



Chapter 6

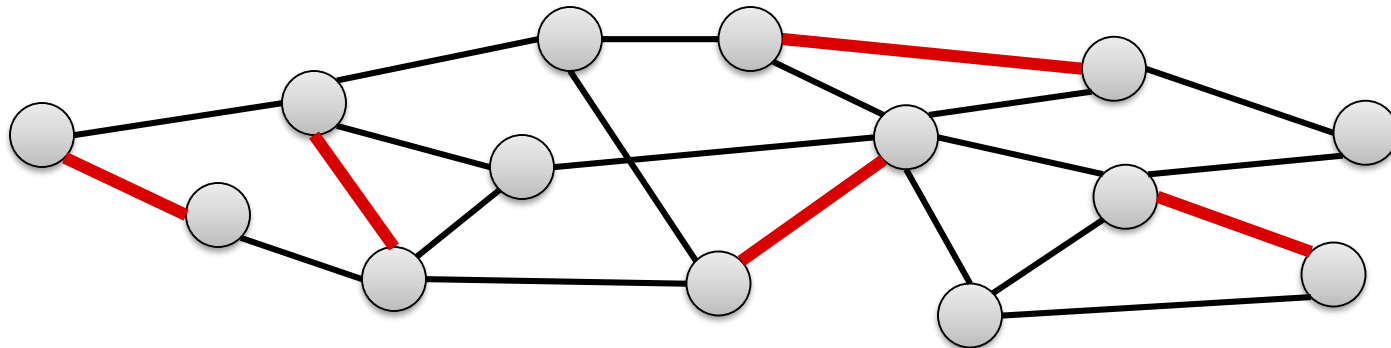
Graph Algorithms

Algorithm Theory
WS 2016/17

Fabian Kuhn

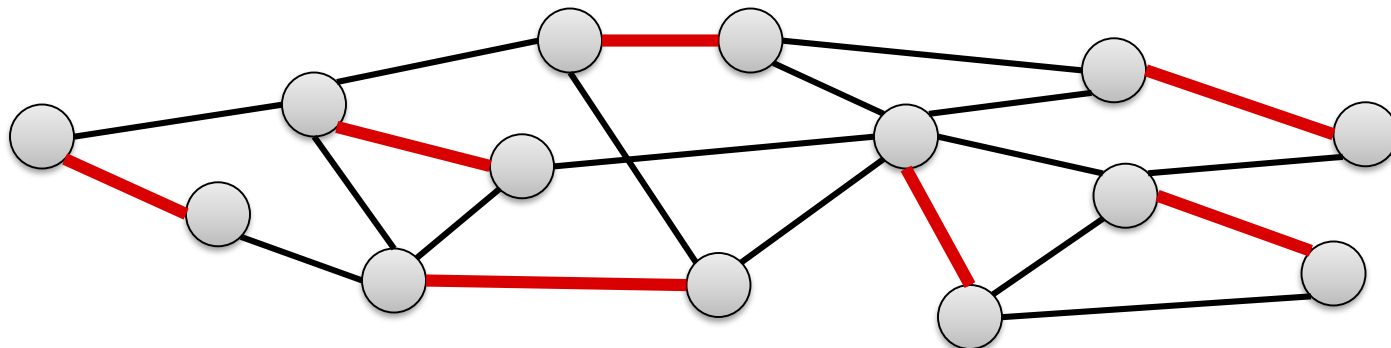
Matching

Matching: Set of pairwise non-incident edges



Maximal Matching: A matching s.t. no more edges can be added

Maximum Matching: A matching of maximum possible size



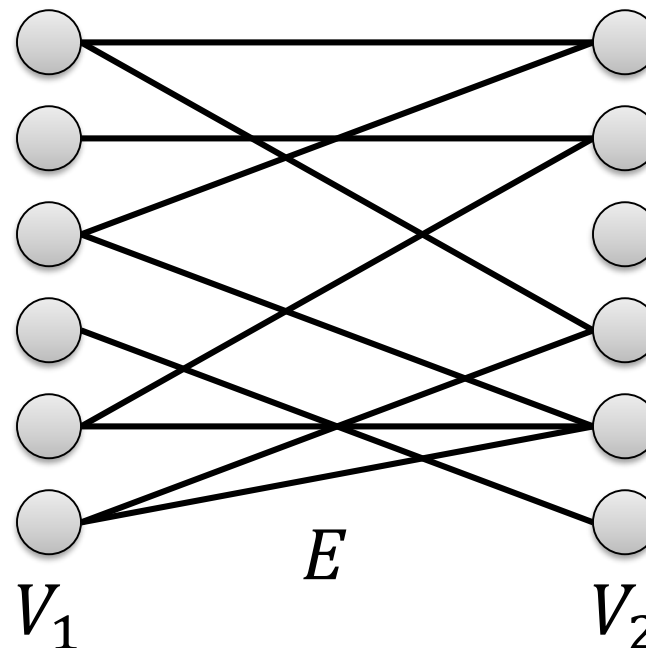
Perfect Matching: Matching of size $n/2$ (every node is matched)

