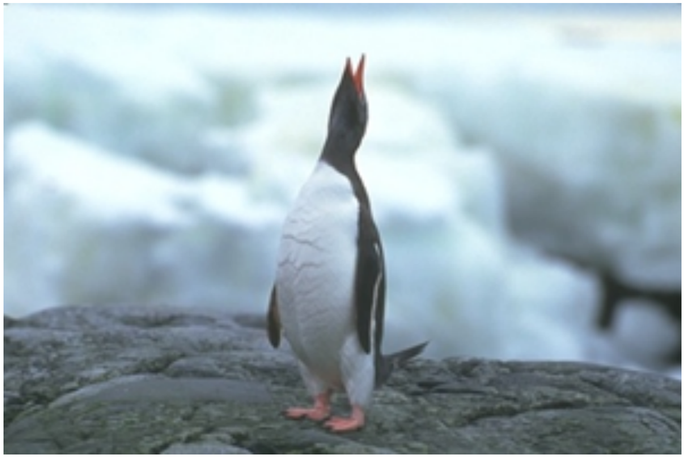
CS3000: Algorithms & Data
Jonathan Ullman

Lecture 21:
More Applications of Network Flow

Nov 30, 2018

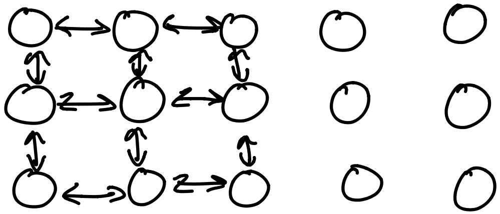
Image Segmentation

Image Segmentation



- Separate image into foreground and background
- We have some idea of:
  - whether pixel i is in the foreground or background
  - whether pair (i,j) are likely to go together

Image Segmentation



- Input:
  - a directed graph $G = (V, E)$; V = "pixels", E = "pairs"
  - likelihoods $a_i, b_i \geq 0$ for every $i \in V$
  - separation penalty $p_{ij} \geq 0$ for every $(i, j) \in E$
- Output:
  - a partition of V into (A, B) that maximizes

"foreground" "background"

$$q(A, B) = \sum_{i \in A} a_i + \sum_{j \in B} b_j - \sum_{\substack{(i,j) \in E \\ \text{from } A \text{ to } B}} p_{ij}$$

"quality"

Reduction to MinCut

- Differences between SEG and MINCUT:
  - SEG asks us to maximize, MINCUT asks us to minimize

$\max_{A,B} \quad q(A,B)$

$\min_{A,B} \quad -q(A,B)$

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

$$\Leftrightarrow$$

$$\min_{A,B} \sum_{i \in A} b_i + \sum_{j \in B} a_j + \sum_{\substack{(i,j) \in E \\ \text{from } A \text{ to } B}} p_{ij}$$

- SEG allows any partition, MINCUT requires $s \in A$, $t \in B$

mincut: graph has nodes $V \cup \{s,t\}$

$\min_{A,B \text{ partitioning } V} \quad cap(A \cup \{s\}, B \cup \{t\})$

- Add two nodes s,t

Reduction to MinCut

- How should the reduction work?
  - capacity of the cut should correspond to the function we're trying to minimize

$$\min_{A,B} \sum_{i \in A} b_i + \sum_{j \in B} a_j + \sum_{\substack{(i,j) \in E \\ \text{from } A \text{ to } B}} p_{ij}$$

$$\min_{A,B} \sum_{\substack{(i,j) \in E \\ \text{from } A \text{ to } B}} cap(i,j)$$

