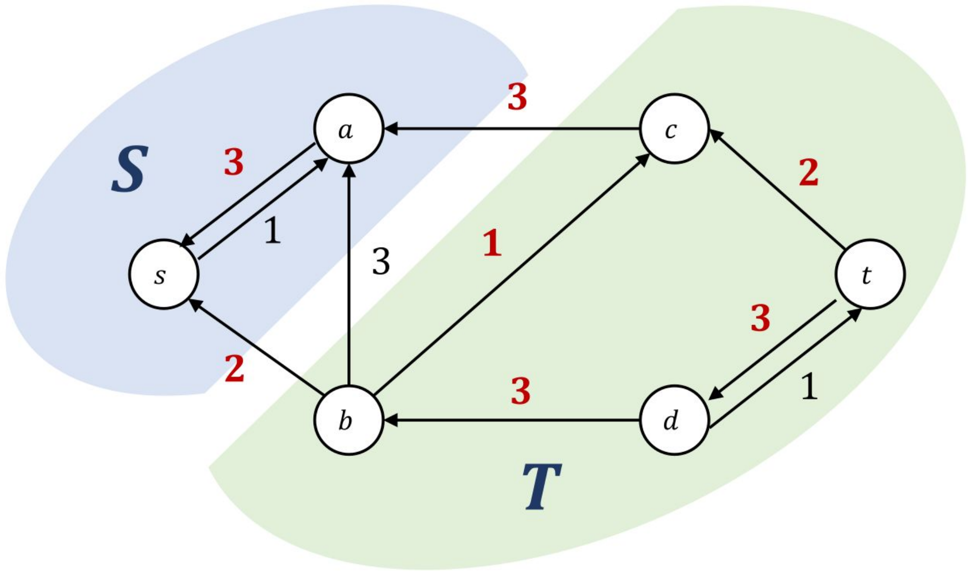
When FF terminates, output flow value = min cut

Recall max flow ≤ min cut

Claim implies: FF finds max flow when it terminates

Claim:

When FF terminates, output flow value = min cut

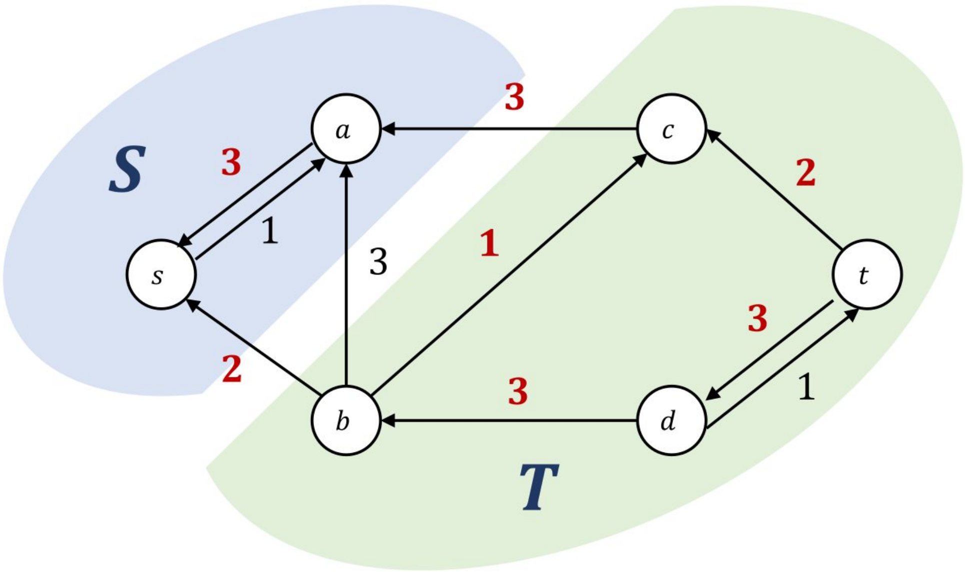


Idea: construct a cut that is equal to flow value

Such a cut must be min cut

Claim:

When FF terminates, output flow value = min cut



Idea: construct a cut that is equal to flow value

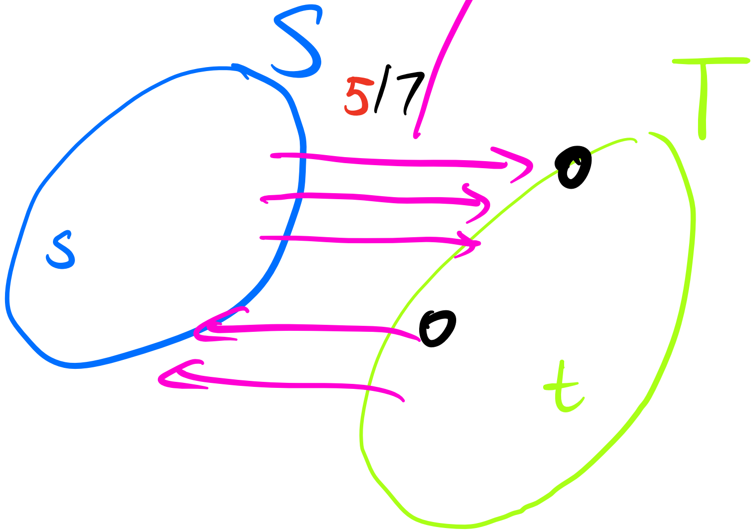
S: vertices reachable from S in **residual graph**

T: V\{S}

Claim:

When FF terminates, output flow value = min cut

edges in original graph G



① forward edge $S \to T$:
must be "saturated"

② reverse edges $T \to S$:
must carry 0 flow

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

S: vertices reachable from S in **residual graph**

T: V\{S}

Theorem: Max flow = min cut

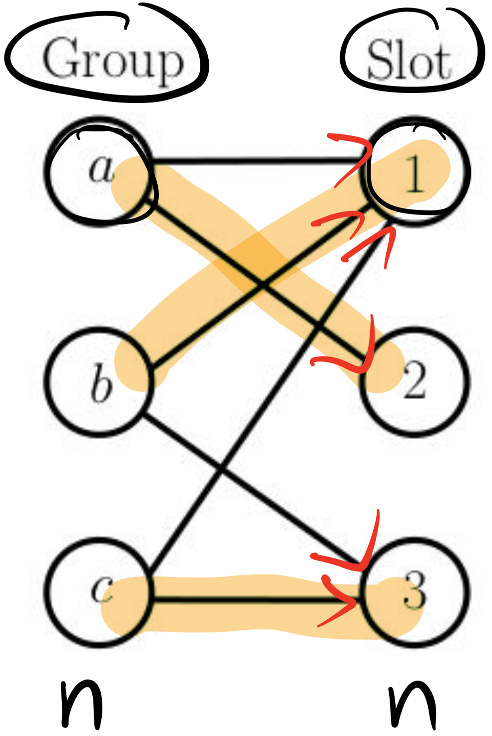
Observation:

for integer capacities, FF finds an integral flow

for integer capacity graph, $\exists$ max flow with integral units

Application: Bipartite matching

Find the maximum matching



it matching size is n

we say it's a perfect matching

matching: set of edges that do not share vertices

Application: Bipartite matching