Bipartite Graph

Definition: A graph $G = (V, E)$ is called bipartite iff its node set can be partitioned into two parts $V = V_1 \cup V_2$ such that for each edge $\{u, v\} \in E$,

$$|\{u, v\} \cap V_1| = 1.$$

- Thus, edges are only between the two parts



Hall's Marriage Theorem

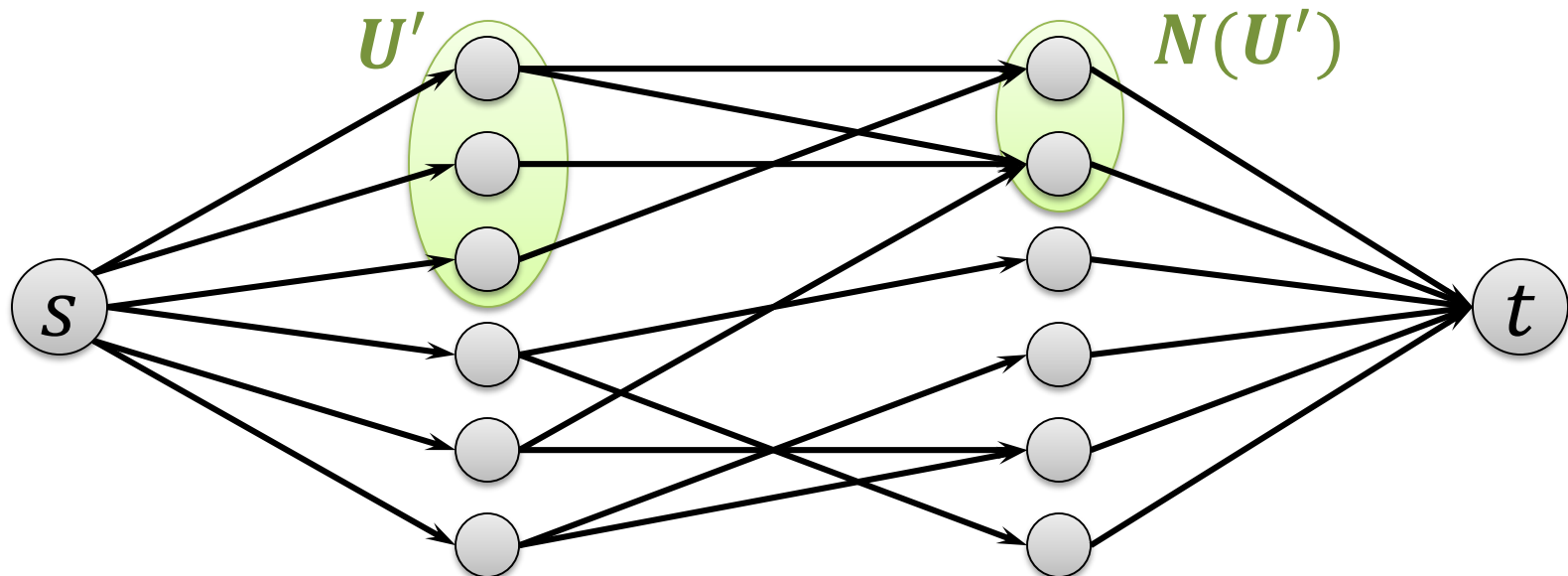
Theorem: A bipartite graph $G = (U \cup V, E)$ for which $|U| = |V|$ has a perfect matching if and only if

$$\forall U' \subseteq U: |N(U')| \geq |U'|,$$

where $N(U') \subseteq V$ is the set of neighbors of nodes in U' .

Proof: No perfect matching \Leftrightarrow some s - t cut has capacity $< n/2$

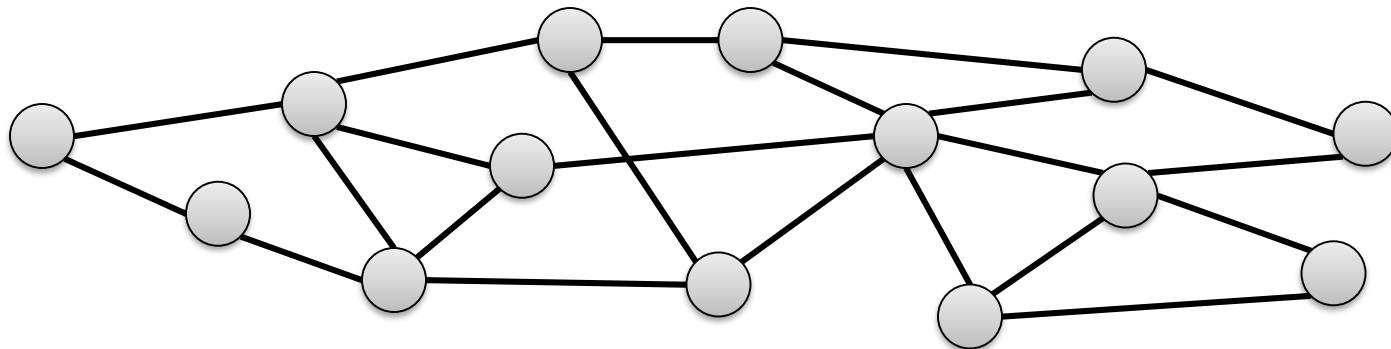
1. Assume there is U' for which $|N(U')| < |U'|$:



What About General Graphs

- Can we efficiently compute a maximum matching if G is not bipartite?
- How good is a **maximal matching**?
 - A matching that cannot be extended...
- **Vertex Cover**: set $S \subseteq V$ of nodes such that

$$\forall \{u, v\} \in E, \quad \{u, v\} \cap S \neq \emptyset.$$



- A vertex cover covers all edges by incident nodes

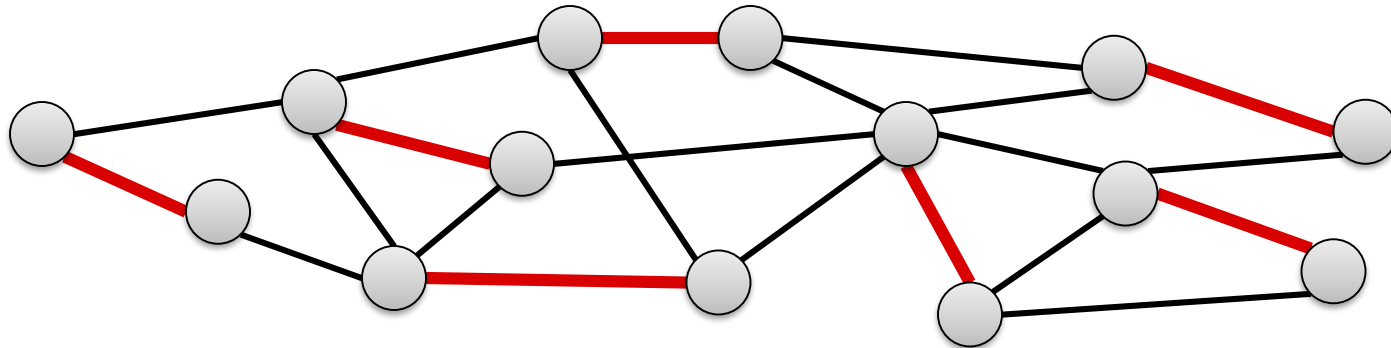
Vertex Cover vs Matching

Consider a matching M and a vertex cover S

Claim: $|M| \leq |S|$

Proof:

- At least one node of every edge $\{u, v\} \in M$ is in S
- Needs to be a different node for different edges from M



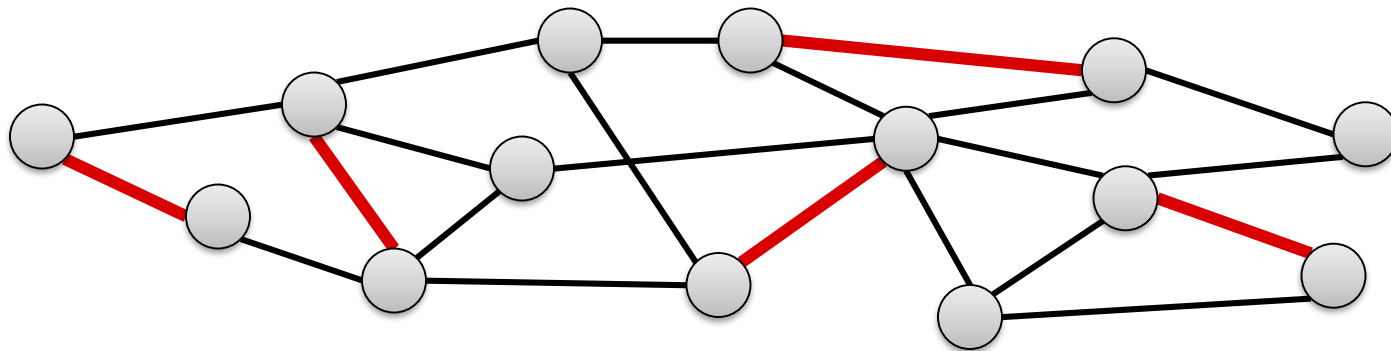
Vertex Cover vs Matching

Consider a matching M and a vertex cover S

Claim: If M is maximal and S is minimum, $|S| \leq 2|M|$

Proof:

- M is maximal: for every edge $\{u, v\} \in E$, either u or v (or both) are matched



- Every edge $e \in E$ is “covered” by at least one matching edge
- Thus, the set of the nodes of all matching edges gives a vertex cover S of size $|S| = 2|M|$.

Maximal Matching Approximation

Theorem: For any maximal matching M and any maximum matching M^* , it holds that

$$|M| \geq \frac{|M^*|}{2}.$$

Proof:

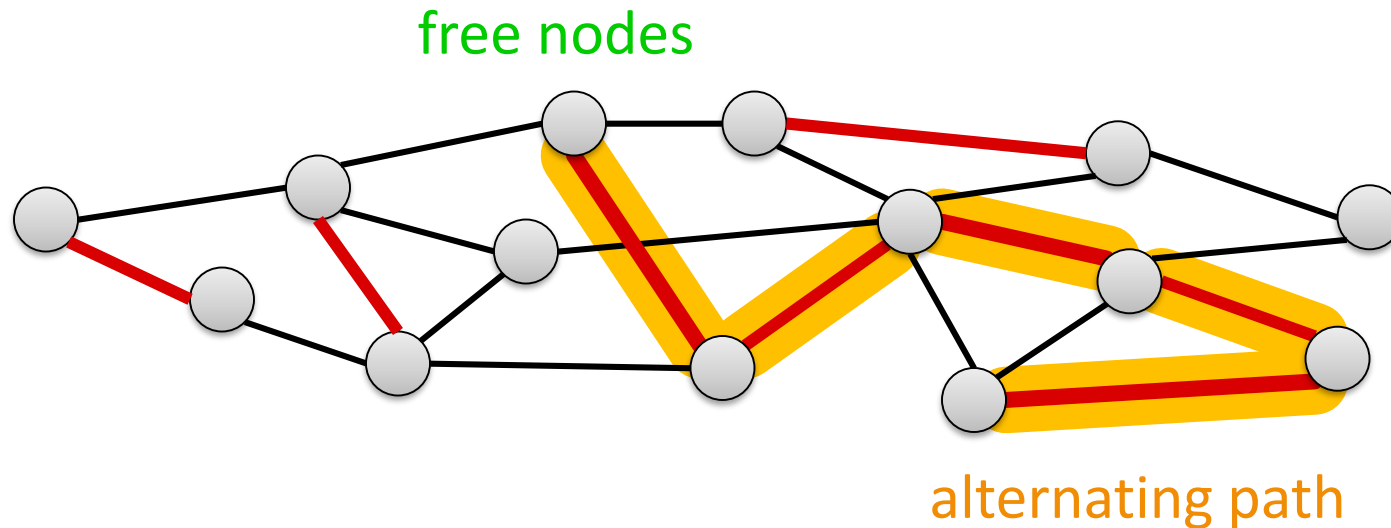
Theorem: The set of all matched nodes of a maximal matching M is a vertex cover of size at most twice the size of a min. vertex cover.