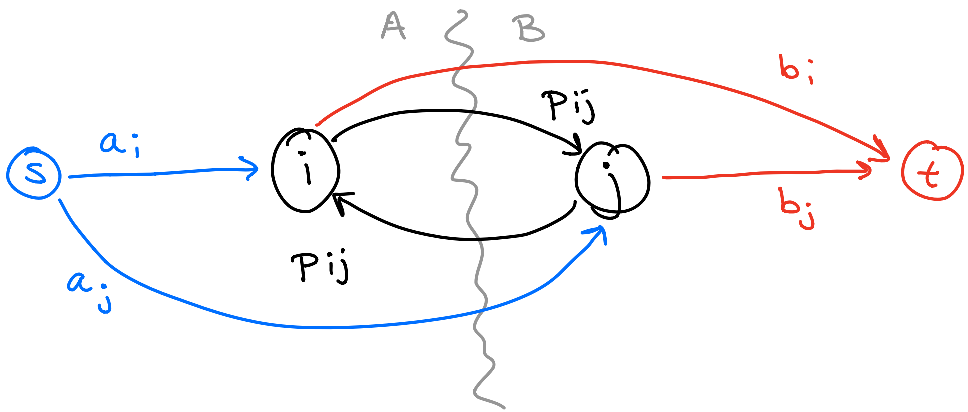
Reduction to MinCut

$\forall A, B \quad cap(A \cup \{s\}, B \cup \{t\})$
$= \sum_{i \in A} b_i + \sum_{j \in B} a_j + \sum_{\substack{(i,j) \in E \\ \text{from } A \text{ to } B}} p_{ij}$

- How should the reduction work?
  - capacity of the cut should correspond to the function we're trying to minimize

Assume G only has nodes $\{i, j\}$

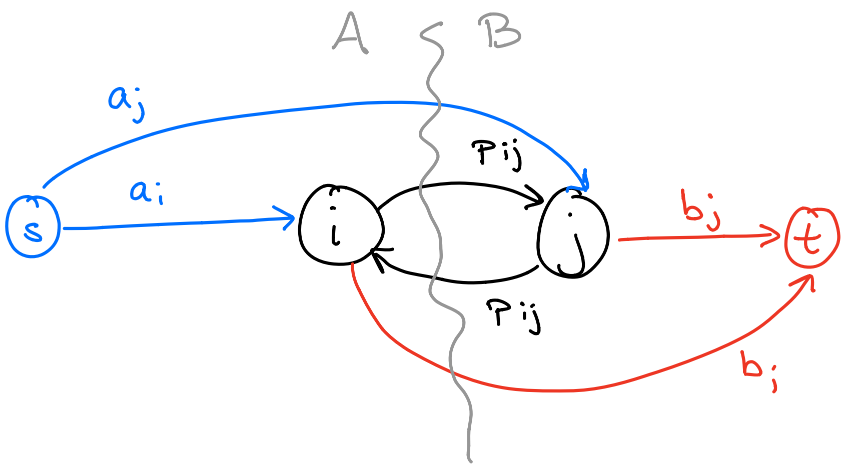
$$\min_{A,B} \sum_{i \in A} b_i + \sum_{j \in B} a_j + \sum_{\substack{(i,j) \in E \\ \text{from } A \text{ to } B}} p_{ij}$$



Reduction to MinCut

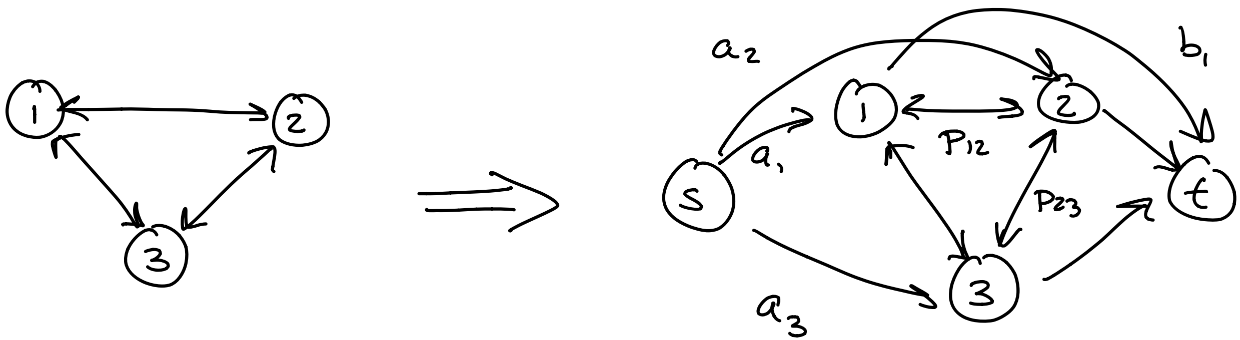
- How should the reduction work?
  - capacity of the cut should correspond to the function we're trying to minimize

$$\min_{A,B} \sum_{i \in A} b_i + \sum_{j \in B} a_j + \sum_{\substack{(i,j) \in E \\ \text{from } A \text{ to } B}} p_{ij}$$



