# Edmonds Blossom Algorithm (Mid-Exam of AI2615)

Haoyu Zhen

June 22, 2022

## Problem 1

"$\Longrightarrow$": trivial.

If a $M$-augmenting path $p = v_1 v_2 \cdots v_n$ exists, then

$$\big(M - \{e | e \in p \wedge e \in M\}\big) \cup \{e | e \in p \wedge e \notin M\}$$

is a matching larger than $M$.

"$\Longleftarrow$":

Proof by contradiction. If $M$ is not a maximum matching and let $M'$ be a maximum one. Then we analyse the property of $M^\dagger \triangleq (M \cup M') - (M \cap M')$. For simplicity, we could count $\mathcal{M}$ as a graph (indeed, it is a set of edges).

1. $\forall e \in M^\dagger$, $e \in M$ XOR $e \in M'$.
2. The paths in $M^\dagger$ are alternating. I.e., for all vertex $v$ in $M^\dagger$, $v$ could be incident to **at most one** edge from $M$ and **at most one** edge from $M'$.
3. By 2., $\forall$ connected component $C$ in $M^\dagger$,$C$ is a path.
4. There exists a connected component $\mathcal{C}$ in $M^\dagger$, such that

   $$\big|\{e | e \in \mathcal{C} \wedge e \in M'\}\big| > \big|\{e | e \in \mathcal{C} \wedge e \in M\}\big|.$$

   (Beacuse $|M'| > |M|$.)
5. The both ends of $\mathcal{C}$ (in 4.) is in $M'$.

Finally, we construct a augmenting path $\mathcal{C}$ for $M$, which leads to a contradition.

## Problem 2

Contract the cycle $v_1 v_2 \cdots v_{2k+1} v_1$ to $u$.

"$\Longrightarrow$":

Proof by contradiction. Suppose that there exists a matching $M'$ of $G'$ such that $|M'| > |M - C|$.
Then there exists $C_{\text{new}}$ in $G$ such that $C_{\text{new}}$ meet no other edge in $M'$, because "$2k + 1 = 2 \times k + 1$".
LHS: the number of vertices in $C_{\text{new}}$. RHS: 1 represents $u$ and $2 \times k$ represents the matched vertices by $C_{\text{new}}$.
Then $M' \cup C_{\text{new}}$ is a larger matching of $G$. Contradiction!

"$\Longleftarrow$":

Similarly, assume that there exists a matching $M'$ for $G$ such that $|M'| > |M|$.
Then in $M'$, there exists a vertex $v$ who is not matched by $C$. Let this $v$ be $u$ and contract the cycle $C$. We get a larger matching of $G'$. Contradiction.

## Problem 3

M-alternating forests (MAF) exist by the fact that empty set is an MAF. Now we find a maximal one greedily:

---

**Algorithm 1** Find an maximal forest $\triangleq \mathcal{F}(G, M)$

---

**Input:** A graph $G$ whose matching is $M$

1: $F_{\max} \leftarrow \varnothing$
2: Select a root $r$ in $G$. $T \leftarrow (\{r\}, \varnothing)$.
3: **while** $\exists$ a alternating path $p$ begins and ends at outer vertices. **do**
4:     Insert $p$ into $T$
5:     Update $O_T$ and $I_T$     *#$O_T$ represents the outer vertices in $T$ ($I_T$: inner)*
6:     Delete all edges (not in $T$) which are incident with the inner vertex
7: **end while**
8: Generate a new graph $H$ and a matching $M_{\text{new}}$ by deleting the tree $T$ from $G$
9: Merge $T$ and $\mathcal{F}(H, M_{\text{new}}) \longrightarrow F_{\max}$     *#Recursion here*
10: **return** $F_{\max}$

---

**Correctness:** we just need to prove that $T$ is "maximal".

- The root of $T$ is the end points of a maximal augmenting path $\implies$ T will not be the sub-tree of other maximal alternating tree.
- There is no alternating path could be added into $T$.

**Complexity:** (We bound the complexity **LOOSELY**)

- Find the root: $\mathcal{O}(nm)$.
- Delete and insert (with the idea of charging): $\mathcal{O}(m)$.
- Find the alternating path (line 3; we use DFS here): $\mathcal{O}(nm)$.
- Update $O_T$ and $I_T$: $\mathcal{O}(n)$.

