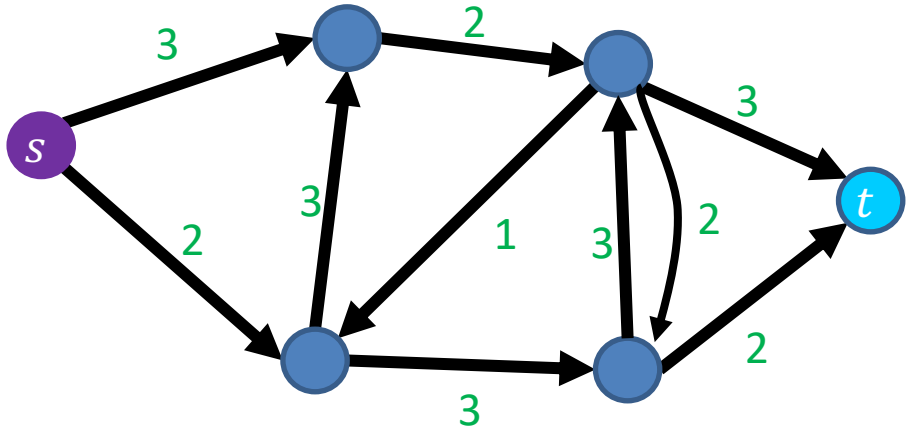
CS4102 Algorithms

Fall 2021 – Horton and Floryan

Min-cut Max-flow, proof of Ford-Fulkerson

Reminders!

Flow Network



Graph $G = (V, E)$

Source node $s \in V$

Sink node $t \in V$

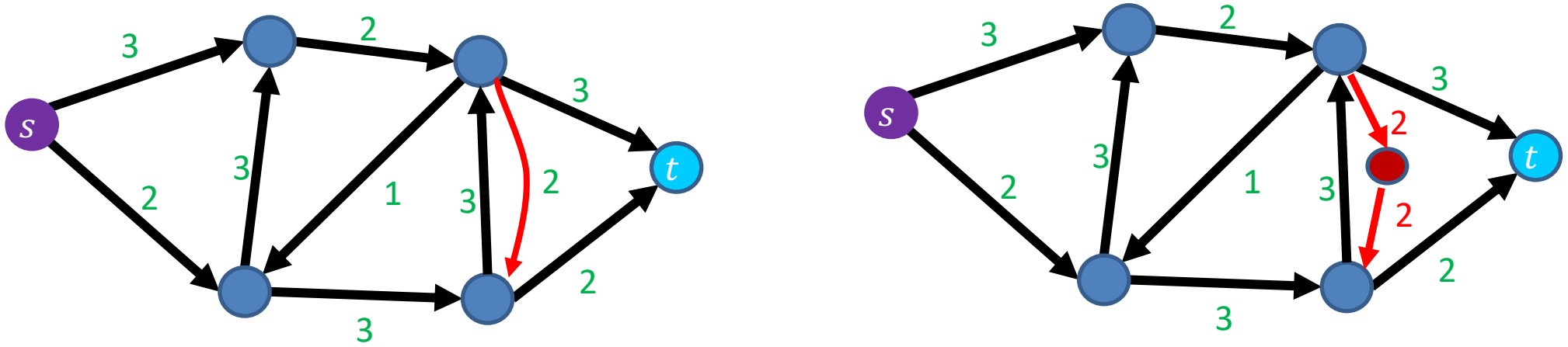
Edge Capacities $c(e)$ ∈ Positive whole* numbers

If $(u, v) \in E$ then $(v, u) \notin E$ (Note our example here violates this!)

Max flow intuition: If s is a faucet, t is a drain, and s connects to t through a network of pipes with given capacities, what is the maximum amount of water which can flow from the faucet to the drain?

