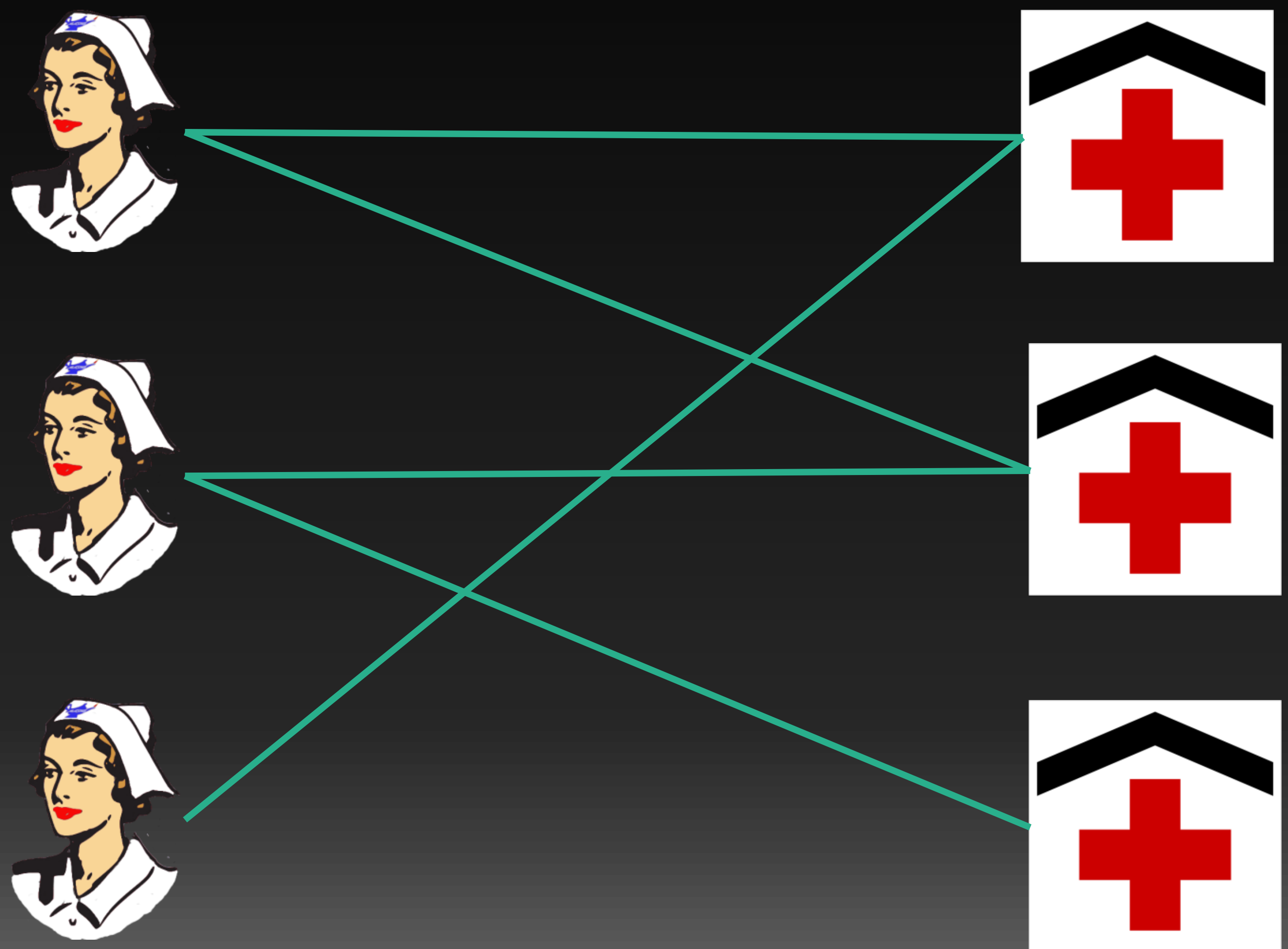
CS 4800: Algorithms & Data

Lecture 21

April 7, 2017

Bipartite matching

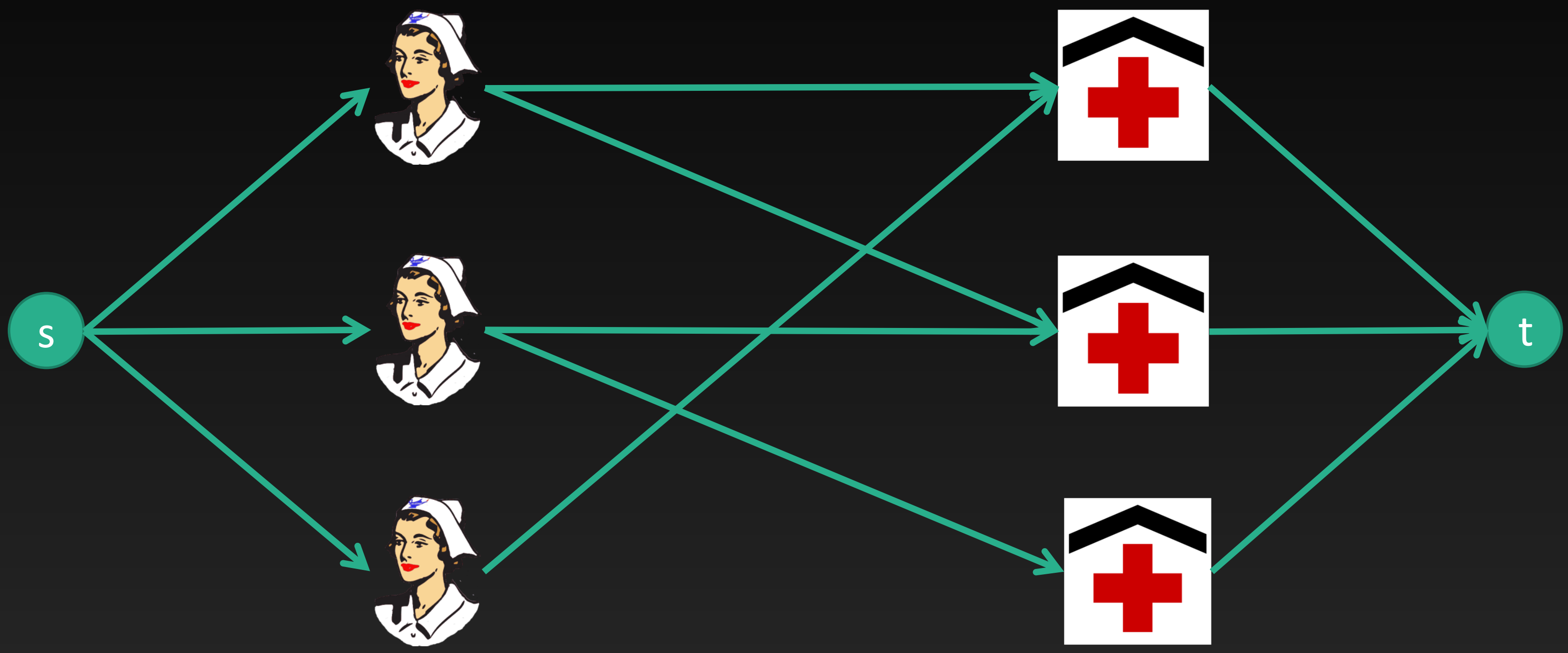
Bipartite matching



Bipartite matching

- Given graph $G = (L \cup R, E)$ where the edges are between L and R
- Find the largest subset $M \subseteq E$ such that each vertex is incident to at most one edge in M

Reduction to max flow



All edges have capacity 1

Find max flow and return all middle edges e with f(e)=1

Correctness

Claim. If there is a matching of size k, then there is a flow of value k.

Proof. Let M be a matching of size k. Construct a flow f as follows.

If $(x, y) \in M$ set f(s,x) = f(x,y) = f(y,t) = 1.

Clearly f satisfies

- Capacity constraints
- Flow conservation

|f| = |M|.

Correctness

Claim. If max flow = k then algorithm finds matching of size k.

Proof. All capacities are integers so Ford-Fulkerson algorithm finds integral flow.

$$M = \{(x, y) \ s.t. \ x \in L, y \in R \ and \ f(x, y) = 1\}$$

Capacities are 1 so all edges have flow = 0 or 1.

c(s,x)=1 so each $x \in L$ is incident to at most one edge in M.
c(y,t)=1 so each $y \in R$ is incident to at most one edge in M.
Thus M is a matching.

|f|=k so there are exactly k vertices $x \in L$ with f(s,x)=1.

Each such x is incident to one edge in M and thus |M|=k.