Augmenting Paths

Consider a matching M of a graph $G = (V, E)$:

- A **node** $v \in V$ is called **free** iff it is **not matched**

Augmenting Path: A (odd-length) path that starts and ends at a free node and visits edges in $E \setminus M$ and edges in M alternately.



- Matching M can be improved using an augmenting path by switching the role of each edge along the path

Augmenting Paths

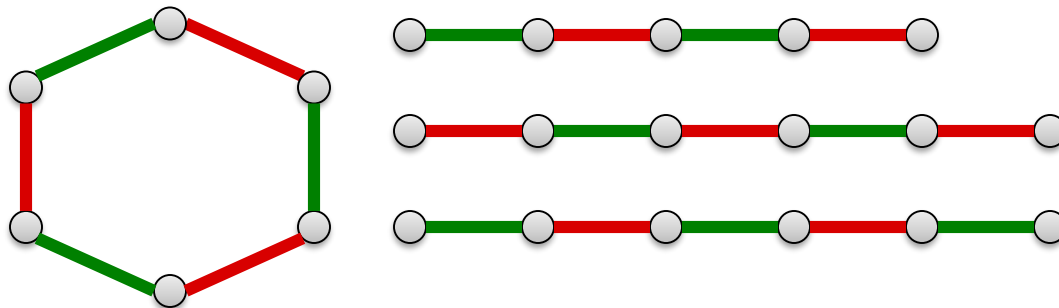
Theorem: A matching M of $G = (V, E)$ is maximum if and only if there is no augmenting path.

Proof:

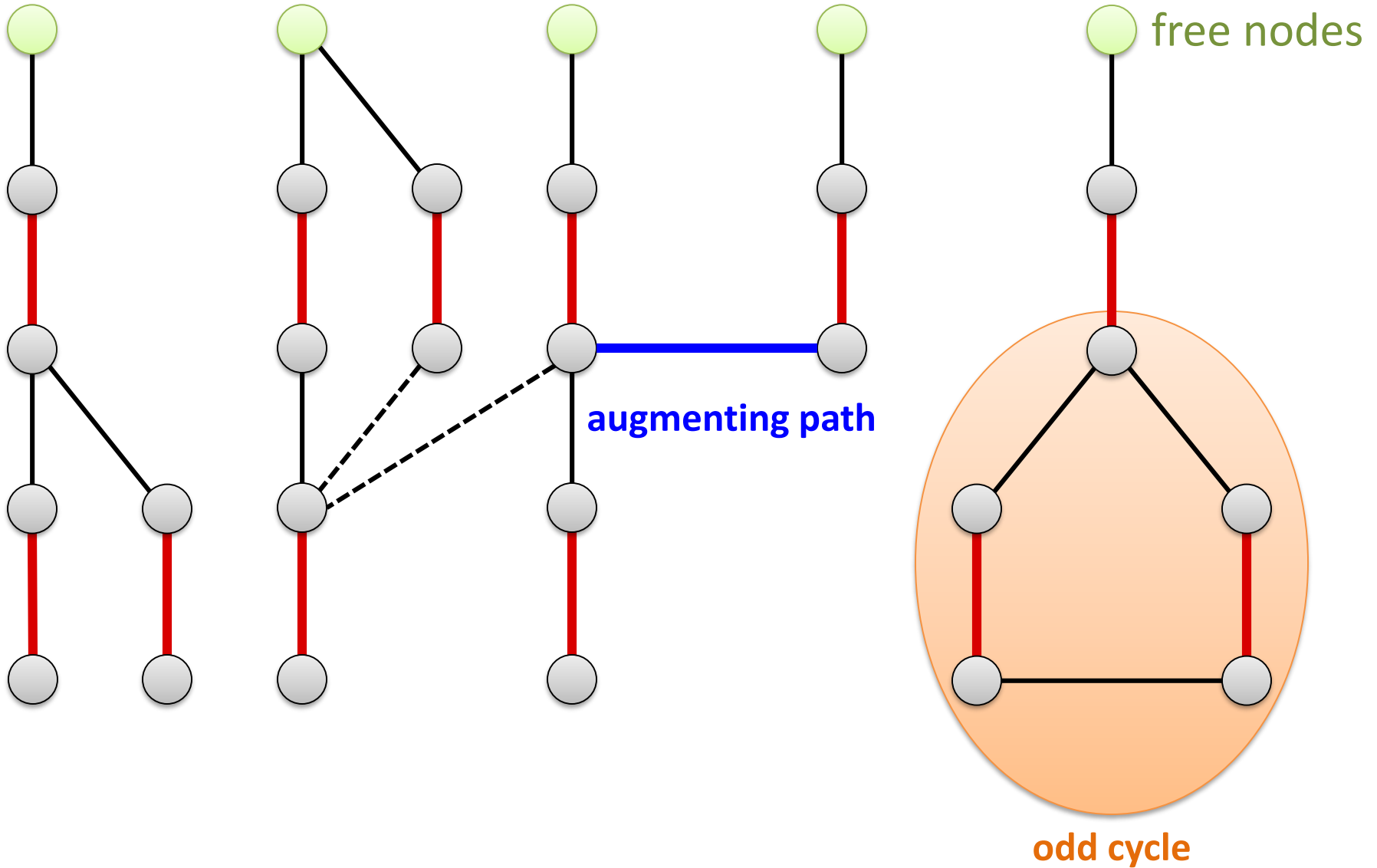
- Consider non-max. matching M and max. matching M^* and define

