

# Network flows

Introduction to Network Science

Carlos Castillo

Topic 20

# Sources

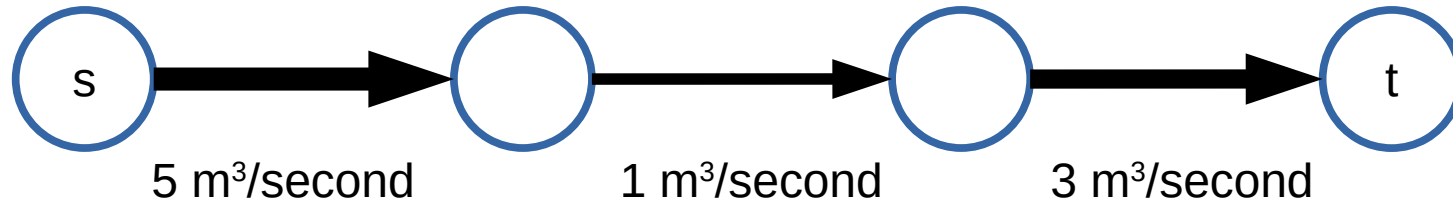
- Barabási 2016 Chapter 9
- Networks, Crowds, and Markets Ch 3
- C. Castillo: Graph partitioning 2017

# Splitting into two communities:

## Max-flow and Min-cut

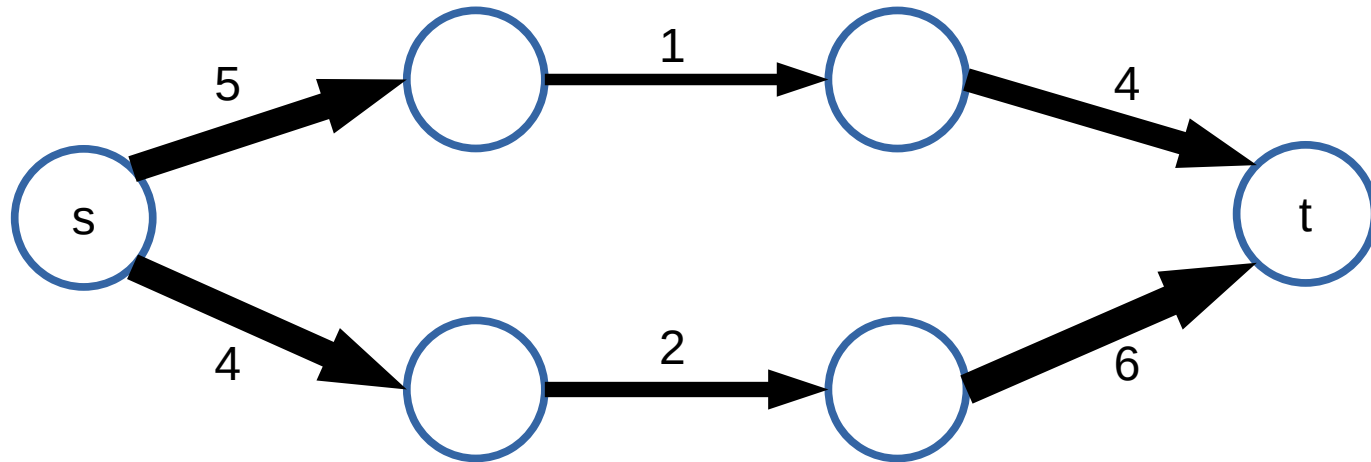
# Maximum flow: example 1

- If edge weights were capacities, what is the **maximum flow** that can be sent from s to t?



# Maximum flow: example 2

- If edge weights were capacities, what is the **maximum flow** that can be sent from s to t?