Flow Network: Antiparallel Edges

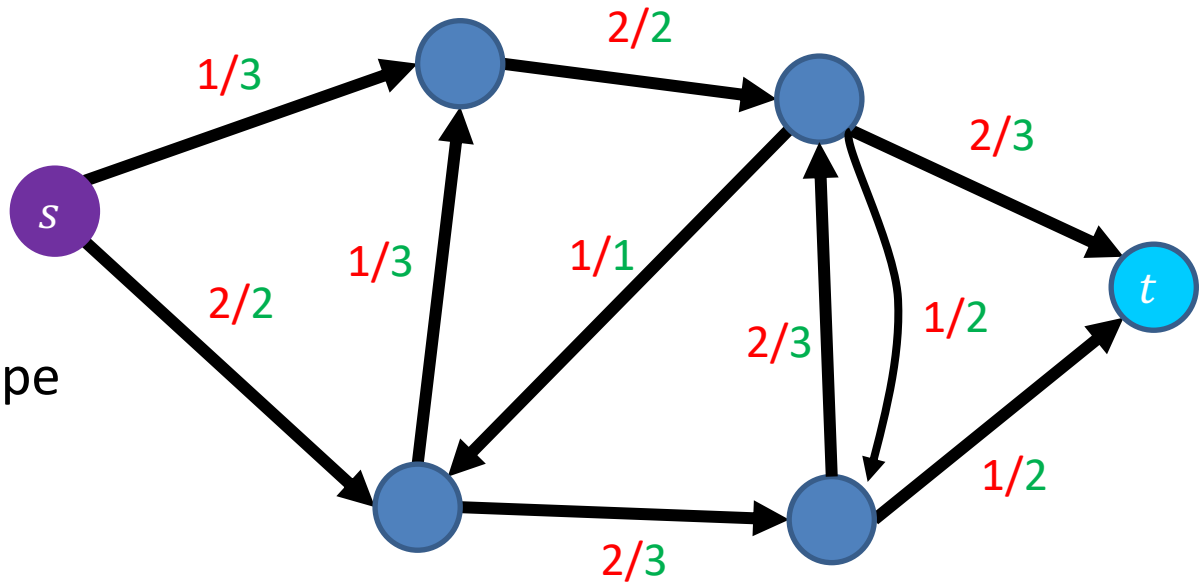
Easy adjustment to remove antiparallel edges and have equivalent flow graph: add intermediate node



(Note: our later examples use graph on the left without this adjustment.)

Flow

- Assignment of values to edges
  - $f(e) = n$
  - E.g. n units of water going through that pipe
- Capacity constraint
  - $f(e) \leq c(e)$
  - Flow cannot exceed capacity
- Flow constraint
  - $\forall v \in V - \{s, t\}, inflow(v) = outflow(v)$
  - $inflow(v) = \sum_{x \in V} f(v, x)$
  - $outflow(v) = \sum_{x \in V} f(x, v)$
  - Water going in must match water coming out
- Flow of G: $|f| = outflow(s) - inflow(s)$
  - Net outflow of s



Flow/Capacity

3 in example above

Ford-Fulkerson: Algorithm overview

- Iterative algorithm: push some flow along some path at each step

- Model or record the *residual* capacities
  - how much capacity is left after taking into account how much flow is going through that edge at this time
- Find a path from *s* to *t* such that the minimum residual capacity of an edge on that path is greater than zero
  - Since each value is an integer, it must be 1 or more
- Update the residual capacities after taking into account this new flow
- Repeat until no more such paths are found

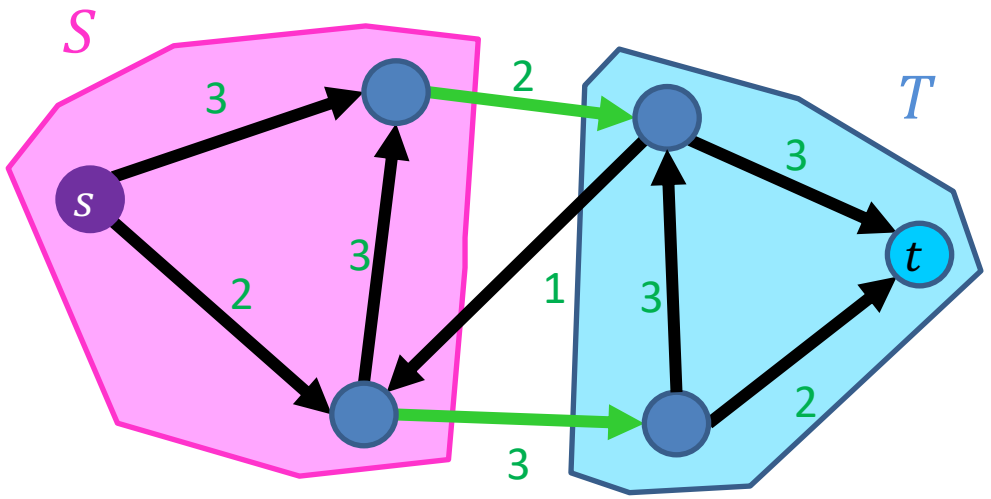
Showing Correctness of Ford-Fulkerson

- To prove Ford-Fulkerson correct
  - We'll use the idea of a **cut** in a network flow graph
  - We'll prove properties related to cuts and flows