$$F := M \setminus M^*, \quad F^* := M^* \setminus M$$

- Note that $F \cap F^* = \emptyset$ and $|F| < |F^*|$
- Each node $v \in V$ is incident to at most one edge in both F and F^*
- $F \cup F^*$ induces even cycles and paths

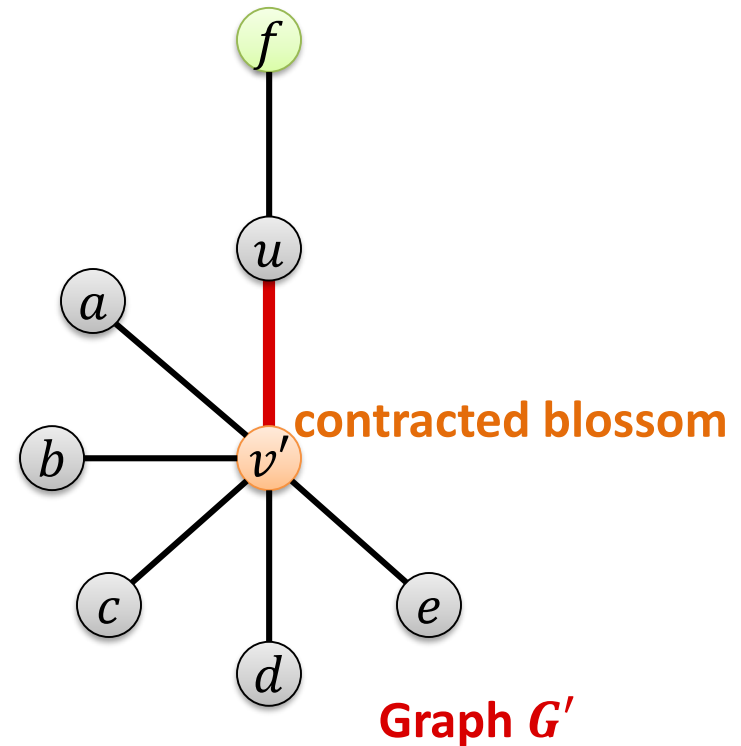
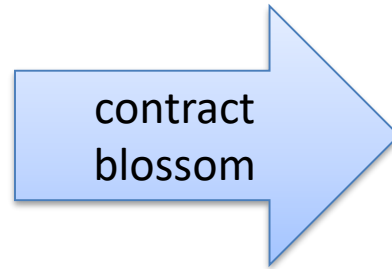
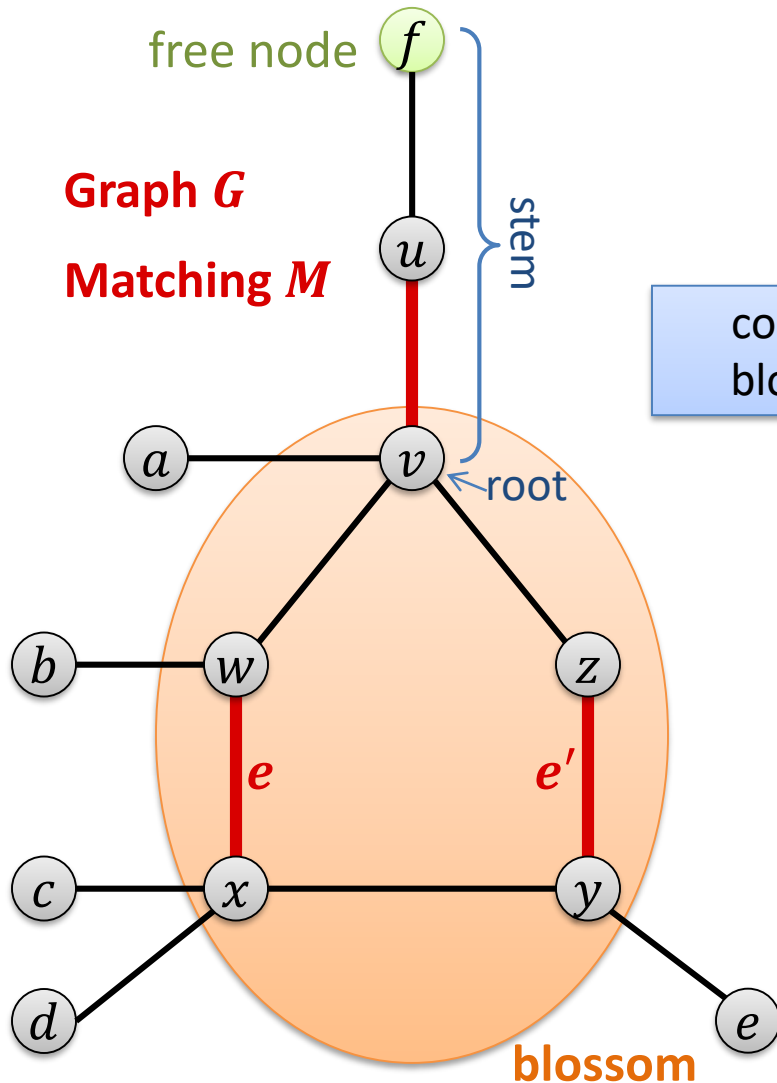


