

# **CSE 421**

## **Hall's Condition, Edge Disjoint Paths, Menger's Theorem, ...**

Shayan Oveis Gharan

# Perfect Bipartite Matching

# Perfect Bipartite Matching

**Def.** A matching  $M \subseteq E$  is **perfect** if each node appears in exactly one edge in  $M$ .

**Q.** When does a bipartite graph have a perfect matching?

Structure of bipartite graphs with perfect matchings:

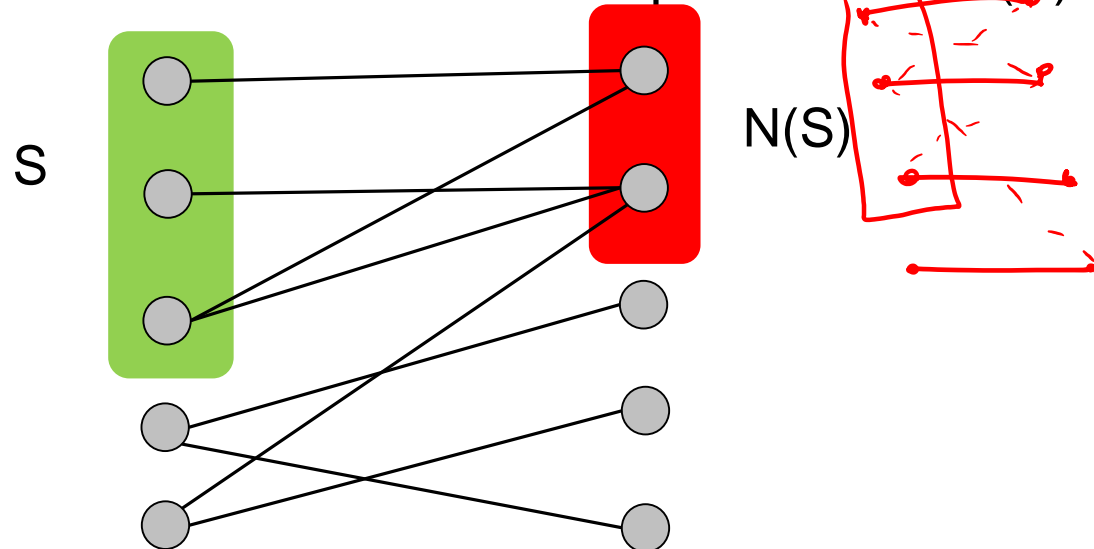
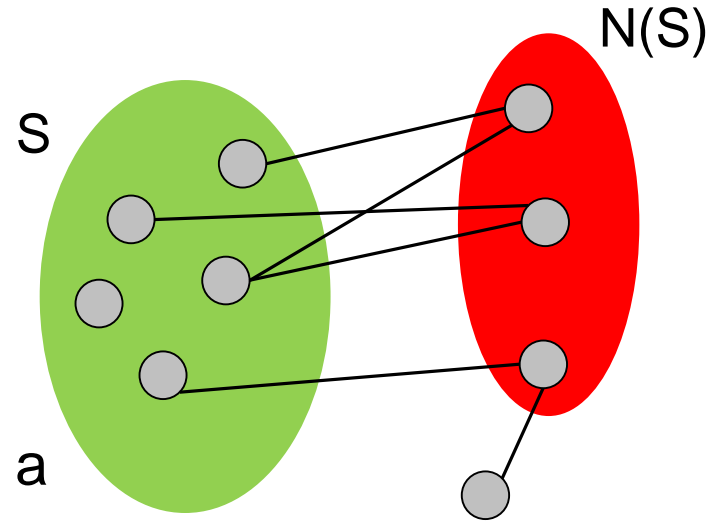
- Clearly we must have  $|X| = |Y|$ .
- What other conditions are necessary?
- What conditions are sufficient?

# Perfect Bipartite Matching: $N(S)$

**Def.** Let  $S$  be a subset of nodes, and let  $N(S)$  be the set of nodes adjacent to nodes in  $S$ .

**Observation.** If a bipartite graph  $G$  has a perfect matching, then  $|N(S)| \geq |S|$  for all subsets  $S \subseteq X$ .