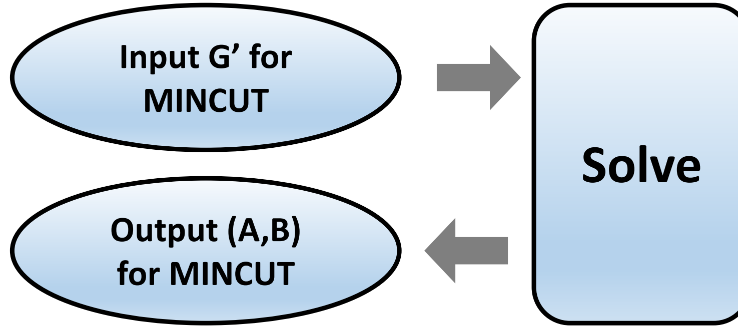
Step 1: Transform the Input

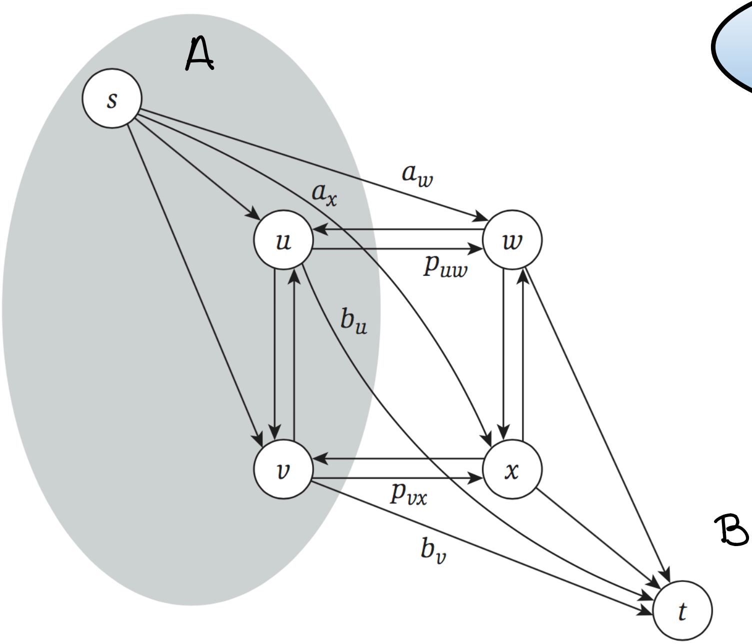
Input G,{a,b,p} for SEG → **Input G’ for MINCUT**



time : $O(m+n)$

Step 2: Receive the Output





time: time to solve min cut on a graph with

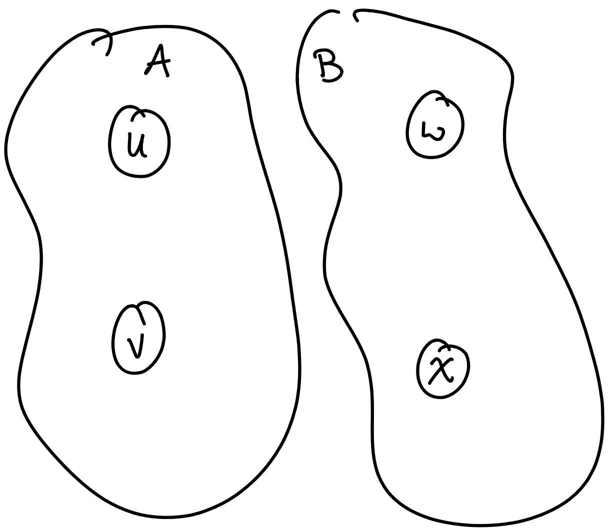
$n+2$ nodes

$2n+m$ edges

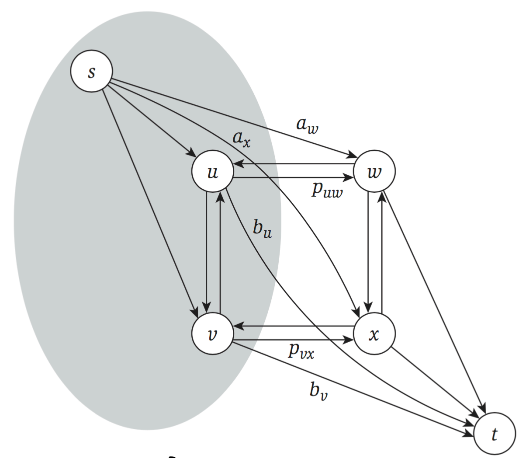
$O(nm)$ time

Step 3: Transform the Output

Output (A,B) for SEG $\Leftarrow$ **Output (A,B) for MINCUT**



$\Leftarrow$



time: $O(n)$

Reduction to MinCut

- correctness?

$$\max_{A,B} q(A,B)$$

$$= \min_{A,B} -q(A,B)$$

$$= \min_{A,B} \operatorname{cap}(A \cup \{s\}, B \cup \{t\})$$

all the work is here

- running time?