Finding Augmenting Paths



Blossoms

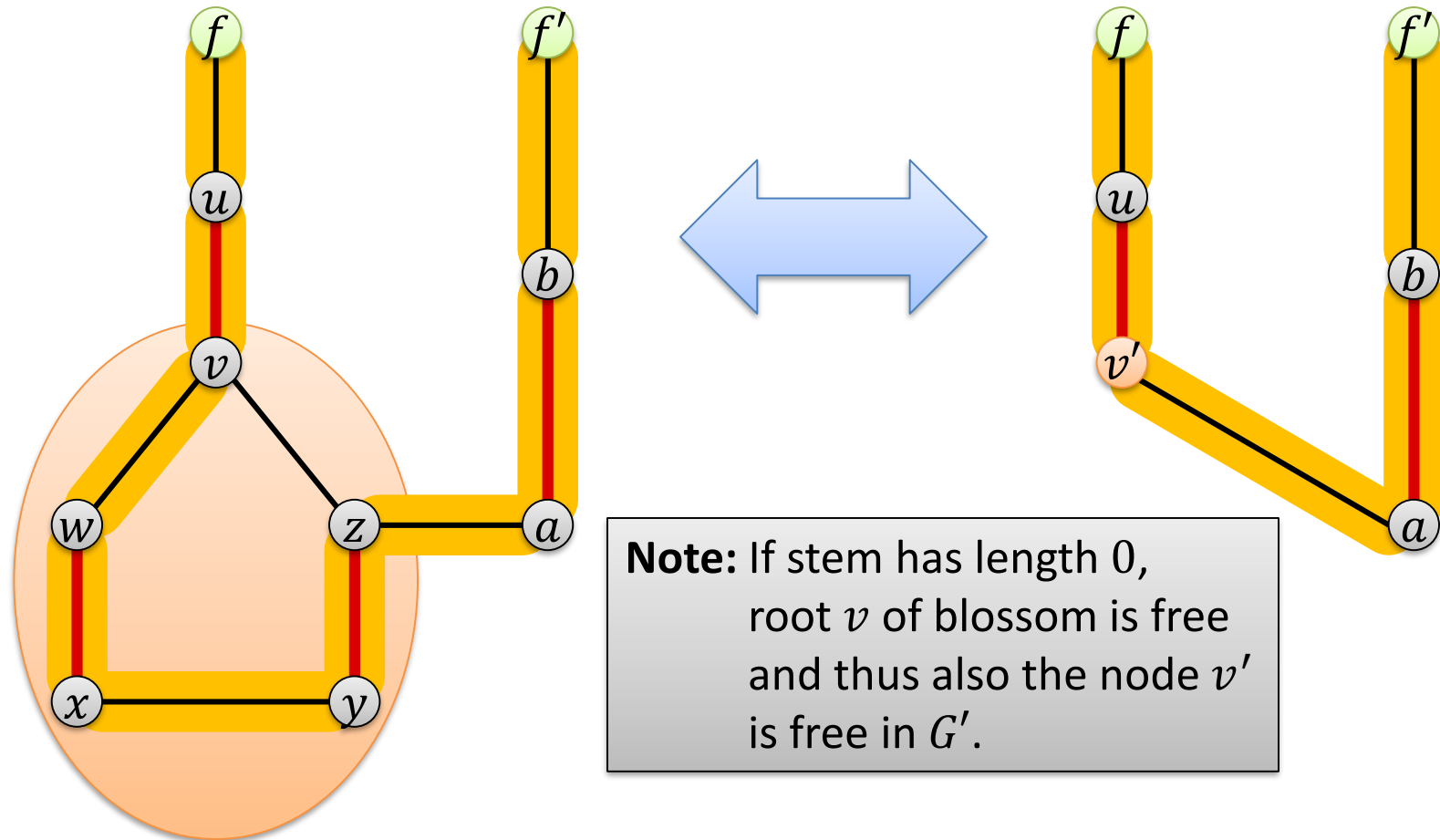
- If we find an odd cycle...



**Matching $M' = M \setminus \{e, e'\}$
is a matching of G' .**

Contracting Blossoms

Lemma: Graph G has an augmenting path w.r.t. matching M iff G' has an augmenting path w.r.t. matching M'



Note: If stem has length 0, root v of blossom is free and thus also the node v' is free in G' .

Also: The matching M can be computed efficiently from M' .