**Pf.** Each  $v \in S$  has to be matched to a unique node in  $N(S)$ .



# Marriage Theorem

Thm: [Frobenius 1917, Hall 1935] Let  $G = (X \cup Y, E)$  be a bipartite graph with  $|X| = |Y|$ .

Then,  $G$  has a perfect matching iff  $|N(S)| \geq |S|$  for all subsets  $S \subseteq X$ .

Pf.  $\Rightarrow$

This was the previous observation.

If  $|N(S)| < |S|$  for some  $S$ , then there is no perfect matching.

# Marriage Theorem

**Pf.**  $\exists S \subseteq X$  s.t.,  $|N(S)| < |S| \iff G$  does not have a perfect matching

Formulate as a max-flow and let  $(A, B)$  be the min s-t cut

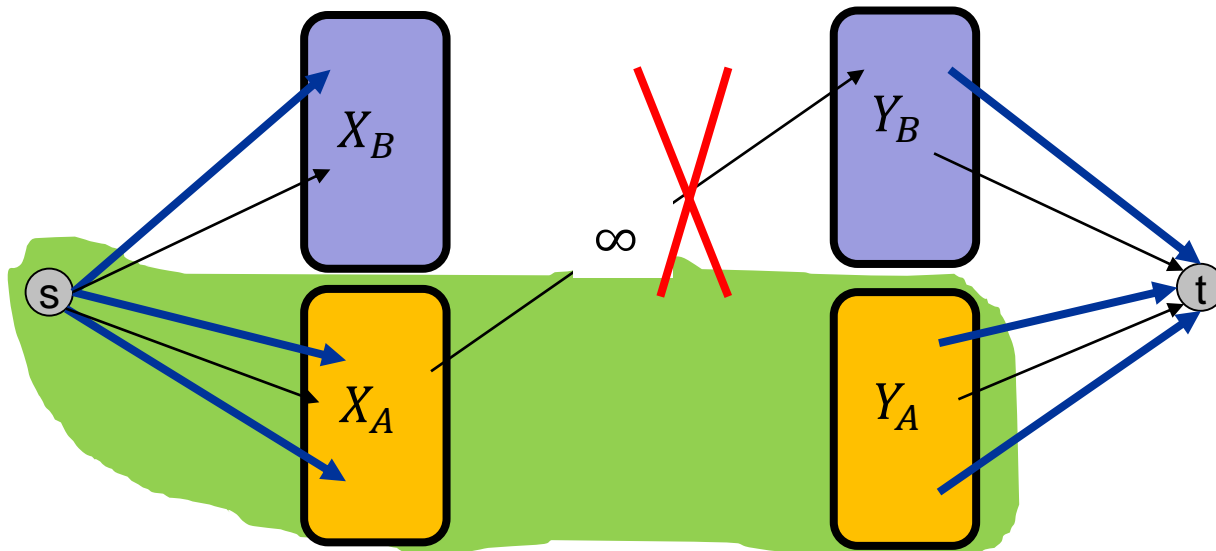
$G$  has no perfect matching  $\Rightarrow v(f^*) < |X|$ . So,  $cap(A, B) < |X|$

Define  $X_A = X \cap A, X_B = X \cap B, Y_A = Y \cap A$

Then,  $cap(A, B) = |X_B| + |Y_A|$

Since min-cut does not use  $\infty$  edges,  $N(X_A) \subseteq Y_A$

$|N(X_A)| \leq |Y_A| = cap(A, B) - |X_B| = cap(A, B) - |X| + |X_A| < |X_A|$



# Bipartite Matching Running Time

Which max flow algorithm to use for bipartite matching?

Generic augmenting path:  $O(m \text{ val}(f^*)) = O(mn)$ .

Capacity scaling:  $O(m^2 \log C) = O(m^2)$ .

Shortest augmenting path:  $O(m n^{1/2})$ .

