18.438 Advanced Combinatorial Optimization September 15, 2009

Lecture 2

Lecturer: Michel X. Goemans *Scribes: Robert Kleinberg (2004), Alex Levin (2009)*

In this lecture, we will present Edmonds's algorithm for computing a maximum matching in a (not necessarily bipartite) graph G. We will later use the analysis of the algorithm to derive the Edmonds-Gallai Decomposition Theorem stated in the last lecture.

1 Recapitulation

Recall the following essential definitions and facts from the last lecture. A *matching* in an undirected graph G is a set of edges, no two of which share a common endpoint. Given a graph G and a matching M, a vertex is *matched* if it is the endpoint of an edge in M, *unmatched* otherwise; we will often designate the set of unmatched vertices by X. Given a graph G with matching M, an *M-alternating path* is a path whose edges are alternately in M and not in M. (Here we use *path* to mean a simple path, i.e. one with no repeated vertices. We'll refer to a non-simple path as a *walk*.) If both endpoints of an M-alternating path belong to the set X of unmatched vertices, it is called an *M-augmenting path*. Recall the following theorem from last time.

Theorem 1 *A matching M is of maximum size if and only if G contains no M-augmenting path.*

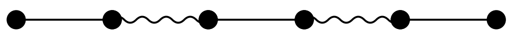


Figure 1: An M-augmenting path

2 Flowers, Stems, and Blossoms

The following construction is useful for finding M-augmenting paths. Given a graph $G = (V, E)$ with matching M; construct a directed graph $\hat{G} = (V, A)$ with the same vertex set as G, and with edge set determined by the rule that $(u, w) \in A$ if and only if there exists v with $(u, v) \in E \setminus M$ and $(v, w) \in M$. Observe that every M-augmenting path in G corresponds to a path in $\hat{G}$ that begins at a vertex in X and ends at a neighbor of X. However, the converse is not true, because an M-alternating walk may begin at a vertex in X and end at a neighbor of X, without being an M-augmenting path, if it contains an odd cycle. Figure 2 illustrates an example of such a walk. This motivates the following definition.

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

1. $v_0 \in X$.
2. $v_0, v_1, v_2, \ldots, v_{t-1}$ *are distinct.*
3. t *is odd.*

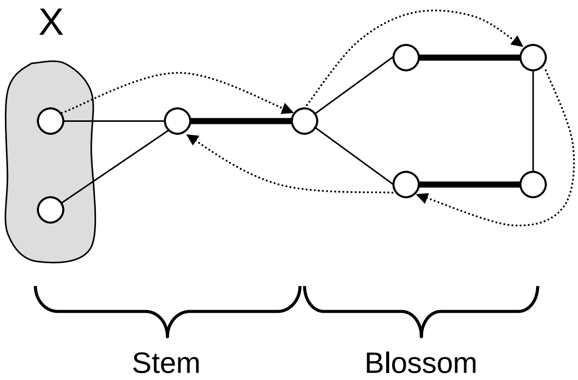


Figure 2: An M-flower. Note that the dashed edges represent edges of $\hat{G}$.

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*.*

Lemma 2 *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 $j < t$.*

Proof: If $v_0, v_1, \ldots, v_t$ are all distinct, P is an M-augmenting path. Otherwise, assume $v_i = v_j$, $i < j$, and let j be as small as possible, so that $v_0, v_1, \ldots, v_{j-1}$ are all distinct. We shall prove that $v_0, v_1, \ldots, v_j$ is an M-flower. Properties 1 and 2 of a flower are automatic, by construction. It cannot be the case that j is even, since then $(v_{j-1}, v_j) \in M$, which gives a contradiction in both of the following cases:

- $i = 0$: $(v_{j-1}, v_j) \in M$ contradicts $v_0 \in X$.
- $0 < i < j - 1$: $(v_{j-1}, v_j) \in M$ contradicts the fact that M is a matching, since v_i is already matched to a vertex other than v_{j-1}.

This proves that j is odd. It remains to show that i is even. Assume, by contradiction, that i is odd. This means that (v_i, v_{i+1}) and (v_j, v_{j+1}) are both edges in M. Then $v_{j+1} = v_{i+1}$ (since both are equal to the other endpoint of the unique matching edge containing $v_j = v_i$), and we may delete the cycle from P to obtain a shorter alternating walk from X to X. (See Figure 3.) $\square$

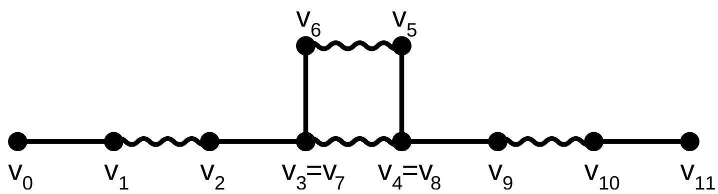


Figure 3: An alternating walk from X to X which can be shortened.

Given a flower $F = (v_0, v_1, \ldots, v_t)$ 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 belongs to an edge of M' except v_j.
2. M' agrees with M outside of F, i.e. $M \triangle M' \subseteq F$.
3. $|M'| = |M|$.

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

Whenever a graph G with matching M contains a blossom B, we may simplify the graph by *shrinking* B, a process which we now define.