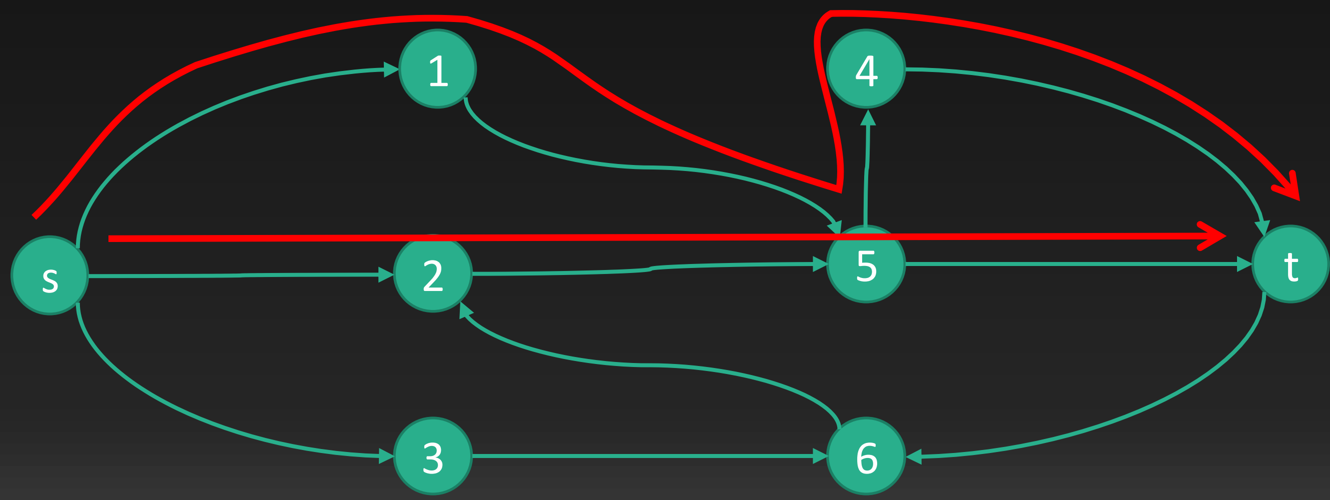
Running time

- Each augmenting path increases flow value by 1
- Max flow is at most V
- Running time of Ford-Fulkerson for bipartite matching is O(VE)

Network design

Edge-disjoint paths

- Given directed graph G = (V, E), source s, destination t
- Find max number of edge-disjoint paths from s to t



Communication network, protection against link failure

Reduction to max flow

Assign capacity 1 to every edge.

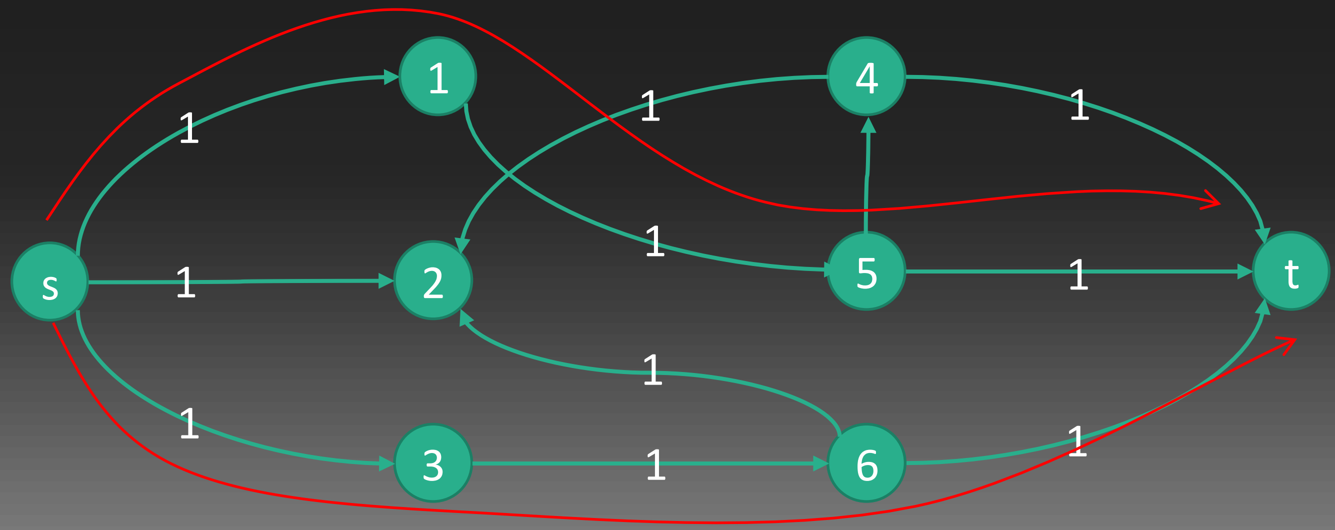
Thm. Max # edge-disjoint paths = max flow.