## Non-bipartite matching.

Structure of non-bipartite graphs is more complicated, but well-understood. [Tutte-Berge, Edmonds-Galai]

Blossom algorithm:  $O(n^4)$ . [Edmonds 1965]

Best known:  $O(m n^{1/2})$ . [Micali-Vazirani 1980]

# Edge Disjoint Paths

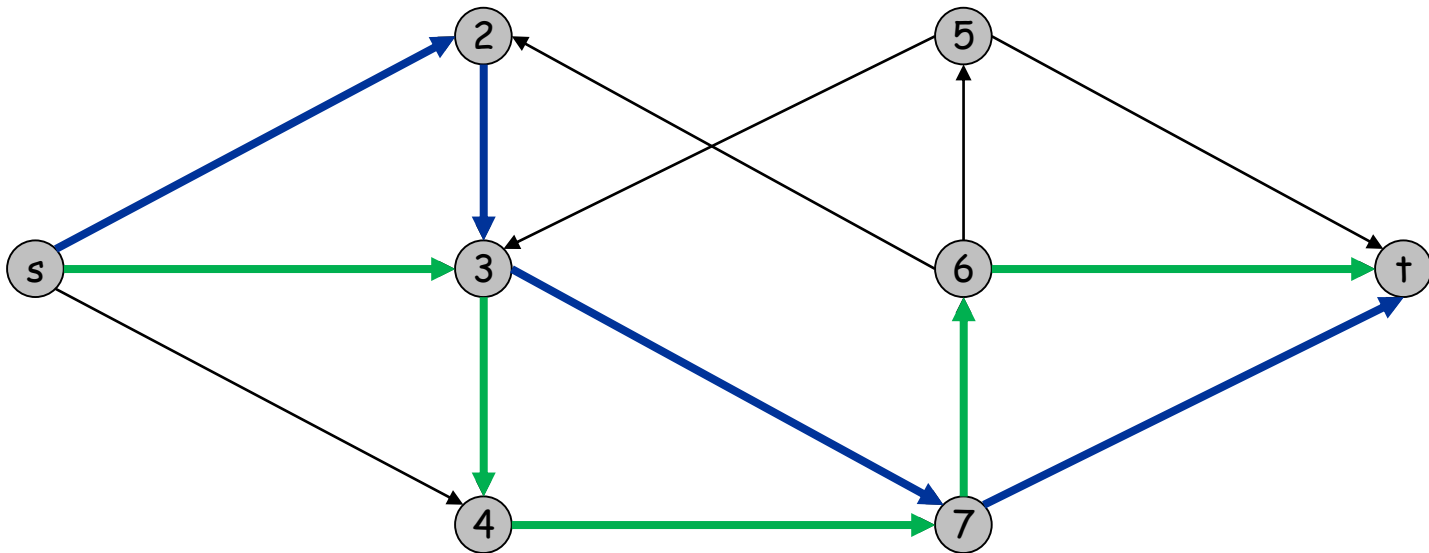


# Edge Disjoint Paths Problem

Given a digraph  $G = (V, E)$  and two nodes  $s$  and  $t$ , find the max number of edge-disjoint  $s$ - $t$  paths.

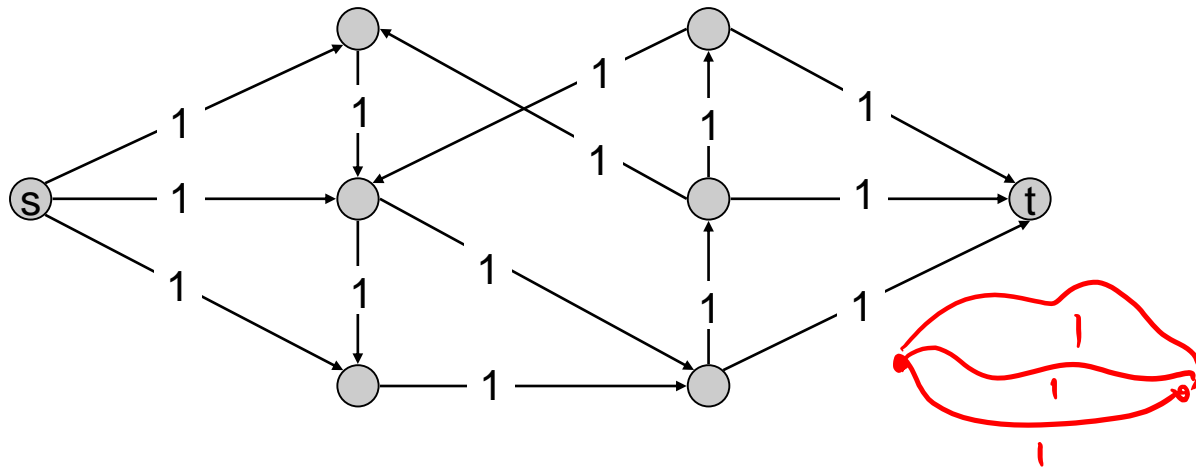
**Def.** Two paths are **edge-disjoint** if they have no edge in common.