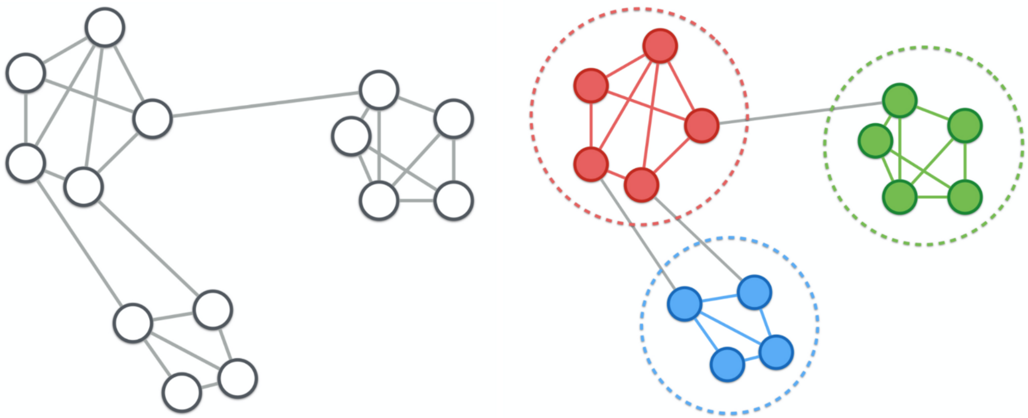
$$\left[\text{transform input}\right] + \left[\text{solve mincut}\right] + \left[\text{transform output}\right]$$

$$= O(n+m) + O(nm) + O(n)$$

$$= O(nm)$$

Densest Subgraph

~~Image Segmentation~~ Community Detection



- Want to identify communities in a network
  - "Community": a set of nodes that have a lot of connections inside and few outside

Densest Subgraph

- Input:
  - an undirected graph $G = (V, E)$
- Output:
  - Find (A, B)
  - a subset of nodes $A \subseteq V$ that maximizes $\frac{2|E(A,A)|}{|A|}$
  - $E(A,B) = \{ (u,v) \in E : u \in A, v \in B \}$
  - Why not $\max_A \frac{|E(A,A)|}{\binom{|A|}{2}}$? (Maximized by one edge)

Reduction to MinCut

We will solve the yes/no question: is there a community with density $\geq \delta$?

- Different objectives
  - find (A, B) to maximize $\frac{2|E(A,A)|}{|A|}$
  - find (A, B) to minimize $|E(A, B)|$

(MinCut if $c(e) \equiv 1$)

$\sum_{i \in A} a_i + \sum_{j \in B} b_j + \sum_{(i,j) \in E(A,B)} c_{ij}$

- Suppose $\frac{2|E(A,A)|}{|A|} \geq \delta$ and see what that implies

$\Leftrightarrow 2|E(A, A)| \geq \delta|A|$

$\Leftrightarrow \Sigma_{v \in A} \deg(v) \; - |E(A, B)| \geq \delta|A|$

$\Leftrightarrow \Sigma_{v \in V} \deg(v) \; - \Sigma_{v \in B} \deg(v) - |E(A, B)| \geq \delta|A|$

$\Leftrightarrow 2|E| \; - \Sigma_{v \in B} \deg(v) - |E(A, B)| \geq \delta|A|$

$\Leftrightarrow \Sigma_{v \in B} \deg(v) + \delta|A| + \; |E(A, B)| \leq 2|E|$

Reduction to MinCut

We will solve the yes/no question: is there a community with density $\geq \delta$?

