Chapter 6
Graph Algorithms

Algorithm Theory
WS 2017/18

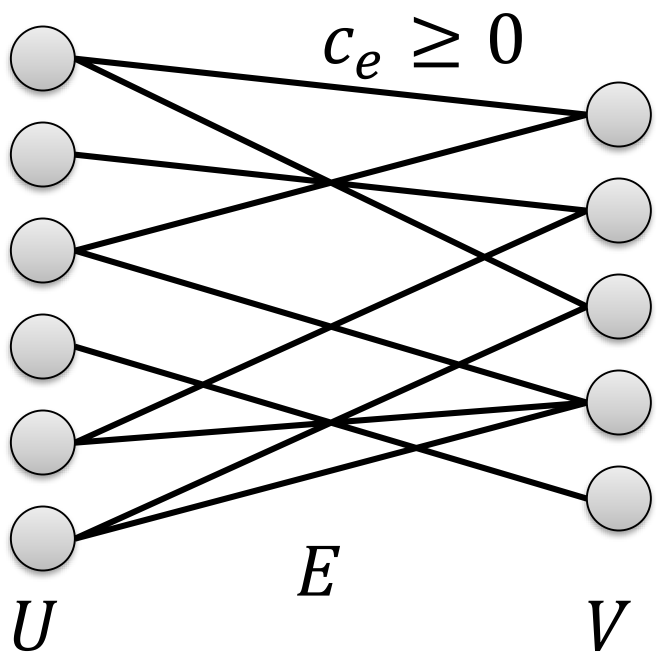
Fabian Kuhn

Maximum Weight Bipartite Matching

- Let's again go back to bipartite graphs…

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

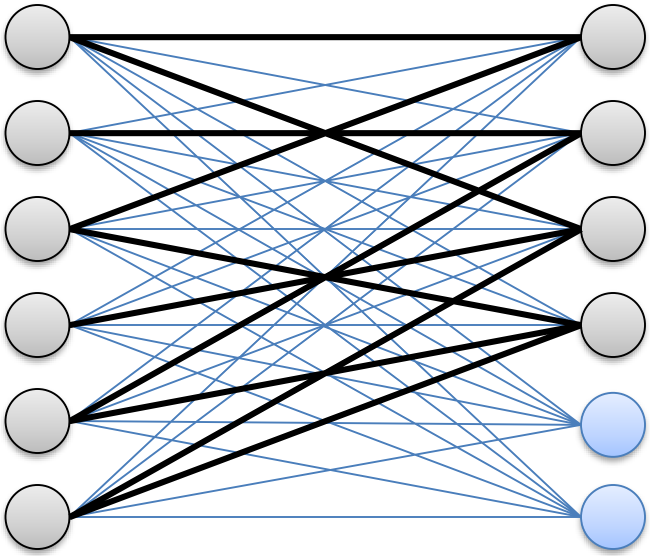
Goal: Find a matching M of maximum total weight



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: $c'_e := \max_f\{c_f\} - c_e$

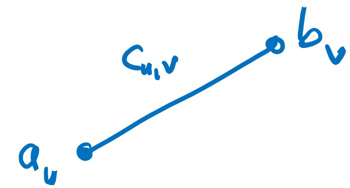


Dual Problem

- 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

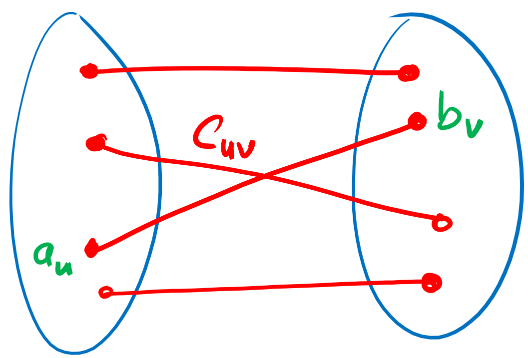
- Assign a variable $a_u \geq 0$ to each node $u \in U$
  and a variable $b_v \geq 0$ to each node $v \in V$



- **Condition:** for every edge $(\boldsymbol{u}, \boldsymbol{v}) \in \boldsymbol{U} \times \boldsymbol{V}$: $\boxed{\boldsymbol{a_u} + \boldsymbol{b_v} \leq \boldsymbol{c_{uv}}}$

- Given perfect matching M:

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



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 $(u, v) \in M$ ≥ 0

$$\boldsymbol{w_{uv}} \coloneqq c_{uv} - a_u - b_v = 0$$

  - 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): $c_{uv} \geq a_u + b_v$