Algorithm Sketch:

1. Build a tree for each free node
2. Starting from an explored node u at even distance from a free node f in the tree of f , explore some unexplored edge $\{u, v\}$:
 1. If v is an unexplored node, v is matched to some neighbor w :
add w to the tree (w is now explored)
 2. If v is explored and in the same tree:
at odd distance from root \rightarrow ignore and move on
at even distance from root \rightarrow **blossom found**
 3. If v is explored and in another tree
at odd distance from root \rightarrow ignore and move on
at even distance from root \rightarrow **augmenting path found**

Running Time

Finding a Blossom: Repeat on smaller graph

Finding an Augmenting Path: Improve matching

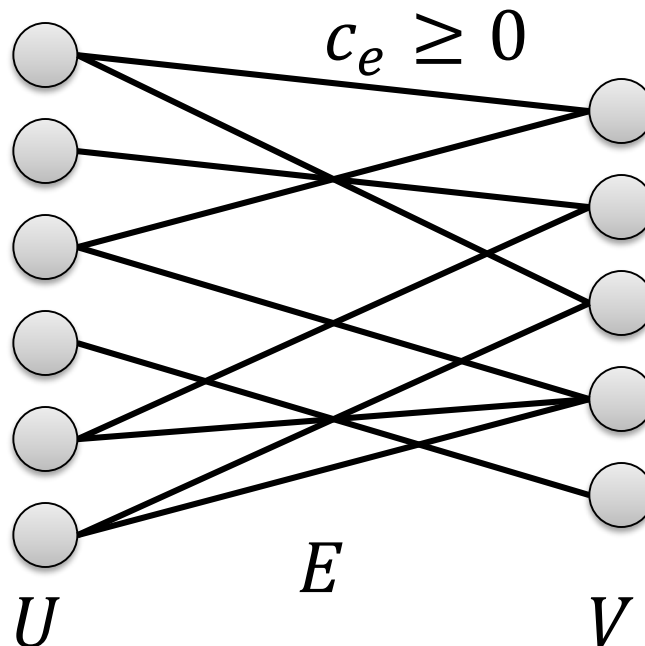
Theorem: The algorithm can be implemented in time $O(mn^2)$.

Maximum Weight Bipartite Matching

- Let's again go back to bipartite graphs...

Given: Bipartite graph $G = (U \dot{\cup} V, E)$ with edge weights $c_e \geq 0$

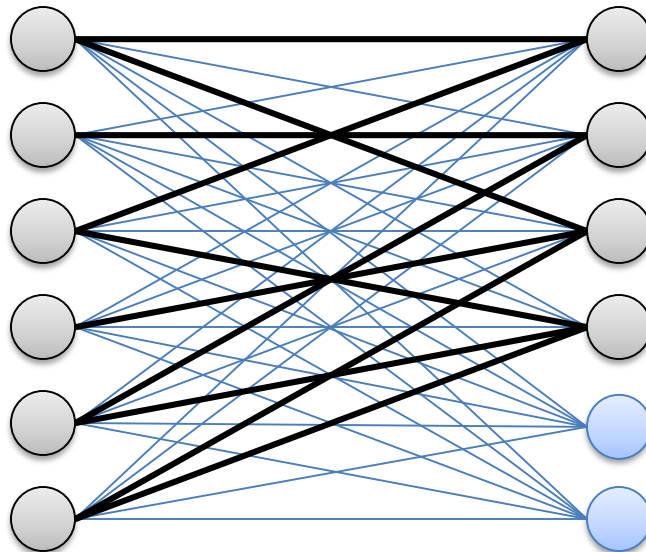
Goal: Find a matching M of maximum total weight



Minimum Weight Perfect Matching

Claim: Max weight bipartite matching is **equivalent** to finding a **minimum weight perfect matching** in a complete bipartite graph.

1. Turn into maximum weight perfect matching
 - add dummy nodes to get two equal-sized sides
 - add edges of weight 0 to make graph complete bipartite
2. Replace weights: $c'_e := \max_f \{c_f\} - c_e$



As an Integer Linear Program

- We can formulate the problem as an **integer linear program**

Var. x_{uv} for every edge $(u, v) \in U \times V$ to encode matching M :

$$x_{uv} = \begin{cases} 1, & \text{if } \{u, v\} \in M \\ 0, & \text{if } \{u, v\} \notin M \end{cases}$$

Minimum Weight Perfect Matching

Linear Programming (LP) Relaxation

Linear Program (LP)

- Continuous optimization problem on multiple variables with a linear objective function and a set of linear side constraints

LP Relaxation of Minimum Weight Perfect Matching

- Weight c_{uv} & variable x_{uv} for every edge $(u, v) \in U \times V$

$$\min \sum_{(u,v) \in U \times V} c_{uv} \cdot x_{uv}$$

s. t.

$$\forall u \in U: \sum_{v \in V} x_{uv} = 1,$$

$$\forall v \in V: \sum_{u \in U} x_{uv} = 1$$

$$\forall u \in U, \forall v \in V: x_{uv} \geq 0$$

Dual Problem

- Every linear program has a dual linear program
 - The dual of a minimization problem is a maximization problem
 - Strong duality: primal LP and dual LP have the same objective value