**Ex:** communication networks.



# Max Flow Formulation

Assign a unit capacity to every edge. Find Max flow from  $s$  to  $t$ .



**Thm.** Max number edge-disjoint  $s$ - $t$  paths equals max flow value.

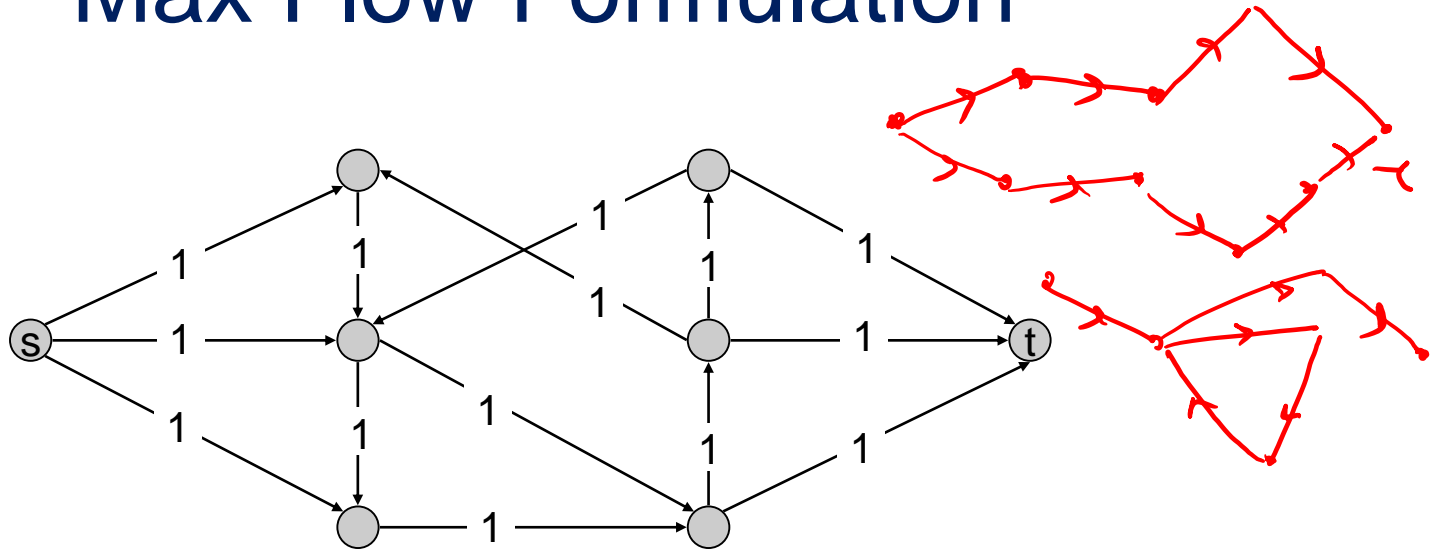
**Pf.**  $\leq$

Suppose there are  $k$  edge-disjoint paths  $P_1, \dots, P_k$ .

Set  $f(e) = 1$  if  $e$  participates in some path  $P_i$ ; else set  $f(e) = 0$ .

Since paths are edge-disjoint,  $f$  is a flow of value  $k$ . ■

# Max Flow Formulation



**Thm.** Max number edge-disjoint s-t paths equals max flow value.

**Pf.  $\geq$**  Suppose max flow value is  $k$

Integrality theorem  $\Rightarrow$  there exists 0-1 flow  $f$  of value  $k$ .

Consider edge  $(s, u)$  with  $f(s, u) = 1$ .