# Maximum flow problem

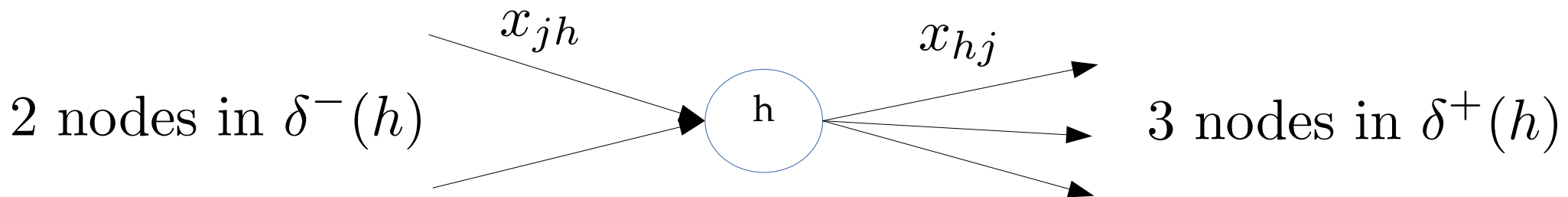
- What is the maximum “flow” that can be carried from  $s$  to  $t$ ?
  - Think of edge weights as capacities (e.g.  $\text{m}^3/\text{s}$  of water)
- What is the flow of an edge?
  - The amount sent through that edge (an assignment)
- What is the net flow of a node?
  - The amount exiting the node minus the amount entering the node

# Formulating the max flow problem

- The flow through each edge should be  $\leq k_{ij}$
- Net flow at node  $h$ :  
$$flow(h) = out\_flow(h) - in\_flow(h)$$
- Node  $s$  has only *out\_flow*, should have positive flow  $v$
- Node  $t$  has only *in\_flow*, should have negative flow  $-v$
- *What should be the flow of the other nodes?*

# Formulating the max flow problem

- Let  $v$  be a feasible flow
- Node  $s$  should have positive flow  $v$
- Node  $t$  should have negative flow  $-v$



- *What should be the flow of an arbitrary node  $h$ ?*

$$\sum_{(h,j) \in \delta^+(h)} x_{hj} - \sum_{(i,h) \in \delta^-(h)} x_{ih} = ?$$



# Max flow as a linear program

N: set of nodes, A: set of edges

$$\max \quad v \quad (1)$$

$$\sum_{(s,j) \in \delta^+(s)} x_{sj} = v \quad (2)$$

$$- \sum_{(i,t) \in \delta^-(t)} x_{it} = -v \quad (3)$$

$$\sum_{(h,j) \in \delta^+(h)} x_{hj} - \sum_{(i,h) \in \delta^-(h)} x_{ih} = 0, \quad h \in N - \{s, t\} \quad (4)$$

$$x_{ij} \leq k_{ij} \quad (i, j) \in A \quad (5)$$

$$x_{ij} \geq 0 \quad (i, j) \in A \quad (6)$$

# Primal-Dual in Linear Programming

## PRIMAL

$$\begin{aligned} & \min \sum_j c_j x_j \quad \text{subject to} \\ & \sum_j a_{ij} x_j \geq b_i \quad \forall i \in [m] \\ & x_j \geq 0 \quad \forall j \in [n] \end{aligned}$$

## DUAL

$$\begin{aligned} & \max \sum_i y_i b_i \quad \text{subject to} \\ & \sum_i y_i a_{ij} \leq c_j \quad \forall j \in [n] \\ & y_i \geq 0 \quad \forall i \in [m] \end{aligned}$$

# Writing the dual: each constraint will become a variable

$$\max \quad v \tag{1}$$

$$\sum_{(s,j) \in \delta^+(s)} x_{sj} = v \quad \text{variable } u_s \tag{2}$$

$$- \sum_{(i,t) \in \delta^-(t)} x_{it} = -v \quad \text{variable } u_t \tag{3}$$

$$\sum_{(h,j) \in \delta^+(h)} x_{hj} - \sum_{(i,h) \in \delta^-(h)} x_{ih} = 0, \quad h \in N - \{s, t\} \quad \text{variables } u_j \tag{4}$$

$$x_{ij} \leq k_{ij} \quad (i, j) \in A \quad \text{variables } y_{ij} \tag{5}$$

$$x_{ij} \geq 0 \quad (i, j) \in A \tag{6}$$

# Writing the dual

- Remember: the infimum of the solutions of the dual is the supremum of the solutions of primal

$$\min \sum_{(i,j) \in A} k_{ij} y_{ij}$$

$$u_i - u_j + y_{ij} \geq 0, (i, j) \in A$$

$$-u_s + u_t = 1$$

$$y_{ij} \geq 0$$

(Think of  $y_{ij}$  as  
0 or 1)

- Variables  $u_i$  don't enter the objective, only their difference is in the constraints
- We can set them arbitrarily, in particular  $u_s = 0, u_t = 1$