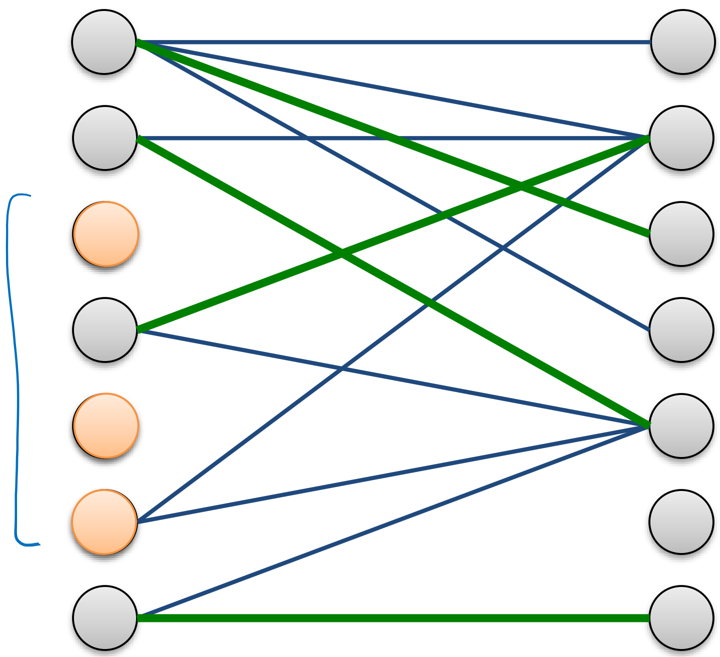
e.g., $a_u = b_v = 0$

- 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 if 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