Definition 2 (Shrinking a blossom) *Given a graph $G = (V, E)$ with a matching M and a blossom B, the* shrunk graph G/B *with matching M/B is defined as follows:*

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

where $E[B]$ denotes the set of edges within B, and b is a new vertex disjoint from V.

Observe that M/B is a matching in G, because the definition of a blossom precludes the possibility that M contains more than one edge with one but not both endpoints in B. Observe also that G/B may contain parallel edges between vertices, if G contains a vertex which is joined to B by more than one edge.

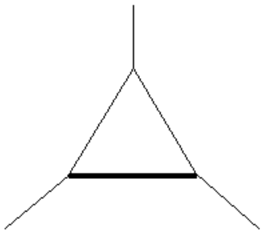
The relation between matchings in G and matchings in G/B is summarized by the following theorem.

Theorem 3 *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.*

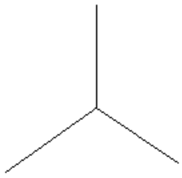
Proof: ($\Longrightarrow$) Suppose N is a matching in G/B larger than M/B. Pulling N back to a set of edges in G, it is incident to at most one vertex of B. Expand this to a matching N^+ in G by adjoining $\frac{1}{2}(|B|-1)$ edges within B to match every other vertex in B. Then we have $|N^+|-|N| = (|B|-1)/2$, while at the same time $|M| - |M/B| = (|B| - 1)/2$ (the latter follows because B is an M-blossom, so there are $(|B| - 1)/2$ edges of M in B; then M/B contains all the corresponding edges in M except those $(|B|-1)/2$). We conclude that $|N^+|$ exceeds $|M|$ by the same amount that $|N|$ exceeds $|M/B|$.

($\Longleftarrow$) If M is not of maximum size, then change it to another matching M', of equal cardinality, in which B is an entire flower. (If S is the stem of the flower whose blossom is B, then we may take $M' = M \triangle S$.) Note that M'/B is of the same cardinality as M/B, and b is an unmatched vertex of M'/B. Since M' is not a maximum-size matching in G, there exists an M'-augmenting path P. At least one of the endpoints of P is not in B. So number the vertices of P $u_0, u_1, \ldots, u_t$ with $u_0 \notin B$, and let u_i be the first node on P which is in B. (If there is no such node, then $u_i = u_t$.) This sub-path $u_0, u_1, \ldots, u_i$ is an (M'/B)-augmenting path in G/B. $\square$

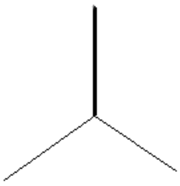
Note that if M is a matching in G that is not of maximum size, and B is blossom with respect to M, then M/B is not a maximum-size matching in G/B. If we find a maximum-size matching N in G/B, then the proof gives us a way to "unshrink" the blossom B in order to turn N into a matching N^+ of G of size larger than that of M. However, it is important to note that N^+ will not, in general, be a maximum-size matching of G, as the example in Figure 4 shows.



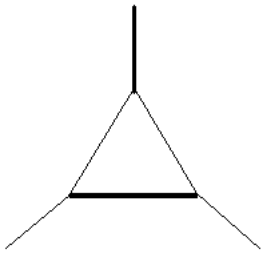
(a) The graph G with the matching M shown in bold



(b) The shrunk graph G/B with the matching M/B (the empty matching)



(c) The shrunk graph G/B with a maximum-size matching N



(d) The graph G with the matching N^+ obtained from N by the unshrinking process. Note that it is not a maximum-size matching

Figure 4: A maximum matching in the graph G/B does not necessarily pull back to a maximum matching in G.

3 A polynomial-time maximum matching algorithm

The algorithm for computing a maximum matching is specified in Figure 5.

The correctness of the algorithm is established by Lemma 2 and Theorem 3. The running time may be analyzed as follows. We can compute X and $\hat{G}$ in linear time, and can find $\hat{P}$ in linear time (by breadth-first search). Shrinking a blossom also takes linear time. We can only perform $O(n)$ such shrinkings before terminating or increasing $|M|$. The number of times we increase $|M|$ is $O(n)$. Therefore the algorithm's running time is $O(mn^2)$. With a little more work, this can be improved to $O(n^3)$. (See Schrijver's book.) The fastest known algorithm, due to Micali and Vazirani, runs in time $O(\sqrt{n}\,m)$.

```
M := ∅
X := {unmatched vertices} /* Initially all of V. */
Form the directed graph Ĝ.
while Ĝ contains a directed path P̂ from X to N(X)
    Find such a path P̂ of minimum length.
    P := the alternating path in G corresponding to P̂
    if P is an M-augmenting path,
        modify M by augmenting along P.
        Update X and construct Ĝ.
    else
        P contains a blossom B.
        Recursively find a maximum-size matching M′ in G/B.
        if |M′| = |M/B|    /* M is already a max matching. */
            return M       /* Done! */
        else               /* M can be enlarged */
            Unshrink M′ as in the proof of Theorem 3,
              to obtain a matching in G of size > |M|.
              /* It is not necessarily maximal */
            Update M and X and construct the graph Ĝ.
end
```

Figure 5: Algorithm for computing a maximum matching