Proof. $\leq$

Suppose there are k paths.

Put f(e)=1 for e on the paths, f(e)=0 otherwise.

Paths are edge-disjoint so f has k edges out of s, $|f|=k$.



Reduction to max flow

Thm. Max # edge-disjoint paths = max flow.

Proof. $\geq$

Suppose $|f| = k$.

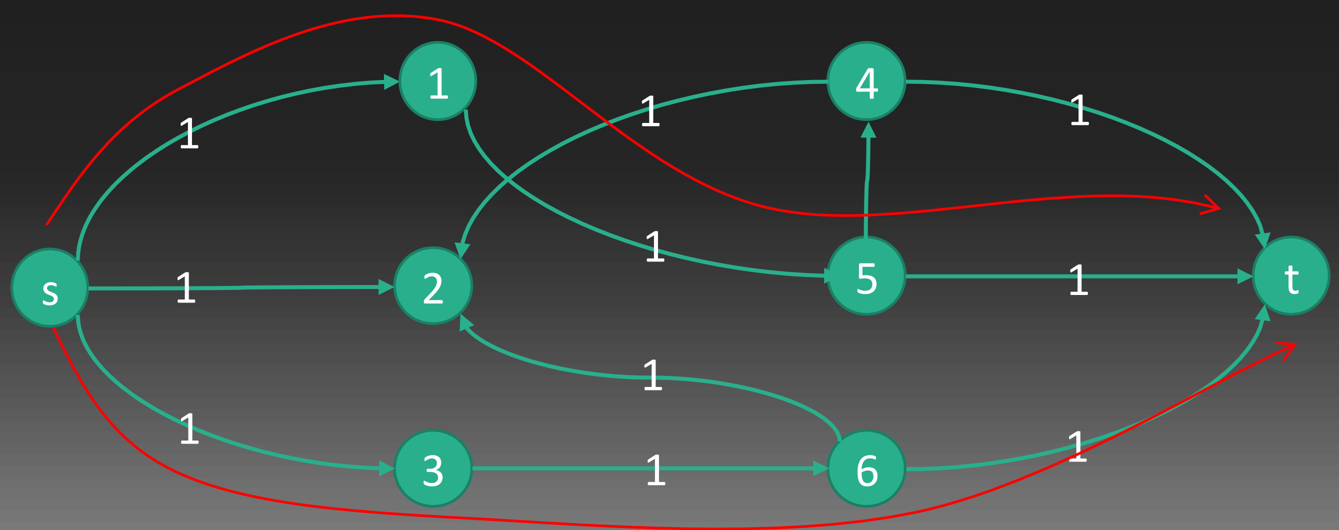
Ford-Fulkerson implies there is an integral flow of value k

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

By flow conservation, there exists (u,v) with f(u,v)=1.

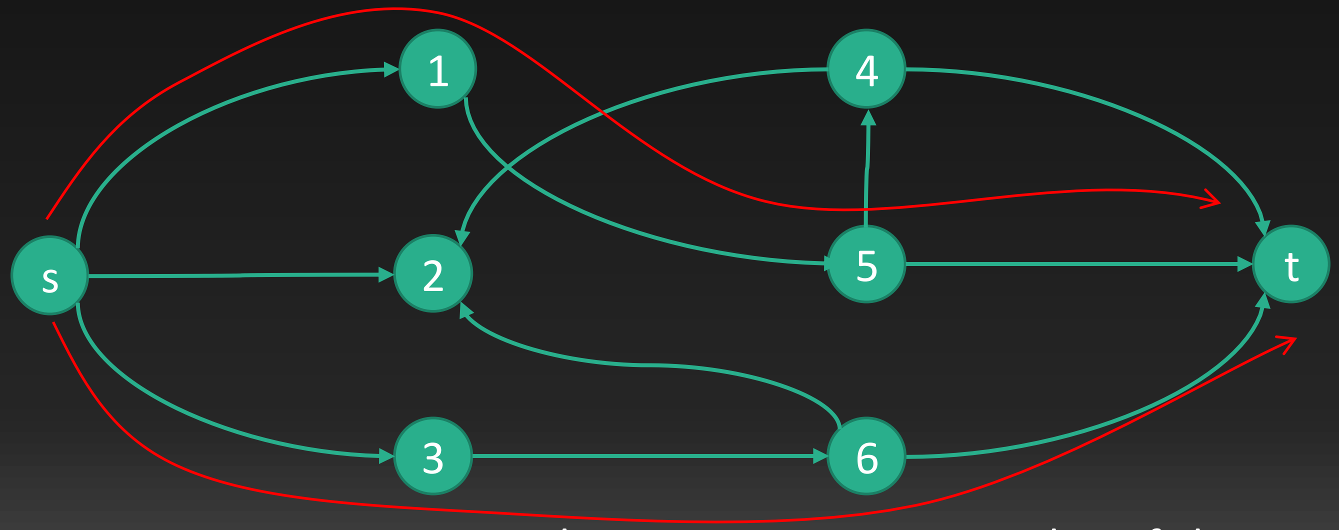
Repeatedly apply flow conservation to trace out a path to t.