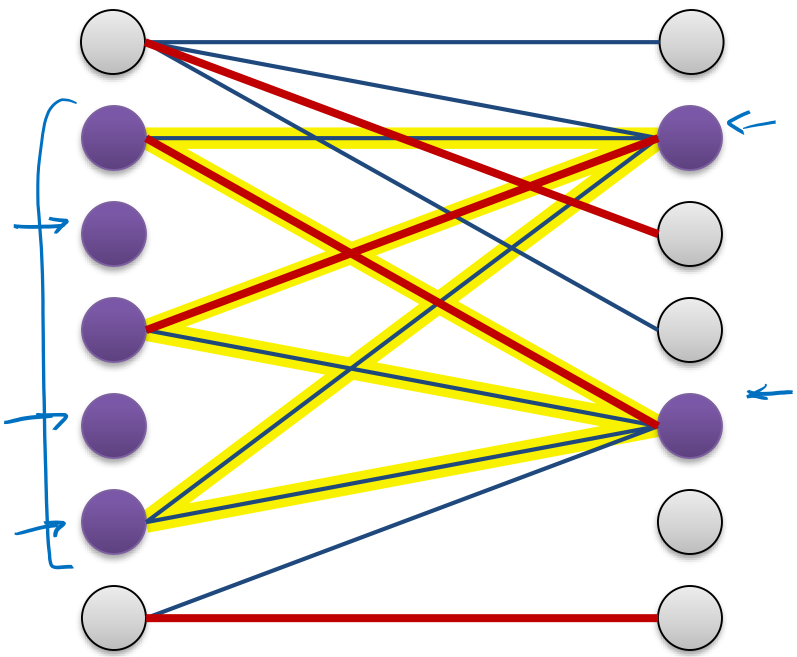
L_0: unmatched nodes in U

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

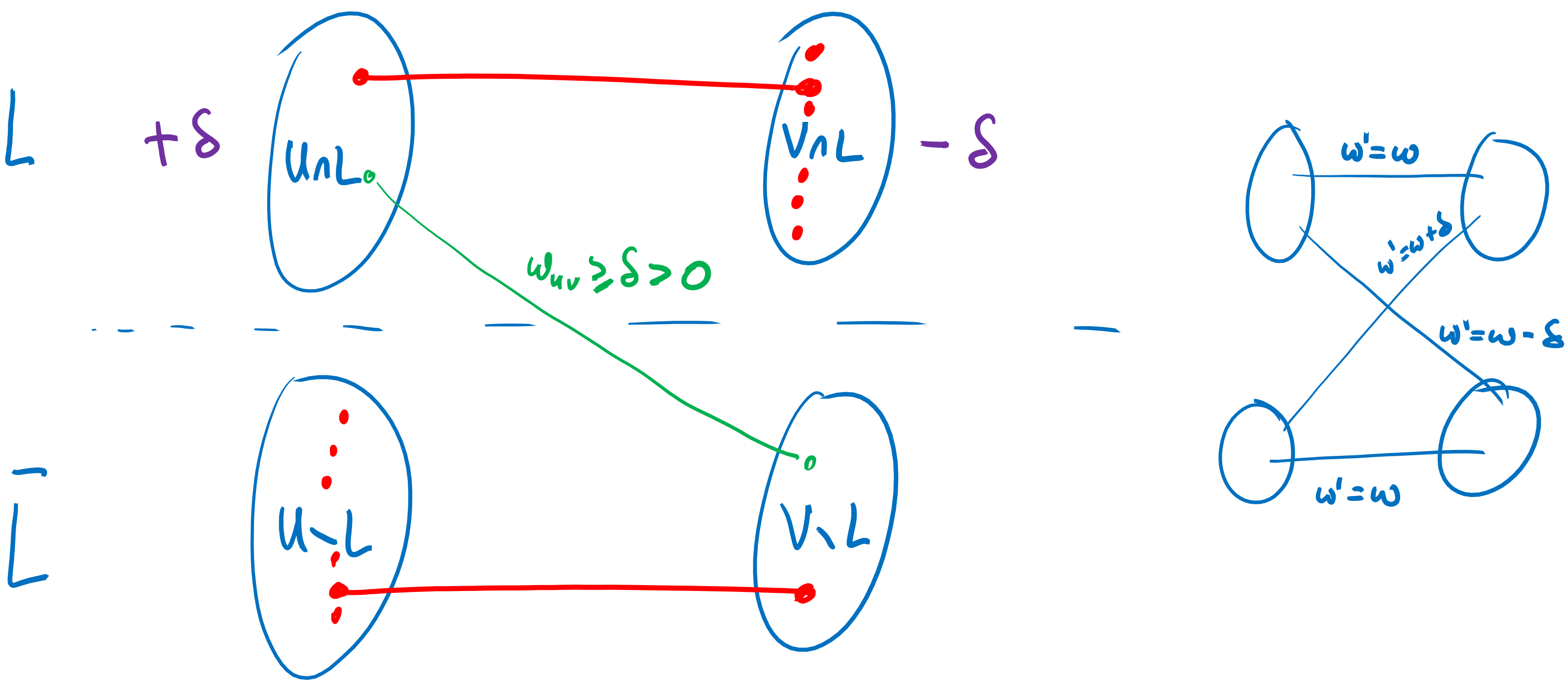
L: all nodes that can be reached on alternating paths starting from L_0

Marked Nodes – Vertex Cover

Lemma:

a) There are no E_0-edges between $U \cap L$ and $V \setminus L$

b) The set $(U \setminus L) \cup (V \cap L)$ is a vertex cover of size $|M|$ of the graph induced by E_0



Improved Dual Solution

Recall: all edges (u, v) between $U \cap L$ and $V \setminus L$ have $w_{uv} > 0$

New dual solution:

$$\delta := \min_{u \in U \cap L,\, v \in V \setminus \mathrm{L}} \{w_{uv}\}$$

$$a'_u := \begin{cases} a_u, & \text{if } u \in U \setminus \mathrm{L} \\ a_u + \delta, & \text{if } u \in U \cap L \end{cases}$$

$$b'_v := \begin{cases} b_v, & \text{if } v \in V \setminus \mathrm{L} \\ a_v - \delta, & \text{if } v \in V \cap L \end{cases}$$

Claim: New dual solution is feasible (all w_{uv} remain ≥ 0)

Improved Dual Solution

$|M| < \frac{n}{2}$

Lemma: Obj. value of the dual solution grows by $\underbrace{\delta\left(\frac{n}{2} - |M|\right)}$.

Proof:

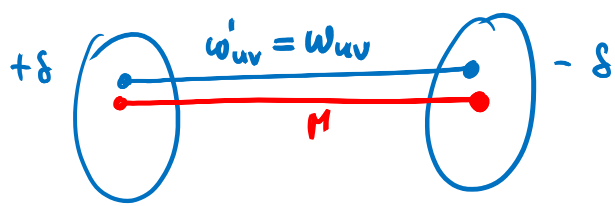
$$\delta \coloneqq \min_{u \in U \cap L,\, v \in V \setminus L} \{w_{uv}\}, \qquad a'_u \coloneqq \begin{cases} a_u, & \text{if } u \in U \setminus L \\ a_u + \delta, & \text{if } u \in U \cap L \end{cases}, \qquad b'_v \coloneqq \begin{cases} b_v, & \text{if } v \in V \setminus L \\ a_v - \delta, & \text{if } v \in V \cap L \end{cases}$$

Termination

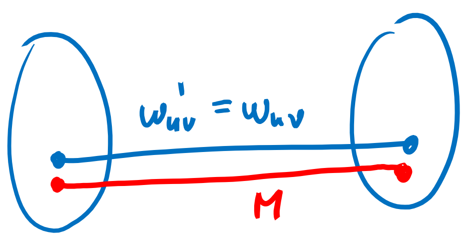
Some terminology

- Old dual solution: $\quad a_u, \quad b_v, \quad w_{uv} \coloneqq c_{uv} - a_u - b_v$
- New dual solution: $\quad a'_u, \quad b'_v, \quad w'_{uv} \coloneqq c_{uv} - a'_u - b'_v$
- $E_0 \coloneqq \{(u,v) : w_{uv} = 0\}, \quad E'_0 \coloneqq \{(u,v) : w'_{uv} = 0\}$
- M, M' : max. cardinality matchings of graphs ind. By $E_0,\ E'_0$

Claim: $|\boldsymbol{M'}| \geq |\boldsymbol{M}|$ and if $|M'| = |M|$, we can assume that $M = M'$.