# Dual (after simplification)

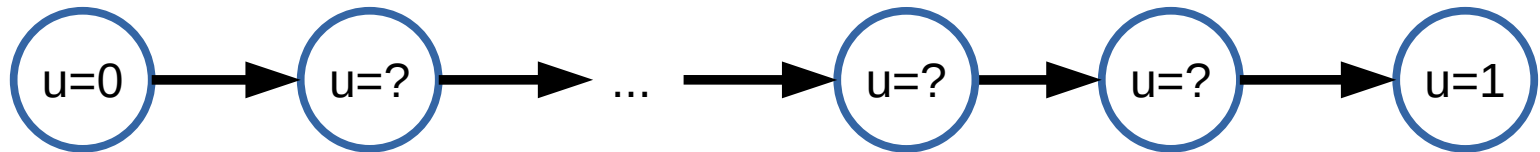
$$\min \sum_{(i,j) \in A} k_{ij} y_{ij}$$

$$u_i - u_j + y_{ij} \geq 0, (i, j) \in A$$

$$y_{ij} \geq 0$$

$$u_s = 0, u_t = 1$$

- What happens with the values of  $u$  in every simple path going from  $s$  to  $t$ ?



# Dual (after simplification)

$$\min \sum_{(i,j) \in A} k_{ij} y_{ij}$$

$$u_i - u_j + y_{ij} \geq 0, (i, j) \in A$$

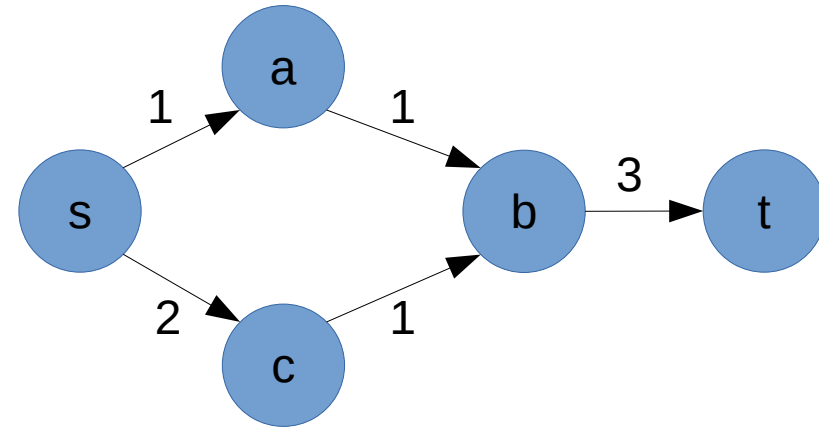
$$y_{ij} \geq 0$$

$$u_s = 0, u_t = 1$$

**Every feasible solution represents a cut (S, S')**

Upload to Nearpod Collaborate  
<https://nearpod.com/student/>  
 Code to be given during class