$|f|$=k so k edges e out of s with f(e)=1 → k edge disjoint paths.



Node-disjoint paths

- Given directed graph G = (V, E), source s, destination t
- Find max number of node-disjoint paths from s to t



Communication network, protection against machine failure

Reduction to max flow

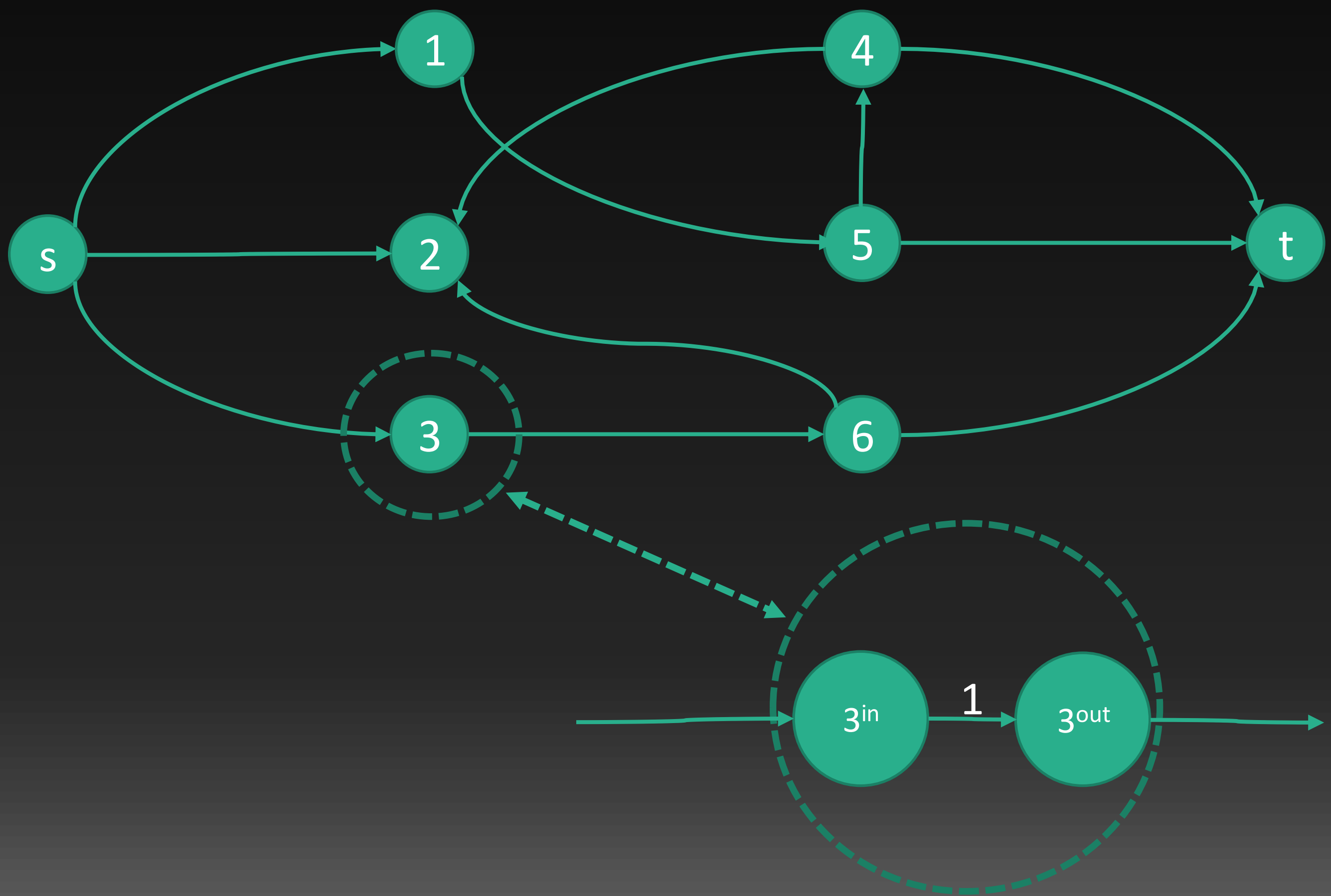
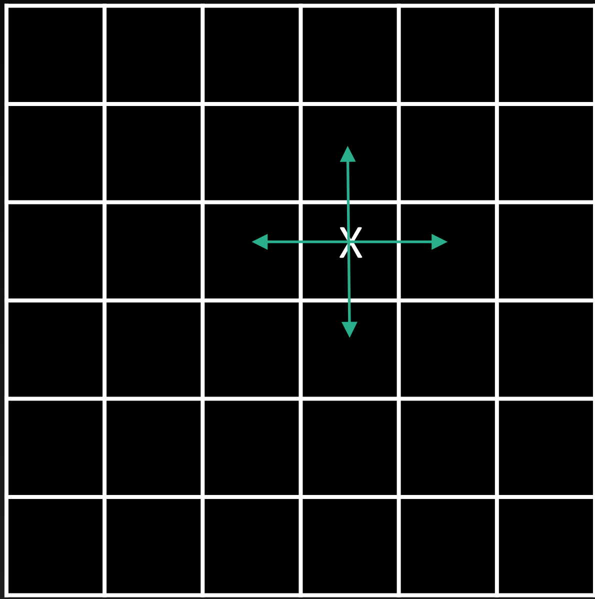


