1 Matchings in general graphs

The matching problem for bipartite graphs was considered in previous lectures. This problem is extended to cover matchings of non-bipartite graphs. The matching problem asks to find a matching \( M \subseteq E \) in a graph \( G = (V, E) \) such that \( |M| \) is maximized. A more general variation of this problem has weighted edges and maximizes the total weight of \( M \). Both are solvable in polynomial time, but we will only prove this for the former case.

This problem is in NP: provided a matching \( M \) such that \( |M| \geq k \), it can be verified in polynomial time that \( M \) is indeed a matching of size at least \( k \). We show this problem is also in co-NP by providing a polynomial time algorithm that shows that no matching \( M \) exists for which \( |M| \geq k \).

1.1 Size of a matching

As each vertex can be covered by at most one edge in \( M \) and every edge in \( M \) covers two vertices, \( |M| \leq \left\lfloor \frac{n}{2} \right\rfloor \) where \( n = |V| \). Therefore, if \( G \) has an odd number of vertices, at least one vertex of \( G \) is unmatched. Define \( o(G) \) to be the number of odd-sized components of \( G \). Because no edges occur between separate components of \( G \), it follows that \( |M| \leq \frac{1}{2}(n - o(G)) \).

We claim that \( M \) has at least \( o(G \setminus S) - |S| \) unmatched vertices for any \( S \subseteq V \). This holds because for every odd sized component in \( G \setminus S \), at least one vertex is unmatched; and because those can only be matched with vertices in \( S \), only \( |S| \) extra edges can be in a matching. It follows that for any \( S \subseteq V \):

\[
|M| \leq \frac{1}{2}(n - (o(G \setminus S) - |S|))
\]

And therefore:

\[
|M| \leq \frac{1}{2} \min_{S \subseteq V} \{n - o(G \setminus S) + |S|\}
\]

The Tutte-Berge formula states that for a maximum matching \( \max |M| \):

\[
\max |M| = \frac{1}{2} \min_{S \subseteq V} \{n - o(G \setminus S) + |S|\}
\]

Or, equivalently

\[
\min \#\text{unmatched vertices} = \max_{S \subseteq V} \{o(G \setminus S) + |S|\}
\]
1.2 Augmenting paths

Define an augmenting path to be a path in $G$ alternating between edges not in $M$ and in $M$ and starting and ending with unmatched vertices. For instance: If $\cdots \notin M$, $\circ \in M$ and $\bullet$ are unmatched, then $\circ \bullet \circ \bullet \cdots \circ$ is an augmenting path.

**Lemma 1** An augmenting path exists for a matching $M$ iff $M$ is not optimal.

**Proof:**
Suppose an augmenting path exists for $M$, then flipping the edges on that path results in a matching having more edges; so an optimal matching contains no augmenting path.
Conversely, if $M$ is not optimal, then a component in $G$ exists with at least two unmatched vertices and there must exist an augmenting path between two such vertices. $\square$

We will show how to detect whether an augmenting path exists.
If no exposed (= unmatched) vertex exists, no augmenting paths exist. Otherwise, we can pick such exposed vertex $v$ and construct a maximal $M$-alternating tree $T$ with root $v$. Each node of a tree is in some layer, determined by its distance from the root. An $M$-alternating tree is created by starting with root node $v$ (at layer 0) and applying the following steps until neither is possible:

1. Expand $T$ with an edge in $G$ between a node in an even layer of $T$ and a node not in $T$.
2. Expand $T$ with an edge in $M$ between a node in an odd layer of $T$ and a node not in $T$.

This may result in a tree like the one in Figure 1. As long as there exist exposed nodes that are not in such tree, a new tree can be created with such node as root. Eventually all exposed nodes are contained in some tree, and only a subgraph of $G$ (without any exposed vertices) is not contained in any tree. If any of those trees have an exposed node at an odd layer, an augmenting path exists between that node and the root of that tree.

**Lemma 2** If no exposed nodes occur on odd layers and no even-even edges (i.e. edges between nodes in even layers of $M$-alternating trees) exist, $M$ is optimal.

**Proof:**
Let $k$ be the number of exposed vertices in $G$. Then, if no exposed node occurs on odd layers, $k$ $M$-alternating trees are built. Let $O_i$ and $E_i$ be the number of edges on odd and even layers of the $i$-th tree, respectively. In each tree, there is one more even vertex than there are odd vertices because every node except the root is paired with some node on an adjacent layer by an edge in $M$, so $E_i - O_i = 1$ for all $i$. Now suppose no even-even edges exist and let $S$ be the set of vertices on the odd layers of the trees, then $G \setminus S$ is an independent set. By the Tutte-Berge formula, we get that the number of exposed vertices in the optimal matching is at least $k$:

$$\min \#\text{unmatched vertices} \geq o(G \setminus S) - |S| \geq \left(\sum_{i=1}^{k} E_i\right) - |S| \geq \sum_{i=1}^{k} E_i - O_i \geq \sum_{i=1}^{k} 1 \geq k$$

So $M$ must be optimal. $\square$
Figure 1: An $M$-alternating tree with an augmenting path.