## Problem 4

The augmenting path should begin at a root $r_1$ and end at another one ($r_2$).
First I introduce my intuition: as figure 4 shown (in `edmonds.pdf`), we need to go **down** and then go **up** along several vertices. The conversion from down to up needs 2 consecutive outer vertices.
*Proof.*

- Define going up and down formally.

  Going down: if we are at a inner vertex, then we will choose an edge in $M$. For outer vertex, choose an dege whiche are not in $M$. Otherwise we are going up.

- Now we consider some possible cases.
- Inner $\to$ inner, "$\to$" is a edge in $M$. Not exists. Because this leads to a cernario that a vertex is matched twice.
- Inner $\to$ outer OR outer $\to$ inner. This does not change the upward and downward trend.
- Outer $\to$ outer, here "$\to$" is a edge not in $M$. Thus, at the second outer vertex, we need to choose an edge in $M$. We go up now!

$\square$

# Problem 5

Quite trivial!

If $u$ and $v$ belong to distinct components of $F$:

We construct the $M$-augmenting path directly: $r_1 \to u - v \to r_2$ where $u(v)$ belongs to the tree with root $r_1(r_2)$.

If $u$ and $v$ belong to the same component of $F$:

Obviously that the path in tree from $u$ to $v$ is even (by alternating property). Thus we have an odd cycle.

# Problem 6

My idea/intuition: I want to design a algorithm by which we could eradicate the augmenting path greedily and recursively. Why to contract the blossoms? Because we wan to find the path easily.

**Algorithm 2** Find the augmenting path $\triangleq \mathcal{P}(G, M)$

**Input:** A matching $M$ of the graph $G$
**Output:** Path $p$

1: Construct the forest $F(G, M)$
2: Find an edge whose endpoints are 2 outer vertices $u$ and $v$. If not, **return** $\varnothing$
3: **if** $u$ and $v$ belong to the same component of $F$ **then**
4:     **return** $r_1 \to u - v \to r_2$
5: **else**
6:     We could find an odd cycle $C$ with root $r$
7:     $v \leftarrow \min_{u \in V(C)} d(u, r)$     *#Find vertex whose distance to $r$ is minimum*
8:     $M' \leftarrow (M - p) \cup \{e | e \notin M \wedge e \in p\}$ where $p$ is the path from $r$ to $v$
9:     Now we could contract the cycle "safely" $\to$ $G'$     *#Satisfy the requirement in P2.*
10:     $P' \leftarrow \mathcal{P}(G', M')$
11:     "Interpret" $P'$:
    If $P'$ contains the vertex contracted from a cycle $C$, find an even path in $C$ to connect $P'$ to $P$
12:     **return** $P$
13: **end if**

Finally, we design an agorithm to find a maximum path. Forany given $G$,

**Algorithm 3** Find the maximum matching

**Input:**
**Output:**

$M \leftarrow \{e\}$ (we pick $e$ randomly)
**while** $p \leftarrow \mathcal{P}(G, M)$ is not an emptyset **do**
    Update $M$: eradicate this augmenting path. (Use the method in P1.)
**end while**
**return** $M$

**Soundness**:

1. Algo 3 holds naturally by P1. Also, $M$ is strictly increasing and bounded by $|E|$ (Algo 3 will terminate).

2. Now we analyse Algo 2.

   - Line 7-8: $v$ represents the vertex in cycle who meets other edges. So we update $M$ without changing its cardinality.

- Line 9: $\mathcal{P}(G', M')$ exists iff $\mathcal{P}(G, M)$ exists
- Line 10-11: convert the augmenting path $P'$ in $G'$ to a path $G$.

**Complexity**:

- While-loop in Algo 3 is bounded by $|E|$.
- Algo 2 is dominated by $\mathcal{F}(G, M)$.

Thus the complexity is $\mathcal{O}(n^2 m)$.

# Reference


1. https://www14.in.tum.de/lehre/2015WS/ea/split/sec-Augmenting-Paths-for-Matchings-single.pdf (for problem 1).
2. https://en.wikipedia.org/wiki/Blossom_algorithm (for problem 6).