- by **conservation**, there exists an edge  $(u, v)$  with  $f(u, v) = 1$
- continue until reach  $t$ , always choosing a new edge

This produces  $k$  (not necessarily simple) edge-disjoint paths. ■

We can return to  $u$  so we can have cycles. But we can eliminate cycles if desired

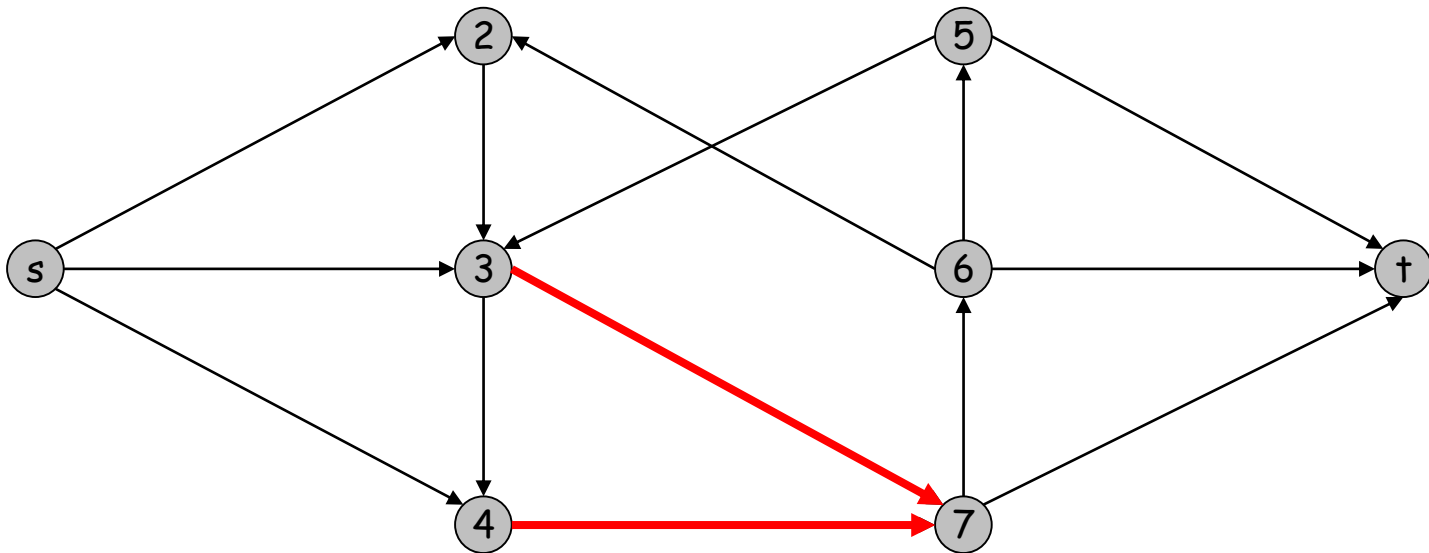
# Network Connectivity

# Network Connectivity

Given a digraph  $G = (V, E)$  and two nodes  $s$  and  $t$ , find min number of edges whose removal disconnects  $t$  from  $s$ .

**Def.** A set of edges  $F \subseteq E$  **disconnects  $t$  from  $s$**  if all  $s$ - $t$  paths uses at least one edge in  $F$ .

Ex: In testing network reliability



# Network Connectivity using Min Cut

Thm. [Menger 1927] The max number of edge-disjoint s-t paths is equal to the min number of edges whose removal disconnects t from s.

Pf.  $\leq$

Suppose the removal of  $F \subseteq E$  disconnects t from s, and  $|F| = k$ .

All s-t paths use at least one edge of F. Hence, the number of edge-disjoint paths is at most k.

# Network Connectivity using Min Cut

Thm. [Menger 1927] The max number of edge-disjoint s-t paths is equal to the min number of edges whose removal disconnects t from s.