Figure 2: A blossom caused by the red even-even edge.

Figure 3: The same blossom, now crushed.
Even-even edges between different maximal $M$-alternating trees cannot exist in $G$ as this would mean that the trees were not maximal in the first place. For the same reason, vertices from an even layer cannot connect to the nodes that do not occur not in any tree. Therefore, every even-even edge must lie between nodes of the same tree. These edges cause blossoms: cycles of odd length consisting of an even-even edge and part of a tree. An example of a blossom is shown in Figure 2. Every blossom $B \subseteq V$ can be crush into a single node. This means that a new graph $G' = (V', E')$ is created with $V' = \{b\} \cup V \setminus B$ and $E' = (E \setminus (B \times B)) \cup \{(v, b) \mid v \notin B \land (v, b) \in E\}$ where $b$ is a node summarizing the blossom in $G$. We use the notation $G/B$ to denote the graph $G'$ where blossom $B$ in graph $G$ is crushed. Furthermore, let $M/B$ denote the vertices in $M$ without the internal vertices of the blossom, specifically $M/B = (M \setminus (V \times B)) \cup \{(v, b) \mid v \notin B \land \exists w \in B \mid (v, w) \in M\}$. We show that when a blossom is crushed, maximal matchings are preserved:

**Lemma 3** Matching $M$ is optimal for $G$ iff $M/B$ is optimal for $G/B$ where $B$ is a blossom.

**Proof:**
We first show that if $M$ is optimal for $G$, then $M/B$ is also optimal for $G/B$. For this, suppose that $M/B$ is not optimal for $G/B$. Then there is an augmenting path $P$ in $G/B$. We show that an augmenting path also exists in $G$ and conclude that $M$ is not optimal for $G$ either. Note that any augmenting path starts at the root and ends in an odd leaf of the tree. Let $t$ be the node of $B$ closest to the root of the tree. If $P$ does not touch $B$ or only touches $B$ in $t$, then there is also an augmenting path in $G/B$. So assume $P$ enters $B$ at node $t$ and leaves $B$ at some other node. The path from the root of the tree to $t$ is of even length, so any augmenting path must continue from $t$ with an unmatched edge. There are two unmatched edges within $B$ adjacent to $t$ and $B$ is an odd length cycle. So no matter where $P$ leaves $B$, both an odd and an even length path from the root of the tree to the node where $P$ leaves $B$ exists such that its edges alternate between those in $M$ and those not in $M$. One of these paths can then be continued to mimic the path from $B$ to an odd leaf of the tree. Therefore an augmenting path always exists in $G$ and if $M$ is optimal for $G$, then $M/B$ is also optimal for $G/B$.

Next, we show that if $M$ is not optimal for $G$, then $M/B$ is not optimal for $G/B$. Suppose $M$ is not optimal for $G$, and let $P$ be an augmenting path in $G$. Every augmenting path starts at the root and ends in an odd leaf of an $M$-alternating tree. Again, if $P$ does not touch $B$, then $P$ is also an augmenting path in $G/B$. So assume $P$ touches $B$, then the edges of $P$ in $B$ can be removed from $P$ to obtain an augmenting path for $G/B$. This can easily be seen because $P$ leaves $B$ with an edge not in $M$ and starts with either the start of the augmenting path, or an edge in $M$. So it also holds that if $M/B$ is optimal for $G/B$, then $M$ is optimal for $G$.  

1.3 An algorithm for maximum matchings

Starting with an empty matching $M \leftarrow \emptyset$, we can repeatedly find an augmenting path $P$ and let $M \leftarrow (M \setminus P) \cup (P \setminus M)$ to obtain a better matching. As soon as no augmenting path exists, the optimal matching is stored in $M$.

Augmenting paths can be detected in $O(n^2)$ time by creating the $M$-alternating trees and crushing blossoms in the meantime. If such path exists, it can be extracted in $O(n^2)$ time using the construction of Lemma 3. Because a maximum matching has up to $n/2$ edges and every improvement adds at least one edge to the matching, the matching can only be improved $n/2$ times. Therefore, the full algorithm to find a maximal matching runs in $O(n^3)$ time.