- Different objectives
  - find (A, B) to maximize $\frac{2|E(A,A)|}{|A|}$
  - find (A, B) to minimize $|E(A, B)|$

(MinCut if $c(e) \equiv 1$)

$$\sum_{i \in A} a_i + \sum_{j \in B} b_j + \sum_{(i,j) \in E(A,B)} c_{ij}$$

- Suppose $\frac{2|E(A,A)|}{|A|} \geq \delta$ and see what that implies

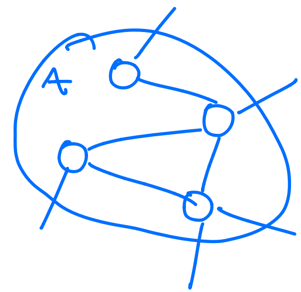
$$\Leftrightarrow 2|E(A,A)| \geq \delta|A|$$

$$\Leftrightarrow \Sigma_{v\in A} \deg(v) \; - |E(A,B)| \geq \delta|A|$$

$$\Leftrightarrow \Sigma_{v\in V} \deg(v) \; - \Sigma_{v\in B} \deg(v) - |E(A,B)| \geq \delta|A|$$

$$\Leftrightarrow 2|E| \; - \Sigma_{v\in B} \deg(v) - |E(A,B)| \geq \delta|A|$$

$$\Leftrightarrow \Sigma_{v\in B} \deg(v) + \delta|A| + \; |E(A,B)| \leq 2|E|$$



$$\sum_{v \in A} \deg(v)$$

$$= 2|E(A,A)| + |E(A,B)|$$

Reduction to MinCut

We will solve the yes/no question: is there a community with density $\geq \delta$?

- Different objectives
  - find (A, B) to maximize $\frac{2|E(A,A)|}{|A|}$
  - find (A, B) to minimize $|E(A, B)|$

(MinCut if $c(e) \equiv 1$)

$\sum_{i \in A} a_i + \sum_{j \in B} b_j + \sum_{(i,j) \in E(A,B)} c_{ij}$

- Suppose $\frac{2|E(A,A)|}{|A|} \geq \delta$ and see what that implies

$\Leftrightarrow 2|E(A, A)| \geq \delta|A|$

$\Leftrightarrow \Sigma_{v \in A} \deg(v) - |E(A, B)| \geq \delta|A|$

$\Leftrightarrow \Sigma_{v \in V} \deg(v) - \Sigma_{v \in B} \deg(v) - |E(A, B)| \geq \delta|A|$

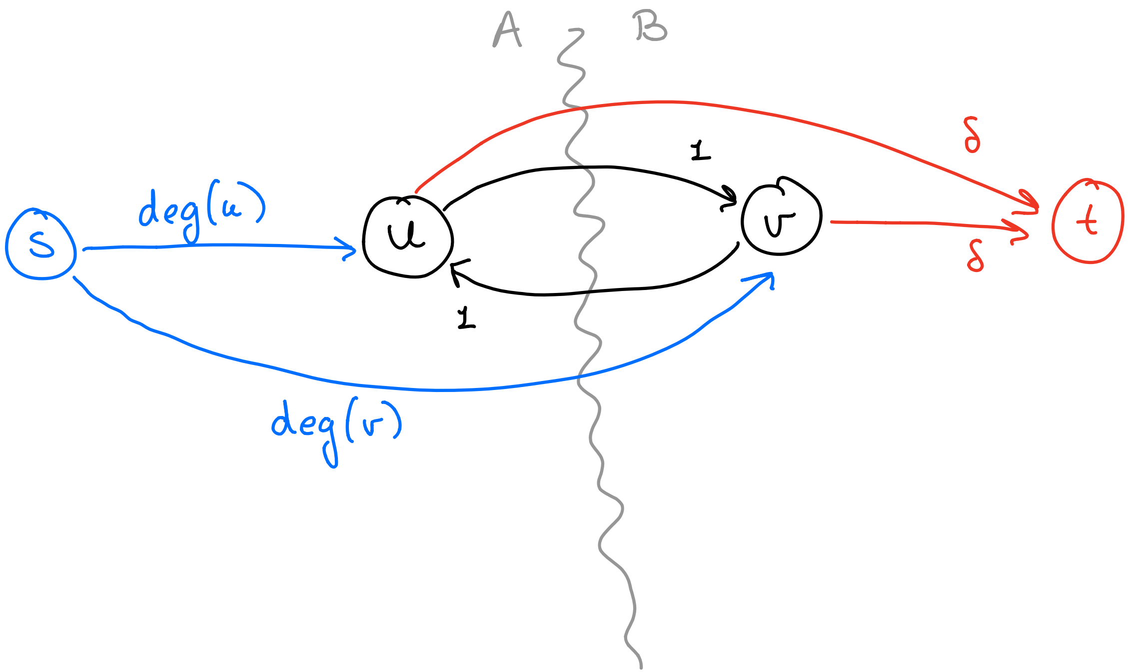
$\Leftrightarrow 2|E| - \Sigma_{v \in B} \deg(v) - |E(A, B)| \geq \delta|A|$