$\Rightarrow M \subseteq E'_0$



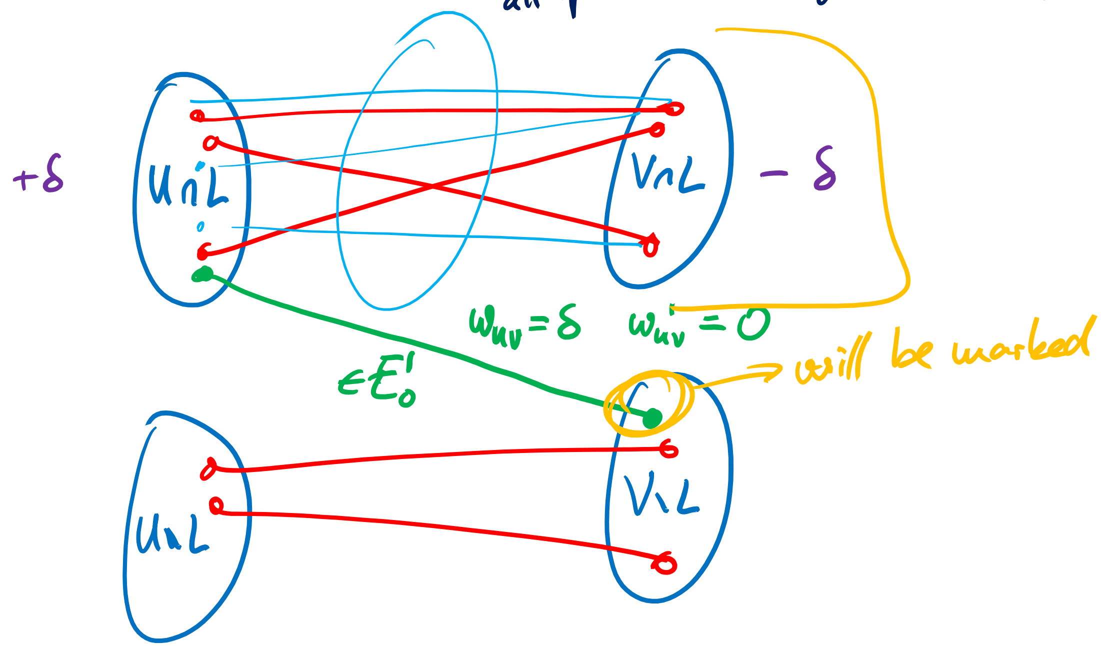
Termination

Lemma: The algorithm terminates in at most $O(n^2)$ iterations.

Proof:

- Each iteration: $M' > M$ or $M' = M$ and $|V \cap L'| > |V \cap L|$

all of those E_0-edges are in E_0'



Min. Weight Perfect Matching: Summary

Theorem: A minimum weight perfect matching can be computed in time $O(n^4)$.

$O(m \cdot n)$

- First dual solution: e.g., $a_u = 0$, $b_v = \min_{u \in U} c_{uv}$

or just $a_u = b_v = 0$

- Compute set E_0: $O(n^2)$

$O(n^2)$ edges

- Compute max. cardinality matching of graph induced by E_0
  - First iteration: $O(n^2) \cdot O(n) = O(n^3)$
  - Other iterations: $O(n^2) \cdot O(1 + |M'| - |M|)$

total cost when improving matching : $O(n^3)$

total cost when $|M| = |M'|$: $O(n^2) \cdot O(n^2) = O(n^4)$

marking : $O(n^2) \cdot O(n^2) = O(n^4)$

Matching Algorithms

We have seen:

- $O(mn)$ time alg. to compute a max. matching in *bipartite graphs*
- $O(mn^2)$ time alg. to compute a max. matching in *general graphs*

Better algorithms:

- Best known running time (bipartite and general gr.): $O(m\sqrt{n})$

Weighted matching:

- Edges have weight, find a matching of **maximum total weight**
- *Bipartite graphs*: polynomial-time primal-dual algorithm
- *General graphs*: can also be solved in polynomial time
  (Edmond's algorithm is used as blackbox)