# Exercise



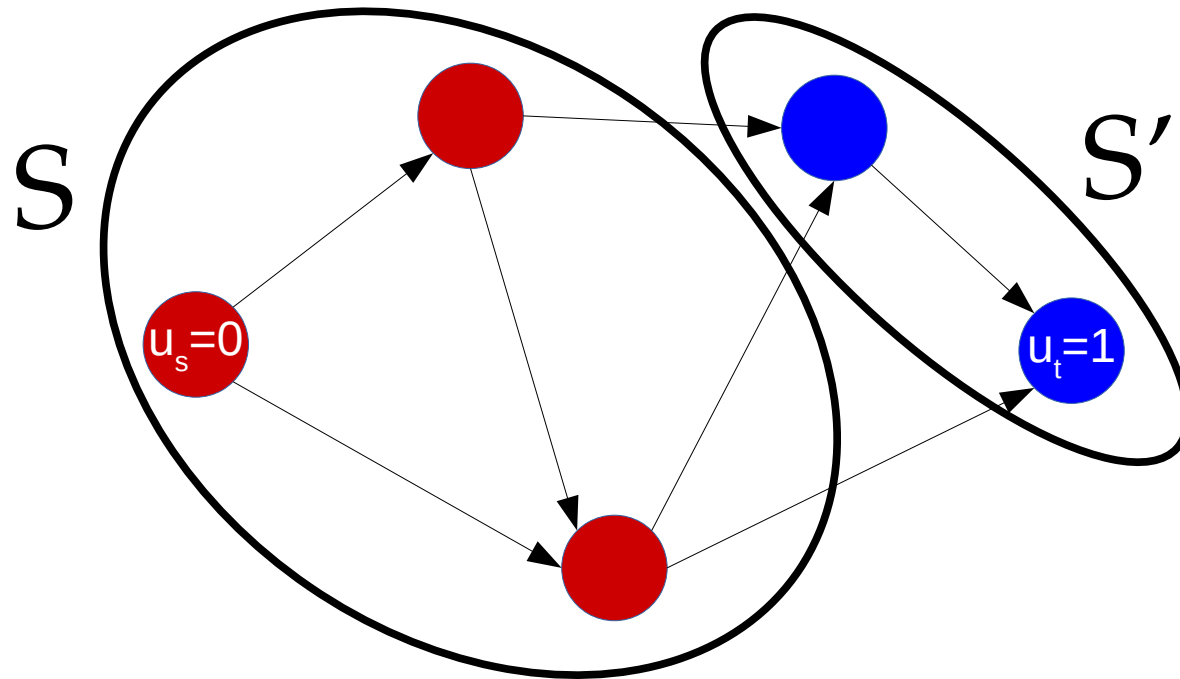
- Write the primal equations for this graph
  - Unknowns:  $x_{sa}, x_{sc}, x_{ab}, x_{cb}, x_{bt}, v$
- Write the dual equations for this graph
  - Unknowns:  $u_a, u_b, u_c, y_{sa}, y_{sc}, y_{ab}, y_{cb}, y_{bt}$
- Guess both solutions, check that you satisfy all constraints

$$\begin{aligned}
 \max \quad & v \\
 \sum_{(s,j) \in \delta^+(s)} x_{sj} &= v \\
 - \sum_{(i,t) \in \delta^-(t)} x_{it} &= -v \\
 \sum_{(h,j) \in \delta^+(h)} x_{hj} - \sum_{(i,h) \in \delta^-(h)} x_{ih} &= 0, \quad h \in N - \{s, t\} \\
 x_{ij} &\leq k_{ij} \quad (i, j) \in A \\
 x_{ij} &\geq 0 \quad (i, j) \in A
 \end{aligned}$$

$$\begin{aligned}
 \min \quad & \sum_{(i,j) \in A} k_{ij} y_{ij} \\
 u_i - u_j + y_{ij} &\geq 0, \quad (i, j) \in A \\
 y_{ij} &\geq 0 \\
 u_s &= 0, u_t = 1
 \end{aligned}$$

# Dual solutions are cuts

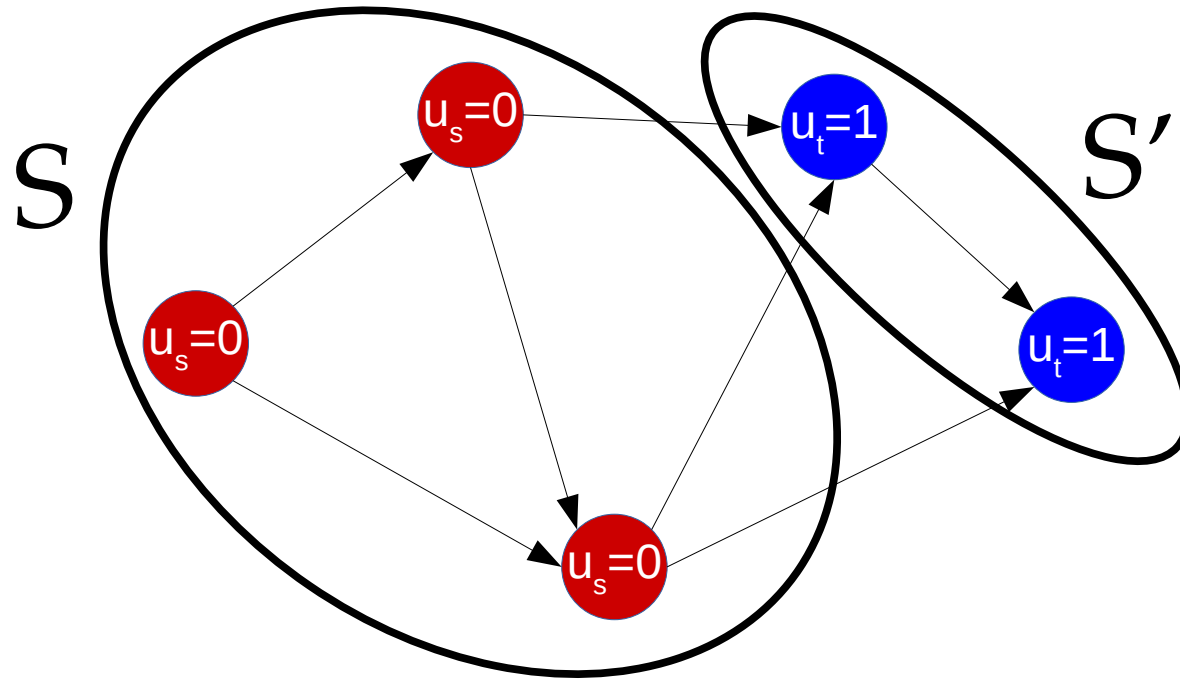
- Every feasible solution of the dual has the form of a cut  $(S, S')$