$\Leftrightarrow \Sigma_{v \in B} \deg(v) + \delta|A| + |E(A, B)| \leq 2|E|$

$\sum_{v \in B} \deg(v) + \sum_{v \in A} \delta + \sum_{(i,j) \in E(A,B)} 1$

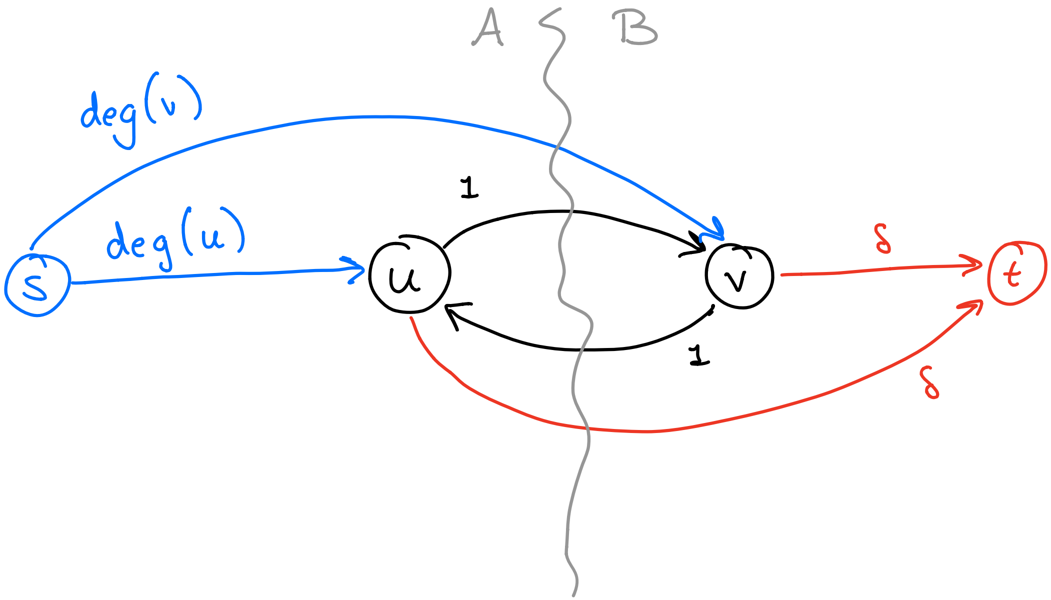
Reduction to MinCut

$$\boxed{\sum_{v \in B} \deg(v) + \delta|A| + |E(A,B)| \leq 2|E|}$$



Reduction to MinCut

$$\boxed{\sum_{v \in B} \deg(v) + \delta |A| + |E(A,B)| \leq 2|E|}$$

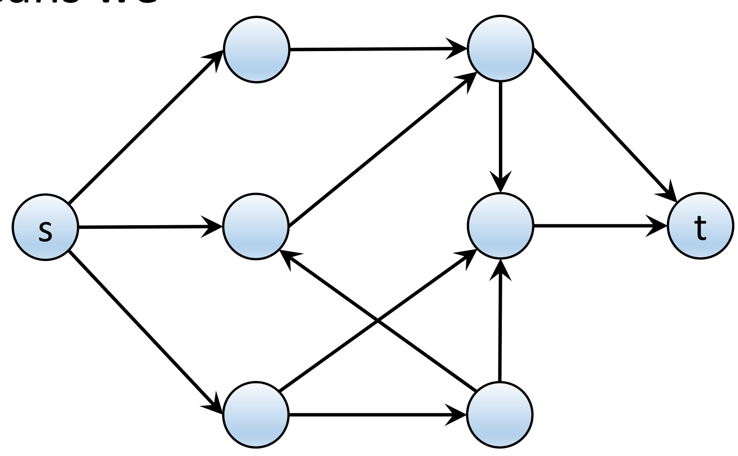


Edge-Disjoint Paths

(Edge) Disjoint Paths

- **Input:** directed graph $G = (V, E, s, t)$
- **Output:** a largest set of edge-disjoint s-t paths
  - A set of s-t paths $P_1, \ldots, P_k$ is edge disjoint if the paths do not share any edges

A large set of disjoint paths means we can tolerate edge failures.



