Advanced Algorithms

Nicole Megow (Universität Bremen)

SoSe 2025

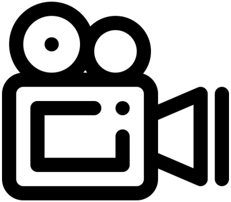
Non-Bipartite Matchings

Lecture 2

Recording of this Lecture

This lecture will be recorded

- Recording only of the lecturers by themselves.
- If there are questions from the audience, please make a clear signal if the microphone shall be muted.
- Our goal is to record the lecture, but it is no guarantee that each lecture will be recorded.



Matchings in non-bipartite Graphs

The General Maximum Matching Problem

Maximum Matching Problem

Input: a graph $G = (V, E)$.
Task: compute a maximum matching M in G, i.e., a matching M, such that for all matchings M' in G it holds: $|M| \geq |M'|$.

Goal of this lecture:

Theorem

We can find a maximum matching in polynomial time.

How did we do this for bipartite graphs? What do we need to change?

Idea for Bipartite Graphs

- Starting Point:

Theorem (Berge 1957)

A matching M in an arbitrary graph is maximum if and only if there is no M-augmenting path.

- How do we find M-augmenting paths?
- Consider the directed graph:

- A directed path from s to an exposed vertex of R gives an M-augmenting path.
- But this does not work in non-bipartite graphs. What do we do?

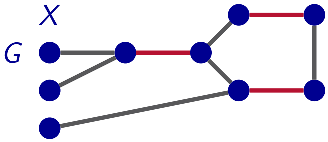
How to find augmenting paths in general?

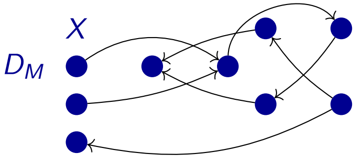
- Starting Point:

Theorem (Berge 1957)

A matching M in an arbitrary graph is maximum if and only if there is no M-augmenting path.

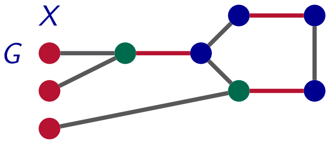
- How do we find M-augmenting paths?
- X is the set of exposed vertices. Consider a new directed graph:
  - $D_M = (V, A)$, where
    $A = \{(u, v) \mid \exists x \in V$ such that $ux \in E \setminus M$ and $xv \in M\}$.

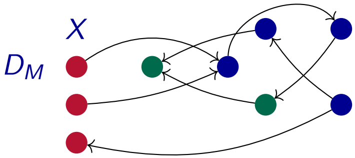




How to find augmenting paths in general?

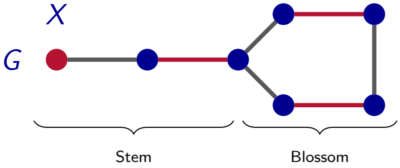
- X is the set of exposed vertices. Consider a new directed graph:
  - $D_M = (V, A)$, where
    $A = \{(u, v) \mid \exists x \in V \text{ such that } ux \in E \setminus M \text{ and } xv \in M\}$.





- An M-augmenting path corresponds to a directed path from an exposed vertex of X (red) to a neighbor of an exposed vertex of X (green).
- But the converse is not true: A directed path from a red vertex to a green vertex might not correspond to an M-augmenting path.

Flower, Blossoms, and Stems



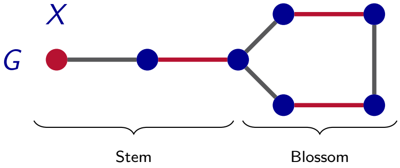
Definition (Flower, Blossom, Stem)

An M-flower is an M-alternating walk $v_0, v_1, v_2, \ldots, v_t$ (numbered so that we have $v_{2k-1}v_{2k} \in M$ and $v_{2k}v_{2k+1} \notin M$) satisfying:

1. $v_0 \in X$.
2. $v_0, v_1, v_2, \ldots, v_{t-1}$ are distinct.
3. t is odd.
4. $v_t = v_i$ for an even i.

The portion of the flower from v_0 to v_i is called the stem, while the portion from v_i to v_t is called the blossom.

Flower, Blossoms, and Stems



- How can we make use of flowers?

Lemma

Let M be a matching in G, and let $P = (v_0, v_1, \dots, v_t)$ be a shortest alternating walk from X to X. Then either P is an M-augmenting path, or $v_0, v_1, \dots, v_j$ is an M-flower for some $j < t$.

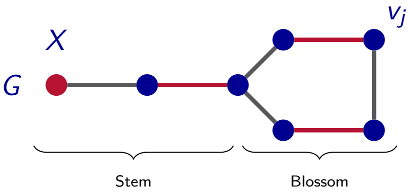
Flower, Blossoms, and Stems

Lemma

Let M be a matching in G, and let $P = (v_0, v_1, \ldots, v_t)$ be a shortest alternating walk from X to X. Then either P is an M-augmenting path, or $v_0, v_1, \ldots, v_j$ is an M-flower for some $j < t$.

- Hence, if we find an M-augmenting path like this $\rightarrow$ Update M!
- But what do we do with M-flowers?
- We will modify flowers first..