Cuts in Network Flow Graphs

- Given a flow network, we want to *cut* edges…

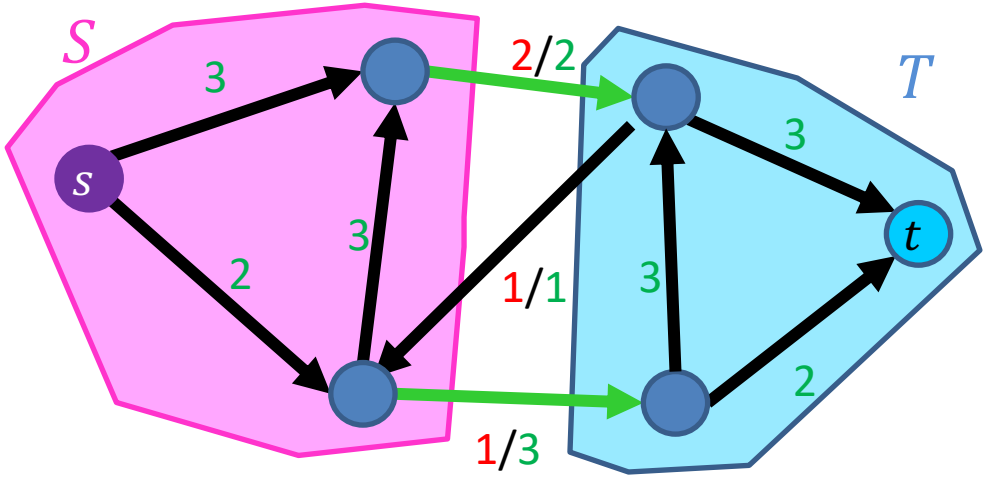
- A cut C = (S, T) where:
  - S is a set of vertices (S is a subset of V)
  - T is a set of vertices (T also a subset of V)
  - S intersect T = null set (no shared vertices)
  - S union T = V (all vertices in either S or T)
  - s in S, t in T



- What do we care about?
  - Well, we care about the edges that go across this cut.
  - Either from node in S to a node in T or vice versa.

Two Properties of a Cut

- Consider a **cut** that separates s and t
  - Let $s \in S$, $t \in T$, s.t. $V = S \cup T$



- **Capacity** of the cut $C = (S, T)$
  - Sum the **capacities** of all edges which go from S to T
  - This example: 2 + 3 = 5
- **Net Flow** of the cut $C = (S, T)$
  - Sum of the **flows** on its edges from S to T minus the sum of the **flows** on its edges from T to S
  - This example: 2 + 1 – 1 = 2