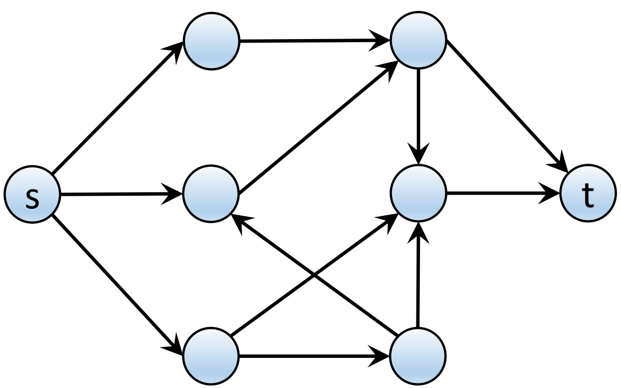
Bipartite Matching

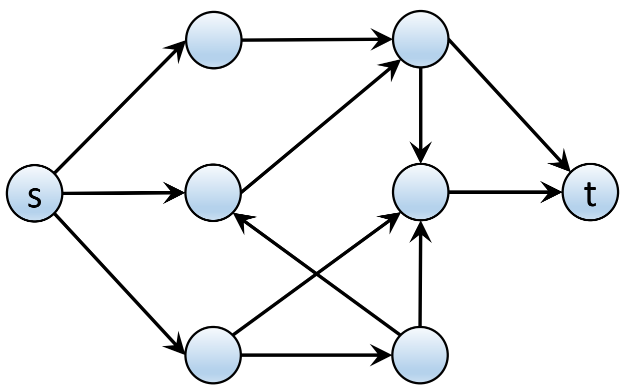
- There is a reduction that uses integer maximum s-t flow to solve edge disjoint paths.