Image segmentation

Image segmentation



- Foreground/background segmentation
- Label each pixel as foreground/background
- V=set of pixels, E=neighboring pixels
- $a_i \geq 0$: likelihood of pixel i in foreground
- $b_i \geq 0$: likelihood of pixel i in background
- $p_{ij} \geq 0$: penalty of separating pixels i, j
- Goal: find partition that maximize # correct labels
- A formulation: find partition V=(A,B) that maximizes

$$\sum_{i \in A} a_i + \sum_{j \in B} b_j - \sum_{(i,j) \in E, |A \cap \{i,j\}|=1} p_{ij}$$

Reduction to min cut

- Maximizing

$$\sum_{i \in A} a_i + \sum_{j \in B} b_j - \sum_{(i,j) \in E, |A \cap \{i,j\}|=1} p_{ij}$$

- Is minimizing

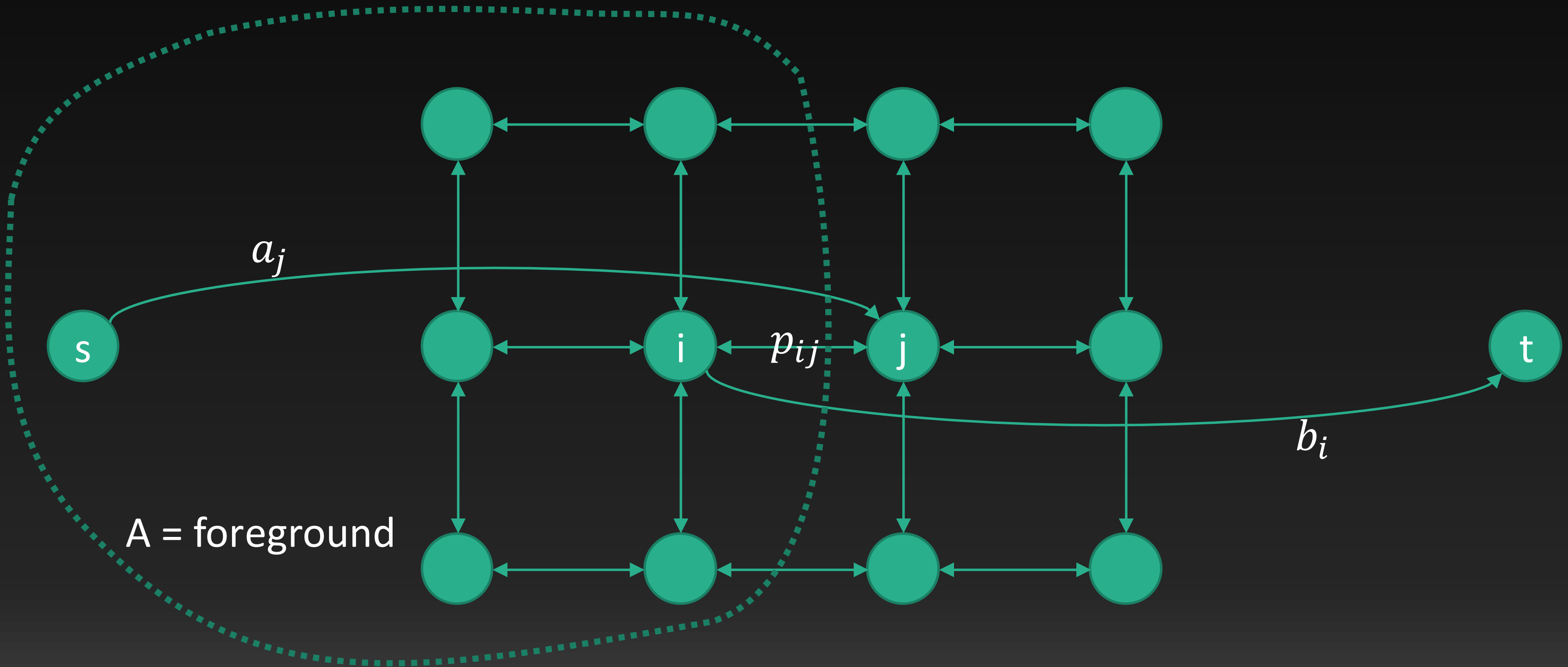
$$\sum_{i \in V} a_i + \sum_{j \in V} b_j - \left(\sum_{i \in A} a_i + \sum_{j \in B} b_j - \sum_{(i,j) \in E, |A \cap \{i,j\}|=1} p_{ij} \right)$$

- New objective

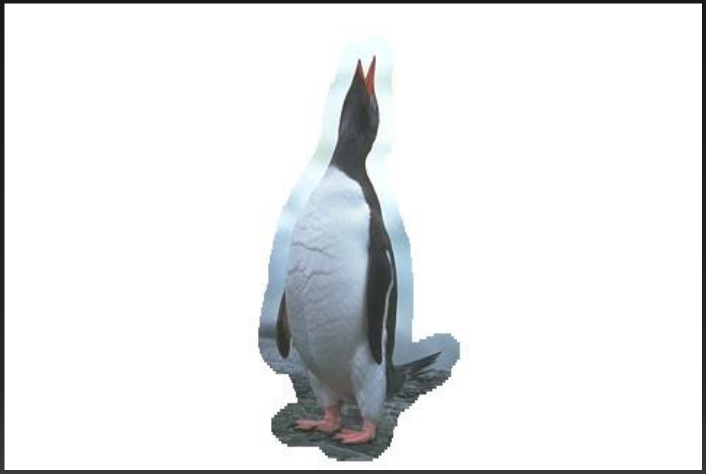
$$\min \sum_{i \in B} a_i + \sum_{j \in A} b_j + \sum_{(i,j) \in E, |A \cap \{i,j\}|=1} p_{ij}$$

Reduction to min cut

- Add source s and sink t



$$\sum_{j \in B} a_j + \sum_{i \in A} b_i + \sum_{(i,j) \in E, |A \cap \{i,j\}| = 1} p_{ij}$$