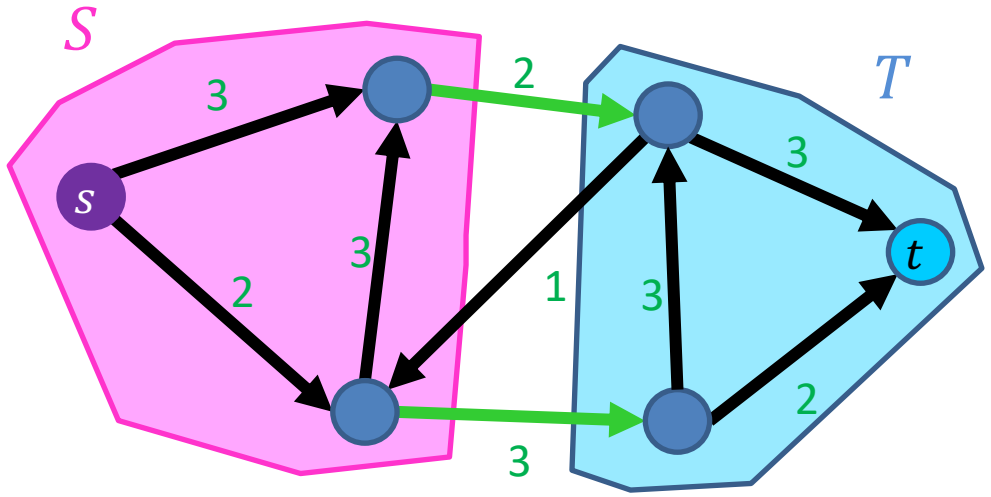
We'll Show These 3 Things

- Weak duality property
- Flow-value lemma
- Max-flow Min-Cut theorem

- …and how we can then argue Ford-Fulkerson is correct

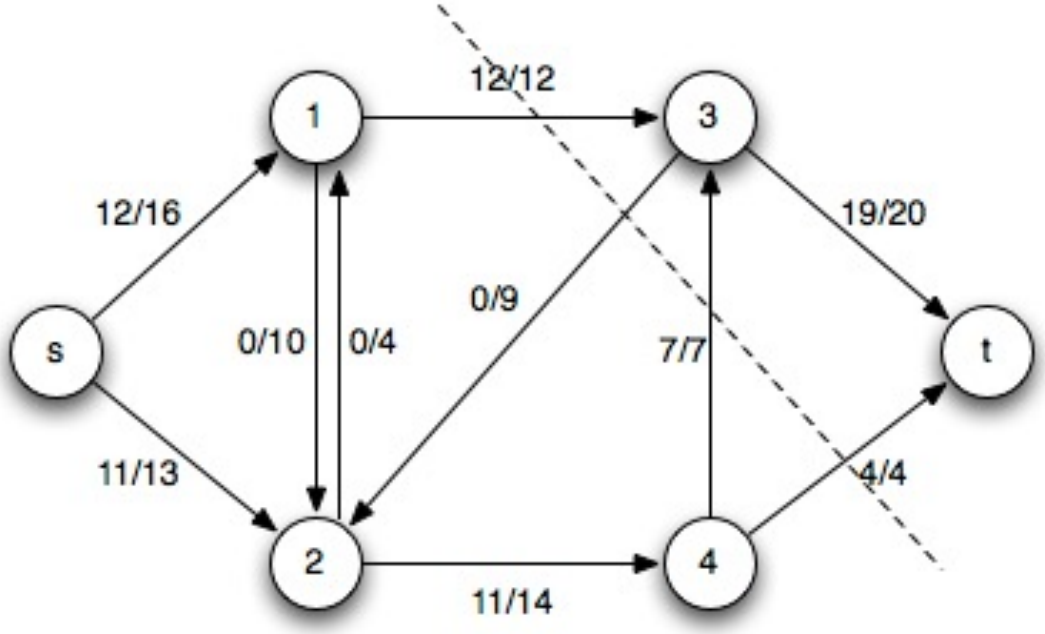
First definition: Weak Duality

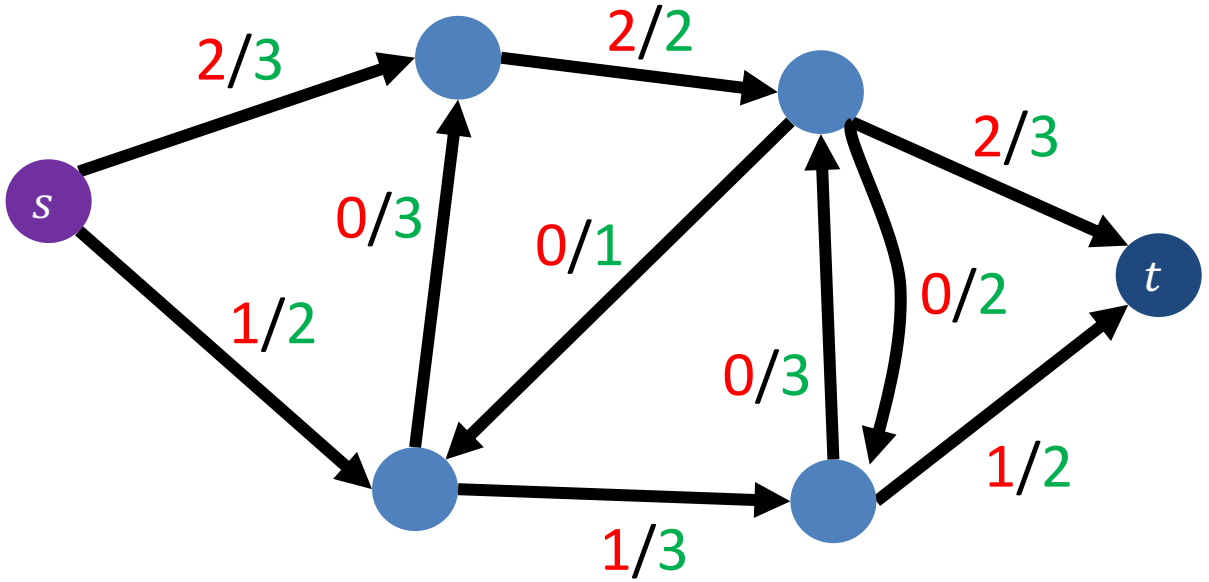
- Let ***f*** be any flow and C = (S, T) be any cut
- Then the value of ***f*** <= capacity of C
  - We'll refer to this as the weak duality property
- Note: We are talking about the **capacity** of C, not the value of the flow across C (i.e. not the net flow of C)



Definition: Flow-value lemma

- Let ***f*** be any flow and C = (S, T) be any cut in G
  - The **net flow** across (S, T) equals the value of the flow ***f***





Proof: Flow-value lemma

- Let ***f*** be any flow and C = (S, T) be any cut
  - The net flow across (S, T) equals the value of the flow ***f***

- Proof by induction on the size of T
  - B.C. T = {t} (T is only the sink)
    - Clearly this is true as the flow across the cut is everyone sinking into t, which is the definition of the flow f
  - I.H. Assume true for some cut C = (S, T)
  - I.S. Move one node from S to T
    - Choose a node p to move that has at least one edge to a node in T
    - We know the flow f never changes
    - How does the value of the new cut C' = (S', T') change?
    - It doesn't! Why? See next slide…

Proof: Flow-value lemma cont.