Pf.  $\geq$

Suppose there are k edge disjoint paths from s to t

So, Max flow is k

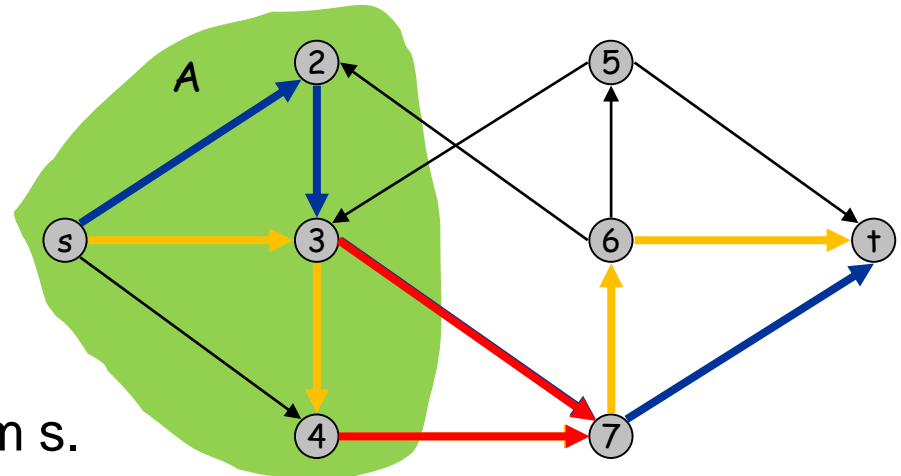
So, there is a s-t cut (A,B) s.t.,

$$\text{cap}(A,B)=k$$

Let F be the edges out of A. So,

$$|F|=k.$$

If we remove F we disconnect t from s.



# Image Segmentation



# Image Segmentation

Given an image we want to separate foreground from background

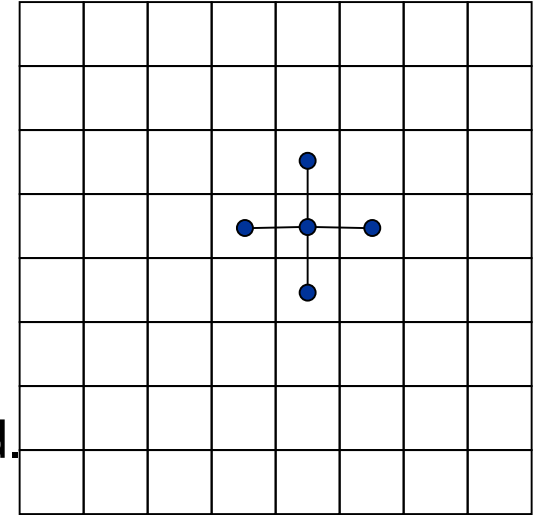
- Central problem in image processing.
- Divide image into coherent regions.



# Foreground / background segmentation

Label each pixel as foreground/background.

- $V$  = set of pixels,  $E$  = pairs of neighboring pixels.
- $a_i \geq 0$  is likelihood pixel  $i$  in foreground.
- $b_i \geq 0$  is likelihood pixel  $i$  in background.
- $p_{i,j} \geq 0$  is separation penalty for labeling one of  $i$  and  $j$  as foreground, and the other as background.



## Goals.

**Accuracy:** if  $a_i > b_i$  in isolation, prefer to label  $i$  in foreground.

**Smoothness:** if many neighbors of  $i$  are labeled foreground, we should be inclined to label  $i$  as foreground.

Find partition  $(A, B)$  that **maximizes:**

$$\sum_{i \in A} a_i + \sum_{j \in B} b_j - \sum_{\substack{(i,j) \in E \\ i \in A, j \in B}} p_{i,j}$$

↖
↖

Foreground
Background

# Image Seg: Min Cut Formulation

## Difficulties:

- Maximization (as opposed to minimization)
- No source or sink
- Undirected graph

## Step 1: Turn into Minimization

Maximizing 
$$\sum_{i \in A} a_i + \sum_{j \in B} b_j - \sum_{\substack{(i,j) \in E \\ i \in A, j \in B}} p_{i,j}$$

Equivalent to minimizing 
$$+ \sum_{i \in V} a_i + \sum_{j \in V} b_j - \sum_{i \in A} a_i - \sum_{j \in B} b_j + \sum_{\substack{(i,j) \in E \\ i \in A, j \in B}} p_{i,j}$$

Equivalent to minimizing 
$$+ \sum_{j \in B} a_j + \sum_{i \in A} b_i + \sum_{\substack{(i,j) \in E \\ i \in A, j \in B}} p_{i,j}$$