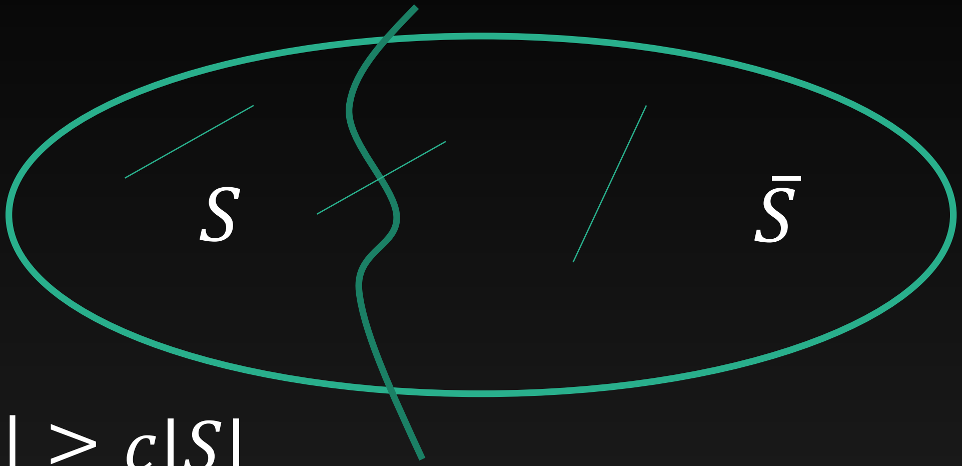


Densest subgraph

Community detection

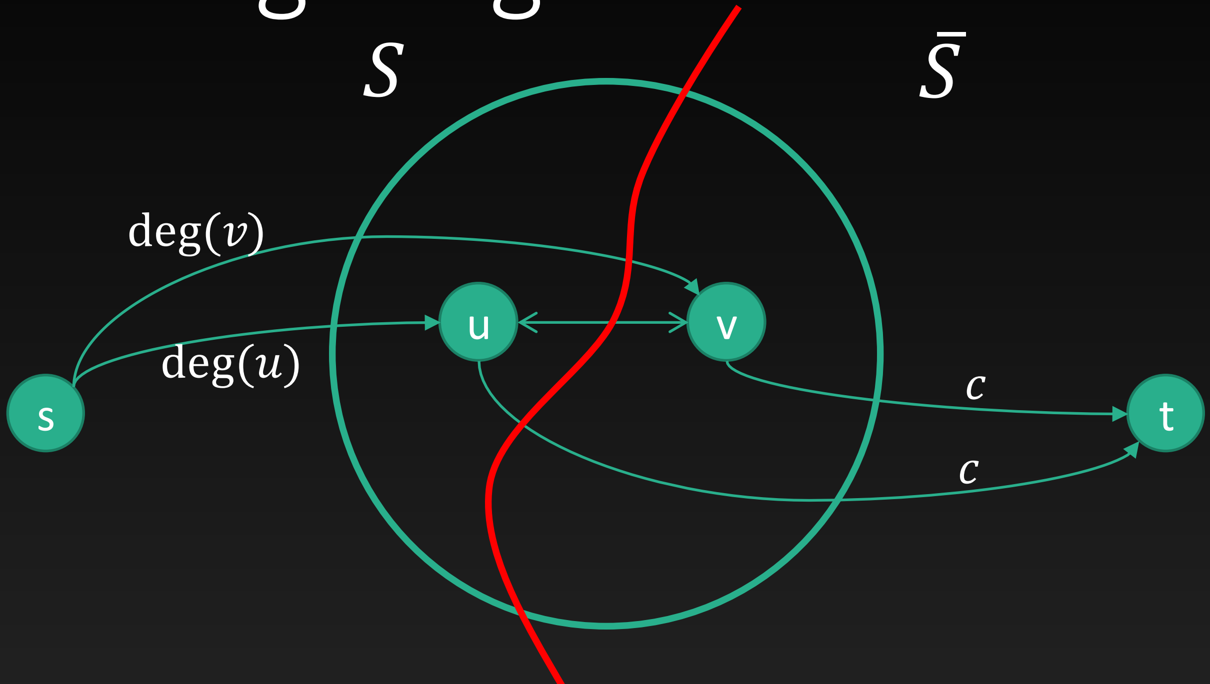
- Social network graph G = (V, E)
- Tight-knit community = dense subgraph
- Find densest subgraph $S \subset V$ that maximizes $\frac{2E(S,S)}{|S|}$

Goldberg's algorithm



- $\frac{2|E(S,S)|}{|S|} \geq c$
- $2|E(S,S)| \geq c|S|$
- $\sum_{v \in S} \deg(v) - |E(S,\bar{S})| \geq c|S|$
- $\sum_{v \in V} \deg(v) - \sum_{v \in \bar{S}} \deg(v) - |E(S,\bar{S})| \geq c|S|$
- $\sum_{v \in \bar{S}} \deg(v) + |E(S,\bar{S})| + c|S| \leq 2|E|$

Goldberg's algorithm



Cut cost = $\sum_{v \in \bar{S}} \deg(v) + |E(S, \bar{S})| + c|S|$

Check if min cut $\leq 2|E|$