- Let ***f*** be any flow and C = (S, T) be any cut
  - The net flow across (S, T) equals the value of the flow ***f***

- Why does C’ = (S’, T’) have the same net flow?
  - Local equilibrium: net flow coming into node p from nodes in S only must equal the flow going out across the cut to nodes in T
  - After node p is moved:
    - everything going across the cut now goes to something in T from T, everything going to or from a node in S now goes across the cut.
  - Thus, by local equilibrium the value of the cut C’ is equivalent to the value of the cut C
  - Induction done!

Max-flow Min-cut Theorem

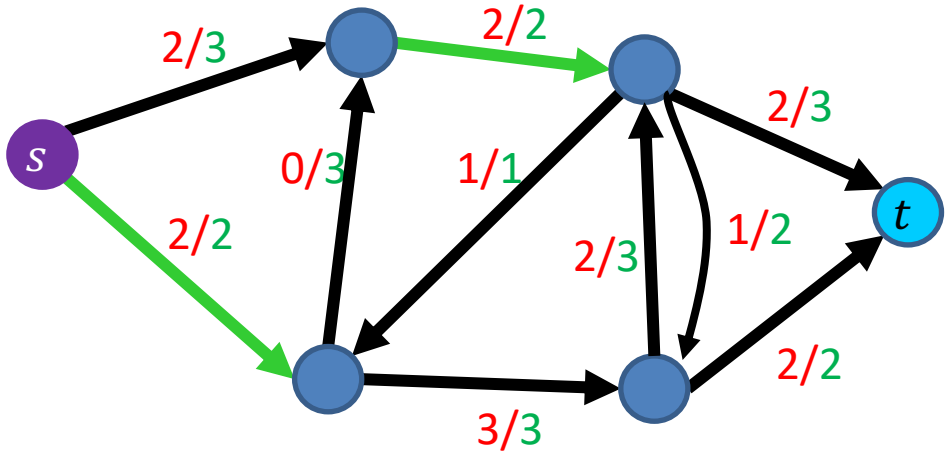
- Reminder about *weak duality*:
  - Let ***f*** be any flow and C = (S, T) be any cut
  - Then the value of ***f*** <= capacity of C
- The max flow ***f*** <= capacity of any cut C
  - So the best ***f*** can be is the capacity of the smallest-capacity cut
  - Can we guarantee that a flow with that value exists? (Yes, in later slides.)

- **The Max-flow Min-cut theorem:** the maximum value of an s-t flow is equal to the minimum capacity of an s-t cut

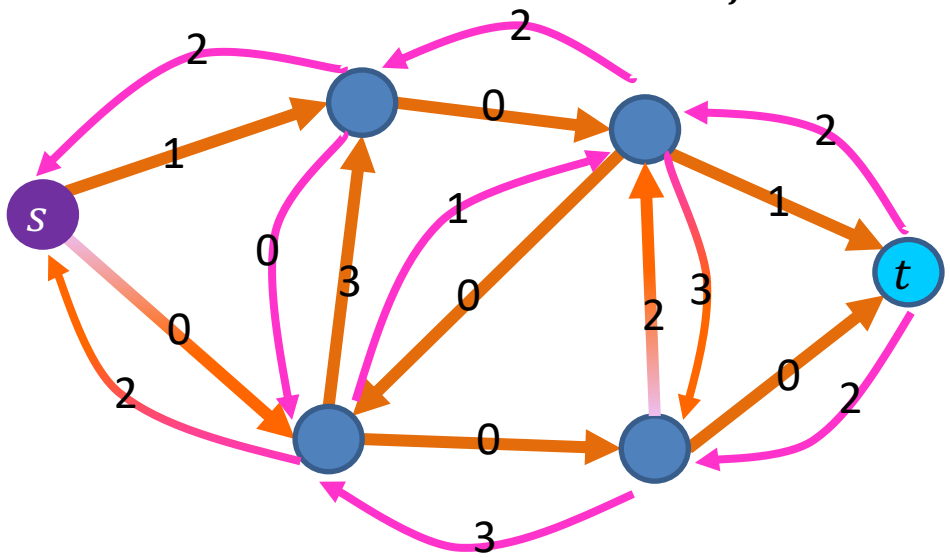
- In other words, if you look at all the possible cuts in the graph, and find the smallest capacity of those cuts, then that value is the value of the maximum flow for that network.