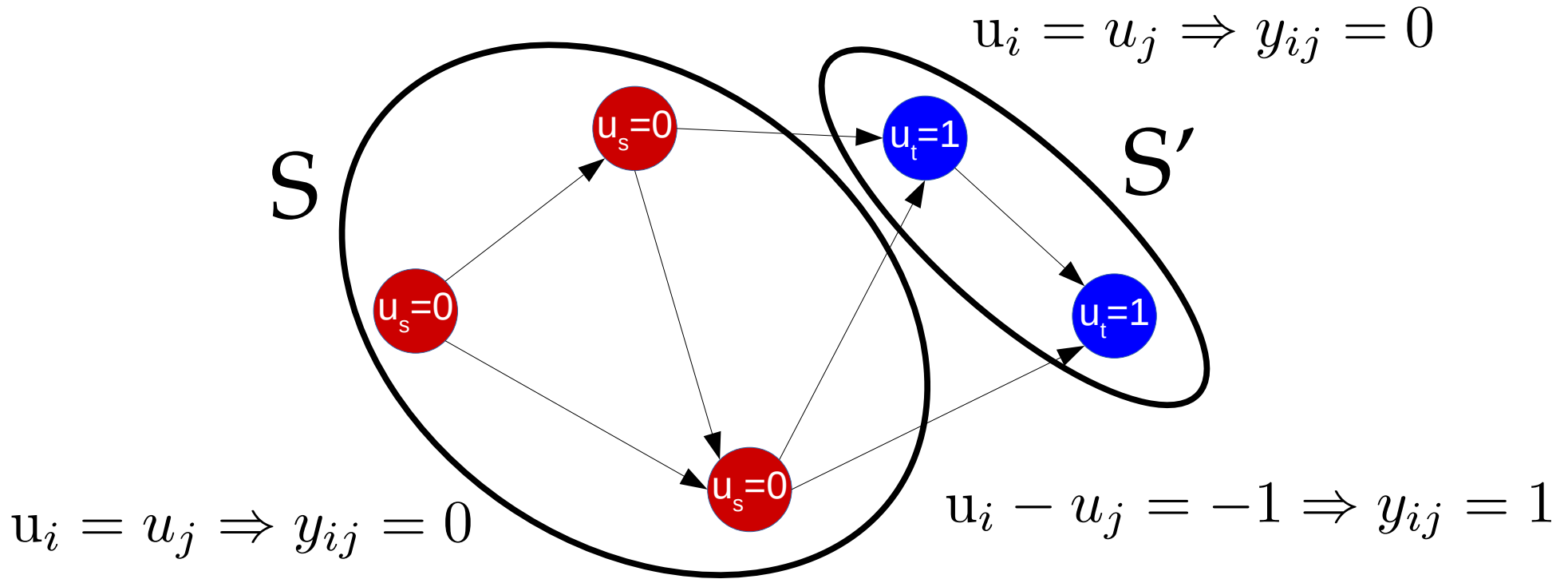
# Dual solutions are cuts

- Every feasible solution of the dual has the form of a cut  $(S, S')$



# Dual solutions are (s-t)-cuts

$u_i - u_j + y_{ij} \geq 0$  and remember we're trying to minimize  $\sum k_{ij} y_{ij}$



# One more thing about the solution

$$\min \sum_{(i,j) \in A} k_{ij} y_{ij}$$

$$u_i - u_j + y_{ij} \geq 0, (i, j) \in A$$

$$y_{ij} \geq 0$$

$$u_s = 1, u_t = 0$$

$y_{ij}$  is a dual variable corresponding to primal constraint  $x_{ij} \leq k_{ij}$

If  $y_{ij}$  is non-zero, then the corresponding constraint is tight

What does it mean for the edges in the cut?