Modify Flowers



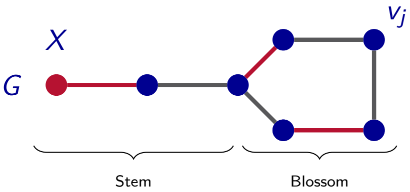
Given a flower $F = (v_0, v_1, \dots, v_t)$ with blossom B, observe that for any vertex $v_j \in B$ it is possible to modify M to a matching M' satisfying:

1. Every vertex of F is the endpoint of an edge of M', except v_j.
2. M' agrees with M outside of F, i.e., $M \Delta M' \subseteq F$.
3. $|M| = |M'|$.

- To do so, we flip edges: M' consist of all the edges of the stem which do not belong to M, together with a matching in the blossom that covers every vertex, except vertex v_j, as well as all the edges in M outside of F.

Universität Bremen

Modify Flowers

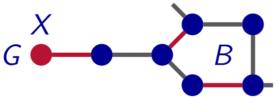


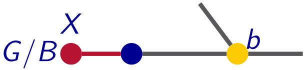
Given a flower $F = (v_0, v_1, ..., v_t)$ with blossom B, observe that for any vertex $v_j \in B$ it is possible to modify M to a matching M' satisfying:

1. Every vertex of F is the endpoint of an edge of M', except v_j.
2. M' agrees with M outside of F, i.e., $M \Delta M' \subseteq F$.
3. $|M| = |M'|$.

- To do so, we flip edges: M' consist of all the edges of the stem which do not belong to M, together with a matching in the blossom that covers every vertex, except vertex v_j, as well as all the edges in M outside of F.

What do we do with a Blossom? Shrink!





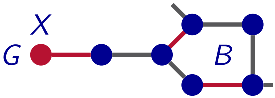
Definition (Shrinking a blossom)

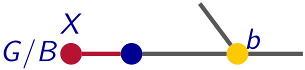
Given a graph $G = (V, E)$ with a matching $M \subseteq E$ and a blossom B, the shrunk graph G/B with matching M/B is defined as follows:

- $V(G/B) = (V \setminus B) \cup \{b\}$
- $E(G/B) = E \setminus E[B] \cup E_B$
- $M/B = M \setminus E[B]$,

where $E[B]$ denotes the set of edges within B, b is a new vertex disjoint from V, and
$E_B = \{ub \mid u \in V \setminus V(B), \text{ and } \exists v \in V(B) \text{ such that } uv \in E\}$.

What do we do with a Blossom? Shrink!





- What is the benefit of shrinking a blossom?

Theorem

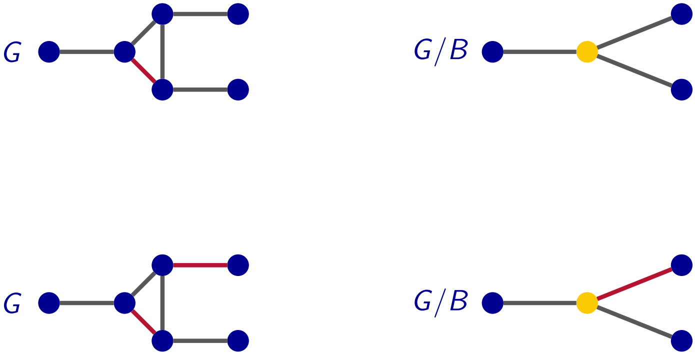
Let M be a matching of G and let B be an M-blossom. Then, M is a maximum-size matching if and only if M/B is a maximum-size matching in G/B.

Algorithm?

- How can we now compute a maximum matching?
- If M is a matching (not maximum), and B is blossom w.r.t. M
- $\Rightarrow$ then M/B is not maximum in G/B
- But: we can compute a maximum matching N in G/B
- Can turn this into matching N^+ for G
- N^+ might still not be maximum for G
- But we can simply proceed with N^+.

Example

- View the example from top left, clockwise.



Algorithm for maximum non-bipartite Matching

1. Start with the empty matching $M = \emptyset$, and let X be the set of exposed vertices (initially V).
2. Construct the directed graph D_M.
3. As long as D_M contains a path $\hat{P}$ from X to $N(X)$
   Find such a path $\hat{P}$ of minimum length (number of edges)
4. Let P be the alternating path in G corresponding to $\hat{P}$.
5. If P is an M-augmenting path, set $M = M \Delta P$. Update X, construct new directed graph D_M and go to point 3.
6. Else: P contains a blossom B. Modify flower as above.
   - Recursively find max-size matching M' in G/B.
   - if $|M'| = |M/B|$, return M (M was maximum).
   - else unshrink M' (as in proof of Theorem) to obtain matching M'' in G, where $|M''| > |M|$. Update M and X, construct D_M and go to point 3.

See also example at the blackboard.

Universität Bremen

Main Theorem

Theorem (Edmonds, 1965)

We can compute a maximum-size matching in an arbitrary graph in time $O(|E| \cdot |V|^2)$.

- The running-time can be improved:

Theorem

We can compute a maximum-size matching in an arbitrary graph in time $O(|E| \cdot \sqrt{|V|})$.

- We can also generalize this to weighted graphs:

Theorem (Edmonds, 1965)

We can compute a maximum-weight matching in an arbitrary edge-weighted graph in time $O(|E| \cdot |V|^2)$.

Recap and Outlook

- Maximum matchings in non-bipartite graphs
  - How to find augmenting paths or blossoms
  - What to do when a blossom is found
  - Algorithm and Correctness
- Next Lectures:
  - Edmonds-Gallai Decomposition
  - Stable matchings