Example: Max-flow/Min-cut

Flow Graph G



Residual Graph G_f



$|f| = 4$

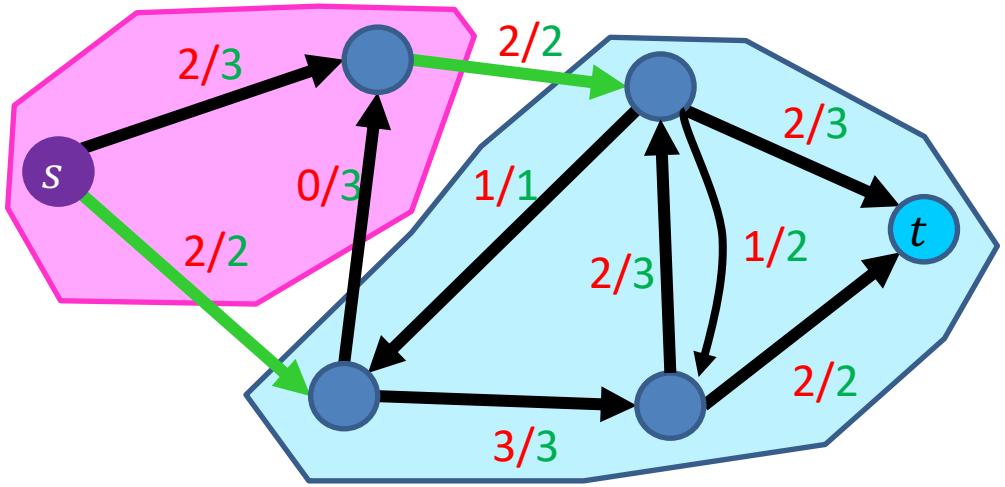
Min Capacity cut $= 4$

Right? Look at other cuts in G.

No Augmenting Paths

Max-flow Min-cut and Ford-Fulkerson

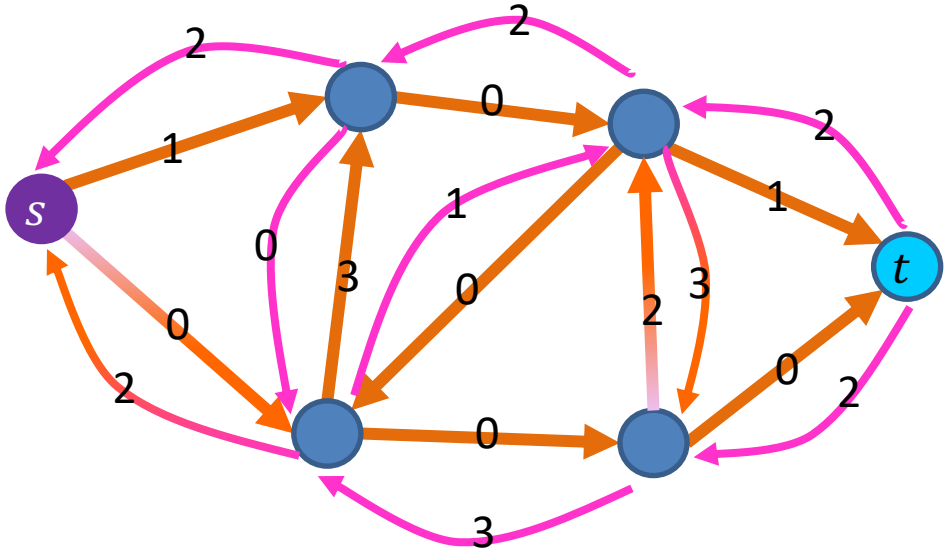
Flow Graph G



$|f| = 4$

Min Capacity cut $= 4$

Residual Graph G_f



No Augmenting Paths

Idea: When there are no more augmenting paths, there exists a cut in the graph with cost matching the flow (We're going to use the flow-value lemma here.)

Ford-Fulkerson Proof: Outline

- To show Ford-Fulkerson is correct we will prove:
  - When there are no more augmenting paths in G_f there is a cut in G with capacity equal to the flow f found by Ford-Fulkerson
  - It's not possible for any other flow to have larger value than this f

- This lets us conclude: the maximum flow through a network matches the minimum-capacity cut
- Duality
  - When we've maximized max flow, we've minimized min cut (and vice-versa), so we can check when we've found one by finding the other

Ford-Fulkerson Proof

- We'll show the following three statements are equivalent, and together this shows Ford-Fulkerson is correct.
- For some flow f

  A) There is no augmenting path in G_f with respect to f

  B) There exists a cut in G whose capacity equals the value of f

  C) The flow f is a maximum flow in G

- Let's prove this equivalence by proving the following three implications:

  A → B

  B → C

  C → A

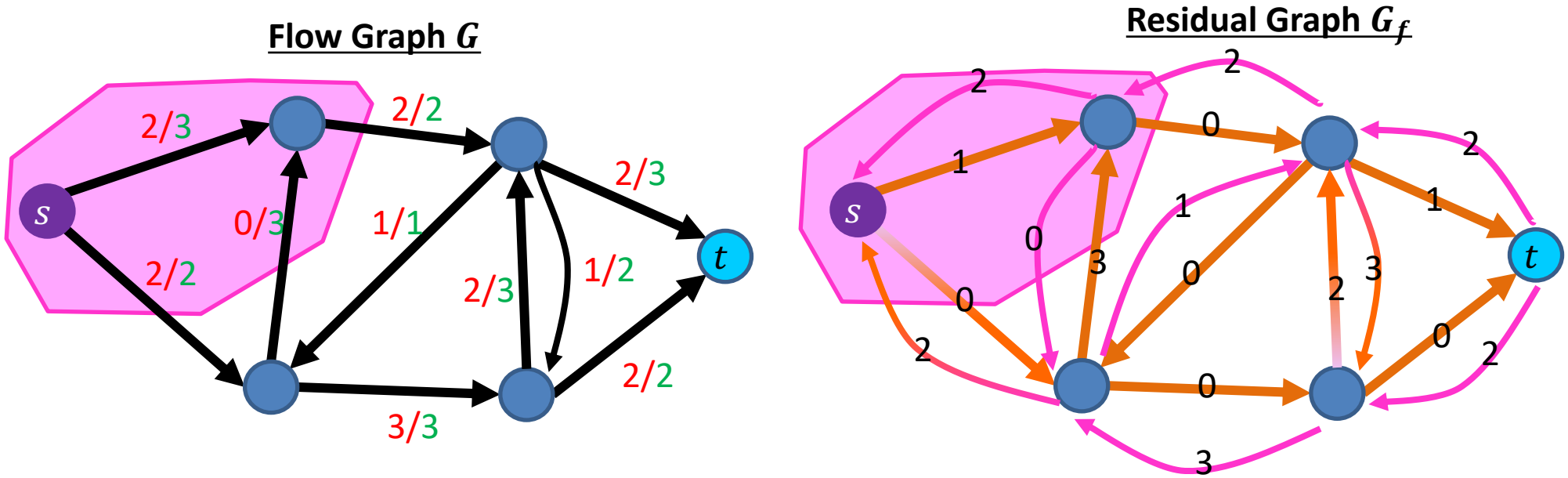
A → B: if no augmenting path, cut with capacity f

If there is no augmenting path in G_f with respect to f , then there exists a cut in G whose capacity equals the value of f

- Define S = nodes reachable from source node s by positive-weight edges in the residual graph

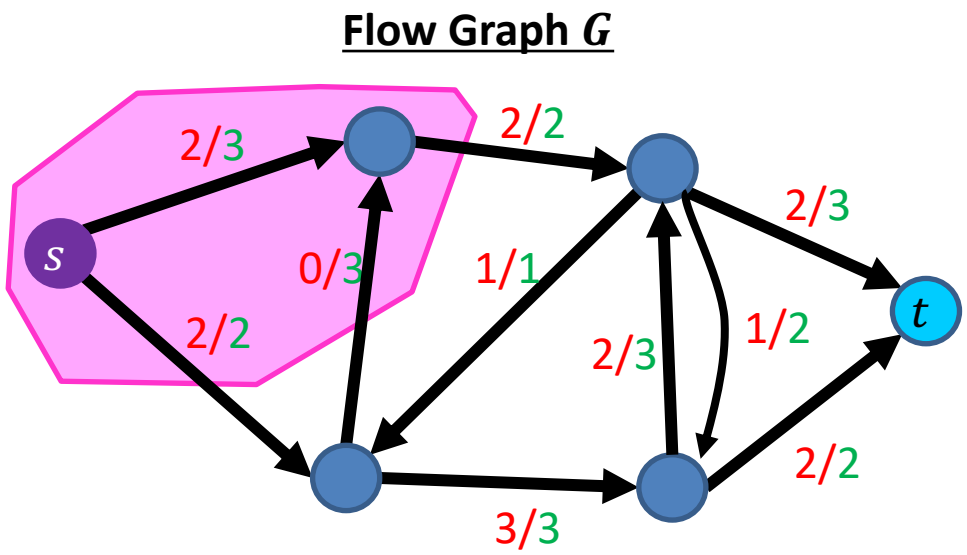
  $T = V - S$ and S separates s , t (otherwise there's an augmenting path)