# This is an efficient method

- Min-cut and Max-flow are equivalent problems
  - Their solutions are also equal: the value of the maximum flow is equivalent to the minimum cut
- Think of a chain that breaks at the weakest link
- Both can be solved exactly in polynomial time

# Two additional videos

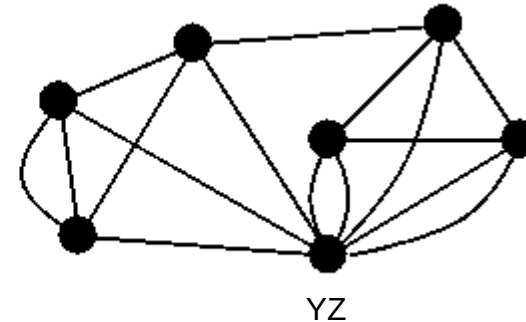
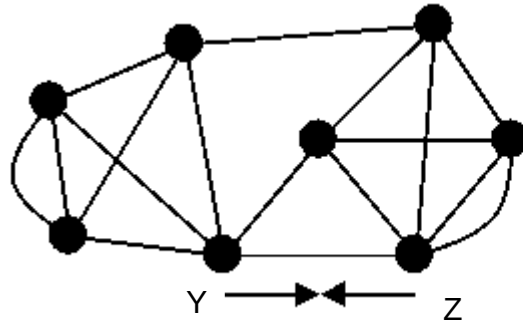
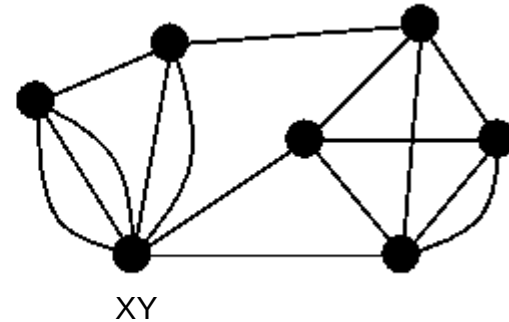
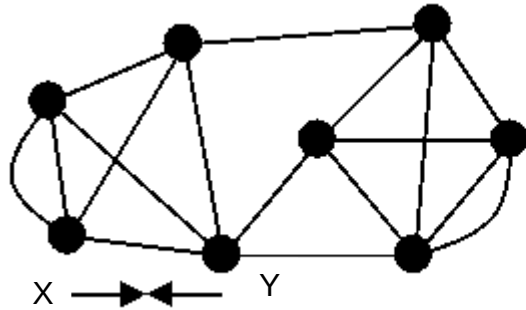
- The min-cut, max-flow duality (2 minutes)
  - <https://www.youtube.com/watch?v=RwhEF-LXzaE>
- Numerical example (10 minutes)
  - <https://www.youtube.com/watch?v=u6FkNw16VJA>

# Randomized algorithm for (s-t)-cuts

# Randomized algorithm for (s-t)-cuts

- Pick an edge at random  $(u, v)$
- Merge  $u$  and  $v$  in new vertex  $uv$
- Edges between  $u$  and  $v$  are removed
- Edges pointing to  $u$  or  $v$  are added as multi-edges to vertex  $uv$
- When only  $s$  and  $t$  remain, the multi-edges are a cut, probably the minimum one