In the case of the minimum weight perfect matching problem

- Assign a variable $a_u \geq 0$ to each node $u \in U$
and a variable $b_v \geq 0$ to each node $v \in V$
- **Condition:** for every edge $(u, v) \in U \times V$: $a_u + b_v \leq c_{uv}$
- Given perfect matching M :

$$\sum_{(u,v) \in M} c_{uv} \geq \sum_{u \in U} a_u + \sum_{v \in V} b_v$$

Dual Linear Program

- Variables $a_u \geq 0$ for $u \in U$ and $b_v \geq 0$ for $v \in V$

$$\max \sum_{u \in U} a_u + \sum_{v \in V} b_v$$

s. t.

$$\forall u \in U, \forall v \in V: a_u + b_v \leq c_{uv}$$

- For every perfect matching M :

$$\sum_{(u,v) \in M} c_{uv} \geq \sum_{u \in U} a_u + \sum_{v \in V} b_v$$

Complementary Slackness

- A perfect matching M is optimal if

$$\sum_{(u,v) \in M} c_{uv} = \sum_{u \in U} a_u + \sum_{v \in V} b_v$$

- In that case, for every $(u, v) \in M$

$$w_{uv} := c_{uv} - a_u - b_v = 0$$

- In this case, M is also an optimal solution to the LP relaxation of the problem
- Every optimal LP solution can be characterized by such a property, which is then generally referred to as complementary slackness
- **Goal:** Find a dual solution a_u, b_v and a perfect matching such that the complementary slackness condition is satisfied!
 - i.e., for every matching edge (u, v) , we want $w_{uv} = 0$
 - We then know that the matching is optimal!

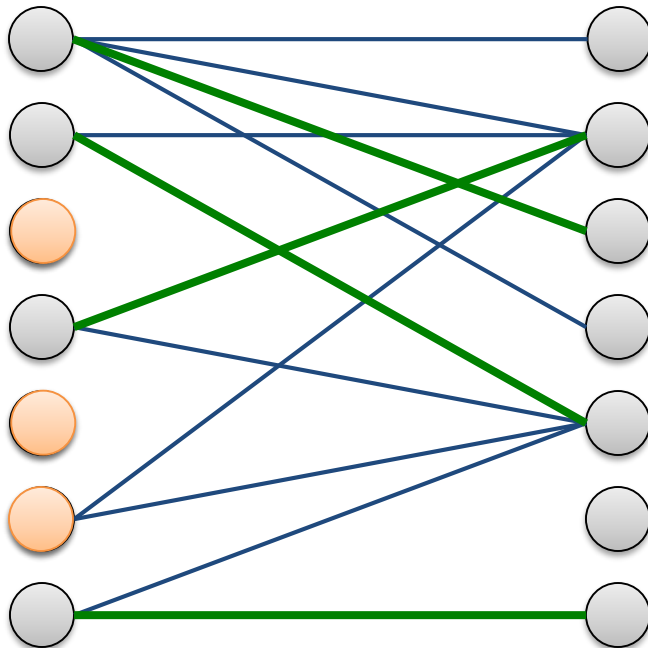
Algorithm Overview

- Start with any feasible dual solution a_u, b_v
 - i.e., solution satisfies that for all (u, v) : $c_{uv} \geq a_u + b_v$
- Let E_0 be the edges for which $w_{uv} = 0$
 - Recall that $w_{uv} = c_{uv} - a_u - b_v$
- Compute **maximum cardinality matching M of E_0**
- All edges (u, v) of M satisfy $w_{uv} = 0$
 - Complementary slackness is satisfied
 - If M is a perfect matching, we are done
- If M is **not a perfect matching, dual solution can be improved**

Marked Nodes

Define set of marked nodes L :

- Set of nodes which can be reached on alternating paths on edges in E_0 starting from unmatched nodes in U



edges E_0 with $w_{uv} = 0$

optimal matching M

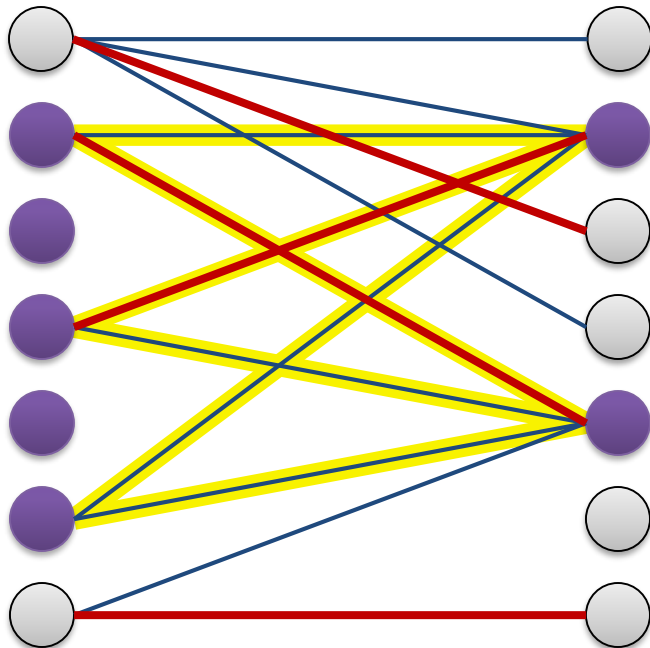
L_0 : unmatched nodes in U

L : all nodes that can be reached on alternating paths starting from L_0

Marked Nodes

Define set of marked nodes L :

- Set of nodes which can be reached on alternating paths on edges in E_0 starting from unmatched nodes in U



edges E_0 with $w_{uv} = 0$

optimal matching M

L_0 : unmatched nodes in U

L : all nodes that can be reached on alternating paths starting from L_0