Step 1: Transform the Input

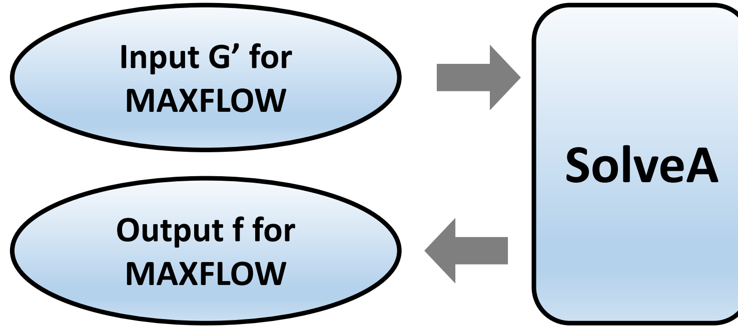
Input G for EDP

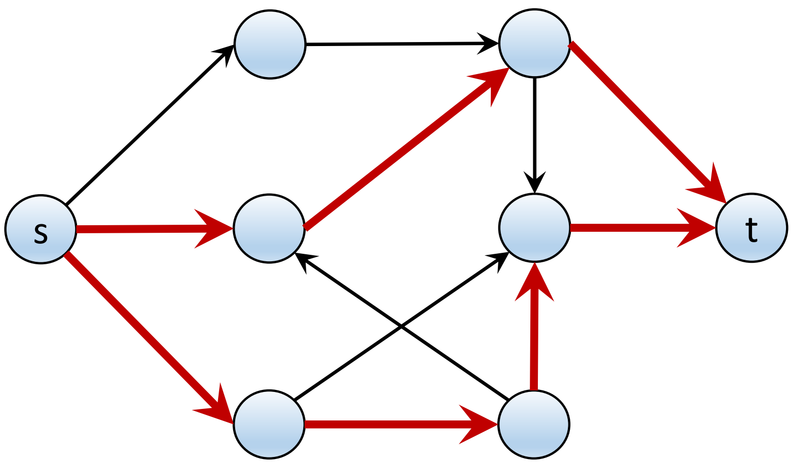
Input G’ for MAXFLOW





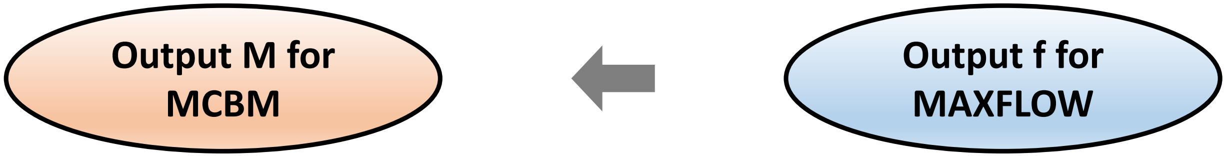
Step 2: Receive the Output





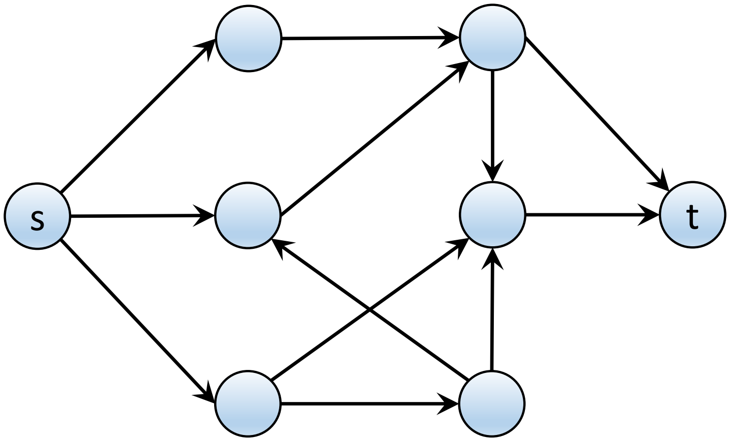
Red arrow means f(e)=1
Black means f(e) = 0

Step 3: Transform the Output



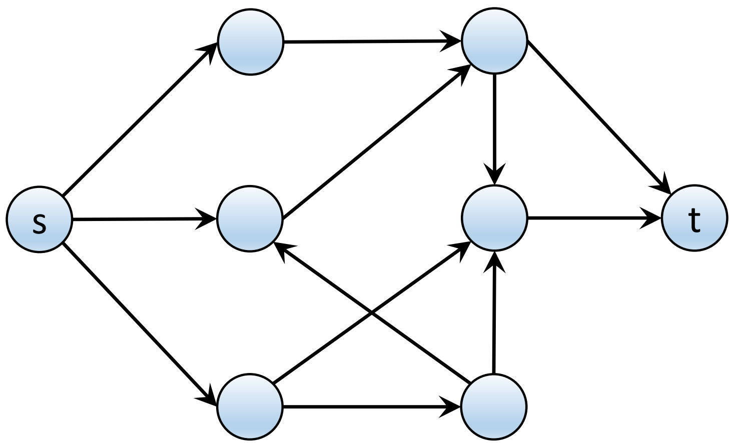
Correctness

- **Easy Direction:** If there are k edge disjoint paths then there is a flow of value k



Correctness

- **Harder Direction:** If there is a flow of value k, then there are k edge disjoint paths



Running Time

- Need to analyze the time for:
  - (1) Producing G’ given G
  - (2) Finding a maximum flow in G’
  - (3) Producing M given G’

Summary

Solving maximum s-t flow in a graph with n nodes and m edges and c(e)=1 in time T

⬇

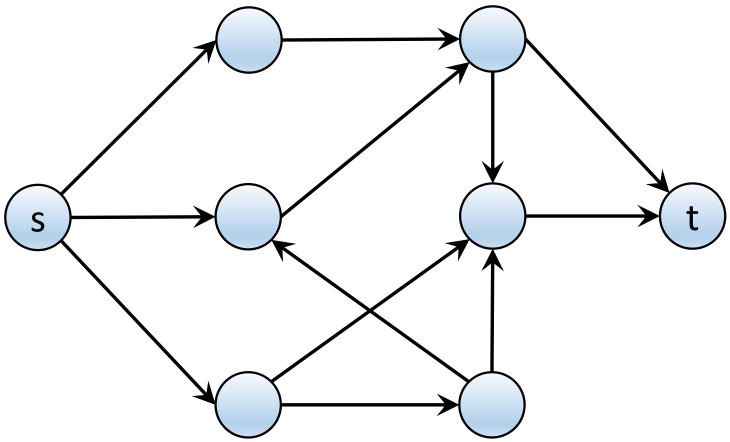
Solving edge disjoint paths in a graph with n nodes and m edges in time T + O(mn)

- Can solve edge disjoint paths in time O(nm) using Ford-Fulkerson

(Node) Disjoint Paths

- Input: directed graph $G = (V, E, s, t)$
- Output: a largest set of node-disjoint s-t paths
  - A set of s-t paths $P_1, \ldots, P_k$ is node-disjoint if the paths do not share any nodes

A large set of disjoint paths means we can tolerate edge failures.



Step 1: Transform the Input

Input G for NDP → **Input G' for EDP**