# Example merges (“contractions”)

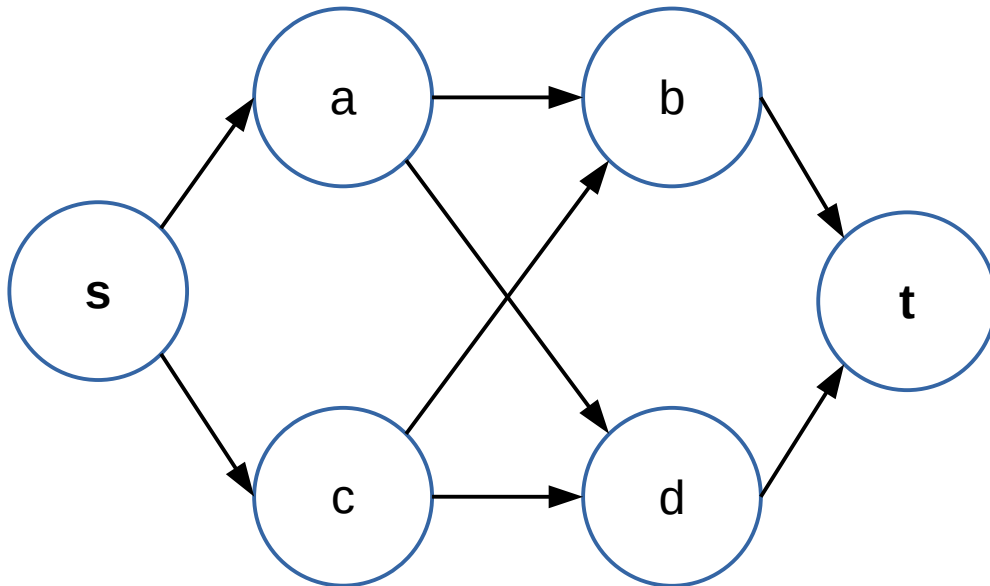




# Exercise

*Run the randomized algorithm on this graph*

- Pick an edge at random  $(u, v)$
- Merge  $u$  and  $v$  in new vertex  $uv$
- Edges between  $u$  and  $v$  are removed
- Edges pointing to  $u$  or  $v$  are added as multi-edges to vertex  $uv$
- When only  $s$  and  $t$  remain, the multi-edges are a cut, probably the minimum one



Upload to Nearpod Collaborate  
<https://nearpod.com/student/>  
Code to be given during class

# The randomized algorithm might miss the min cut

- Multiple runs are required
- The probability that this finds the min cut in one run is about  $1/\log(n)$ , so  $O(\log n)$  iterations are required to find min cut
- Each iteration costs  $O(n^2 \log n)$
- $O(n^2 \log^2 n)$  operations needed to find min cut
- Exact algorithm:  $O(n^3 + n^2 \log n)$ ; the  $n^3$  is because of  $|V||E|$  operations required

# Summary

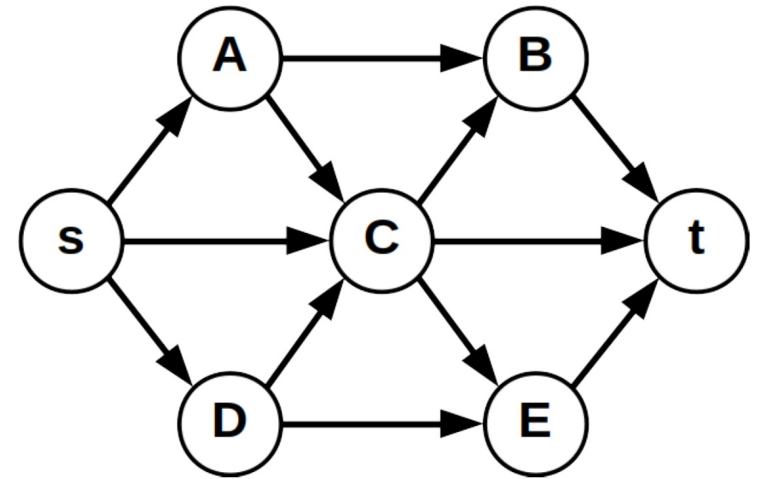
# Things remember

- Minimum s-t cut in a graph = set of edges
- The sum of the capacities of those edges is the maximum s-t flow the graph can carry
- Solvable in polynomial time
- Approximate randomized algorithm exists

# Practice on your own

Consider  $(s, t)$ -cuts on the graph on the right, where  $s$  is the source node and  $t$  is the terminal node. Assume every edge has cost equal to 1.

1. By visual inspection, what is the minimum cost of an  $(s, t)$ -cut in this graph, and what is an example of a cut having that cost?
2. Run the algorithm for randomized  $(s, t)$ -cuts we saw in class, drawing all intermediate graphs, and indicate the cost of the resulting cut.



# Practice on your own (cont.)

- Create a graph
- Write the min-flow, max-cut equations
- Find an optimal solution
- Run the randomized s-t cut algorithm