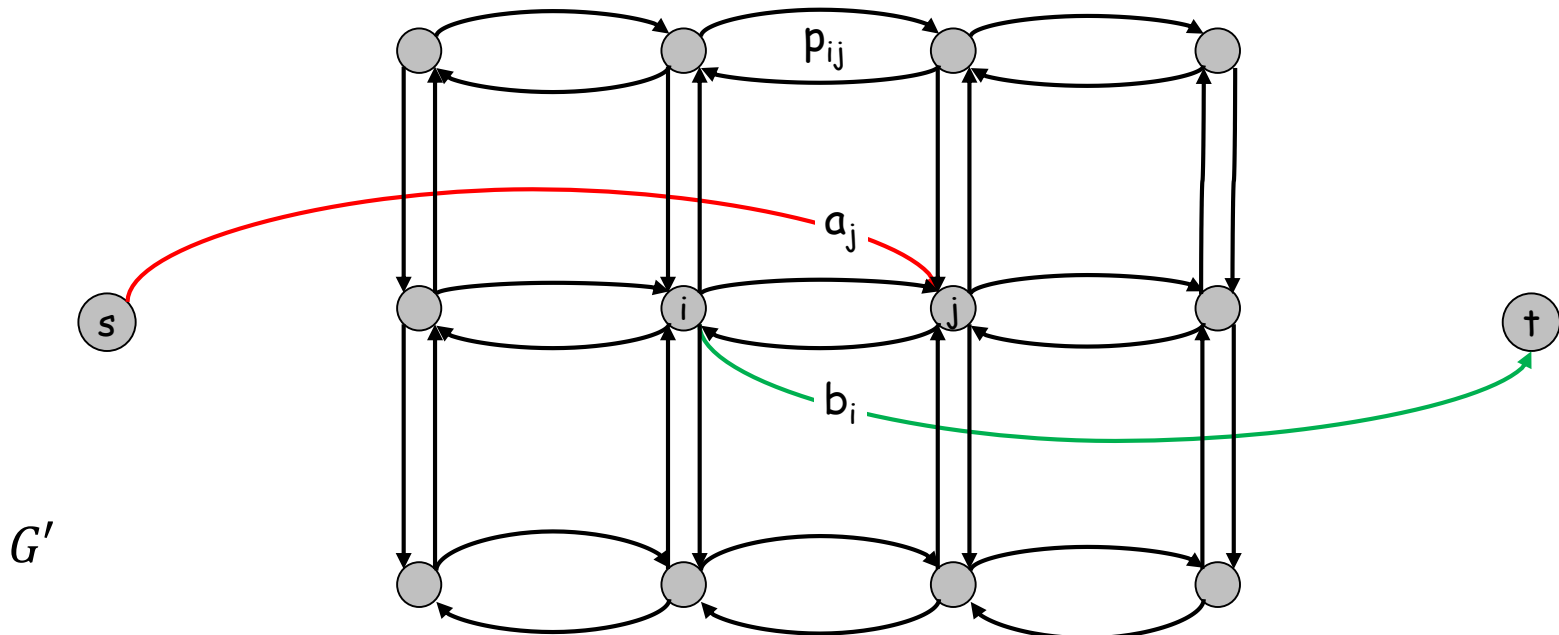
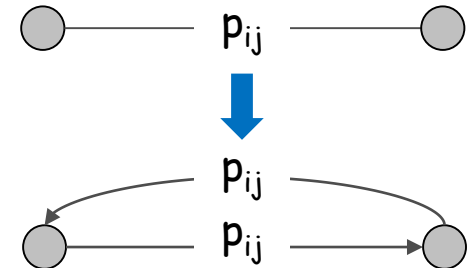
# Min cut Formulation (cont'd)

$G' = (V', E')$ .

Add  $s$  to correspond to foreground;

Add  $t$  to correspond to background

Use two anti-parallel edges  
instead of undirected edge.



# Min cut Formulation (cont'd)

Consider min cut  $(A, B)$  in  $G'$ . ( $A$  = foreground.)

$$cap(A, B) = \sum_{j \in B} a_j + \sum_{i \in A} b_i + \sum_{\substack{(i,j) \in E \\ i \in A, j \in B}} p_{i,j}$$

Precisely the quantity we want to minimize.