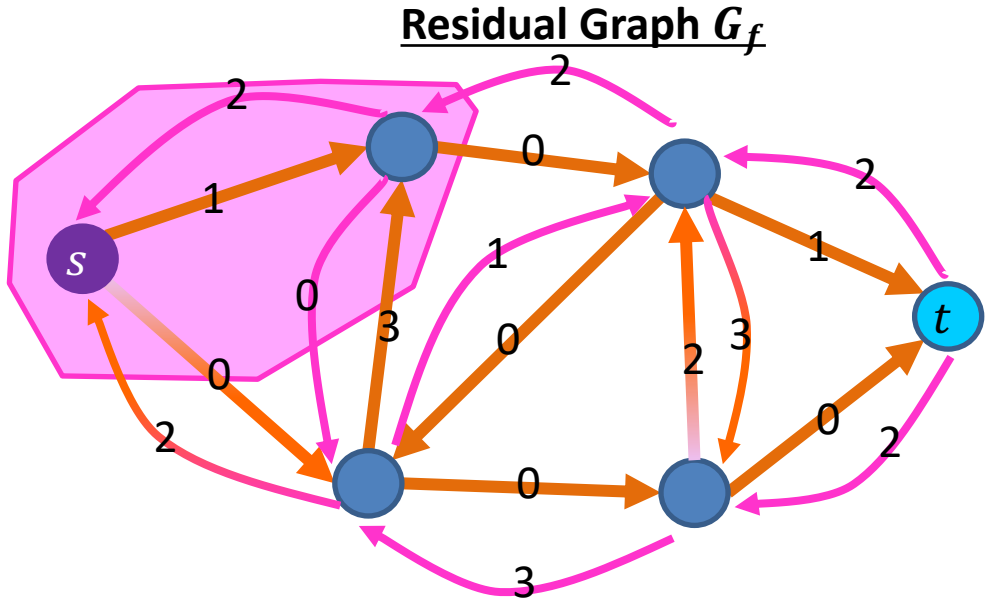
- Let *C = (S, T)* be the cut that separates *S* and *T*



A → B: if no augmenting path, cut with capacity f

- Let *C = (S, T)* be the cut that separates *S* and *T*
- We're trying to show capacity of C is f
- We can see that Net-Flow(C) = Capacity(C). Why? Because…
  - Each of the "cut edges" in *G* have forward-flow in G_f of 0 and back-flow in G_f equal to edge's capacity.
  - This means flow in G for each of these edges is max-capacity





A → B: if no augmenting path, cut with capacity f

- Let *C = (S, T)* be the cut that separates *S* and *T*
- We're trying to show capacity of C is f
- We know **Net-Flow(C) = Capacity(C)**. Why? Because…
  - Each of the "cut edges" in *G* have forward-flow in G_f of 0 and back-flow in G_f equal to edge's capacity.
  - This means flow in G for each of these edges is max-capacity
- Flow-value Lemma: The net flow across C equals the value of the flow ***f***
- Therefore: The capacity across C equals the value of the flow ***f***
- We've thus proven:
  If there is no augmenting path in G_f with respect to f , then there exists a cut in G whose capacity equals the value of f
- Tell me: how could we identify this cut *C = (S, T)*?

B → C: if cut with capacity f, f is max-flow

If there exists a cut in G whose capacity equals the value of f , then the flow f is a maximum flow in G

Proof by contradiction

- Let *f'* be a flow larger than *f* (the capacity of the cut *C* found by Ford-Fulkerson)
- Flow-value lemma tells us: *f' = Net-Flow(C)*
- Given*: f = Capacity(C)*
- If *f' > f* , then *Net-Flow(C) > Capacity(C)*
- **Contradicts** weak-duality property:
  Value of any flow through C <= Capacity(C)

These Prove Correctness of Ford-Fulkerson

- We said: To show Ford-Fulkerson is correct we will prove:
  - When there are no more augmenting paths in G_f there is a cut in G with capacity equal to the flow f found by Ford-Fulkerson
  - It's not possible for any other flow to have larger value than this f

A → B: *If there is no augmenting path in G_f with respect to f , then there exists a cut in G whose capacity equals the value of f*

B → C: *If there exists a cut in G whose capacity equals the value of f , then the flow f is a maximum flow in G*

C → A: if f is max-flow, then no augmenting path

Just to go "full circle", let's show this too!

If the flow f is a maximum flow in G, then there is no augmenting path in G_f with respect to f

Proof by contradiction, using weak duality property

- Assume *f* is max-flow but there exists an augmenting path *p* in G_f
- Path *p* can add positive-value flow f_p to the overall flow through G
- So overall flow of $f + f_p$ is possible in G
- **Contradiction:** we said *f* is maximum flow that's possible

Other Max-flow algorithms

- **Ford-Fulkerson**
  - $\boldsymbol{\Theta(E|f|)}$
- **Edmonds-Karp**
  - $\boldsymbol{\Theta(E^2V)}$
- Push-Relabel (Tarjan)
  - $\Theta(EV^2)$
- Faster Push-Relabel (also Tarjan)
  - $\Theta(V^3)$