Lecture notes for "Analysis of Algorithms": **Maximum matching in general graphs**

Lecturer: *Uri Zwick* *

Spring 2006

Abstract

We present Edmonds' *blossom shrinking* algorithm for finding a maximum cardinality matching in a general graph. En route, we obtain an efficient algorithm for finding a minimum vertex cover in a bipartite graph and show that its size is equal to the size of the maximum matching in the graph. We also show that the size of a maximum matching in a general graph is equal to the size of a minimum odd cover of the graph.

1 The maximum matching problem

Let $G = (V, E)$ be an undirected graph. A set $M \subseteq E$ is a *matching* if no two edges in M touch each other or, in other words, if the degree of every vertex in the subgraph (V, M) is at most 1. A vertex v is *matched* by M if there is an edge of M that touches v. Otherwise, v in *unmatched*. In the *maximum matching problem* we are asked to find a matching M in the graph $G = (V, E)$ of maximum size.

As we have seen, the maximum matching problem in *bipartite* graphs can be easily reduced to a maximum flow problem which could then be solved in $O(m\sqrt{n})$ time. The maximum matching problem in general, not necessarily bipartite, graphs is more challenging. We present here a classical algorithm of Edmonds [Edm65] for solving the problem.

2 Alternating and augmenting paths

If A and B are sets, we let $A \oplus B = (A - B) \cup (B - A)$ be their *symmetric difference*. The following lemma is obvious.

*School of Computer Science, Tel Aviv University, Tel Aviv 69978, Israel. E–mail: zwick@cs.tau.ac.il

Lemma 2.1 *Let $G = (V, E)$ be an undirected graph and let M_1 and M_2 be matchings in G. Then, the subgraph $(V, M_1 \oplus M_2)$ is composed of isolated vertices, alternating paths and alternating cycles that alternatingly contain edges from M_1 and M_2.*

An *alternating path* P with respect to a matching M is a simple path in G such that among any two consecutive edges on P exactly one belongs to M. An *alternating cycle* C with respect to a matching M is defined similarly. The following lemma is again easily verified.

Lemma 2.2 *If P is an alternating path with respect to M and each endpoint of P is either unmatched by M or matched by the edge of P touching it, then $M \oplus P$ is also a matching.*

An *augmenting path* P with respect to a matching M is an alternating path that starts and ends in *unmatched* vertices. Note that an augmenting path in necessarily of odd length and that the number of edges on P that do not belong to M is by one larger than the number of edges that do belong to M.

Theorem 2.3 *Let $G = (V, E)$ be an undirected graph and let M be a matching in G. Then, M is a maximum matching in G if and only if there are no augmenting paths with respect to M in G.*

Proof: If P is an augmenting path with respect to M, then $M \oplus P$ is also a matching and $|M \oplus P| > |M|$, so M is not a maximum cardinality matching of G.

Conversely, suppose that M is not a maximum matching in G. Let M' be a matching with $|M'| > |M|$. By Lemma 2.1 $M \oplus M'$ is composed of alternating paths and cycles. All the alternating cycles are of even length and contain the same number of edges from M and M'. At least one of the alternating paths must contain more edges from M' and this path is then an augmenting path with respect to M. □

Theorem 2.3 suggests the following simple algorithm for finding a maximum matching. Start with some initial matching M, possibly the empty one. As long as there is an augmenting path P with respect to M, augment M using P and repeat. All that remains, therefore, is to devise a procedure for finding augmenting paths, if they exist.

3 Alternating trees

In this section we try a natural approach for finding an augmenting path with respect to matching M, if one exists. The approach will succeed, however, only in bipartite graphs.

We construct a forest of *alternating trees*. An alternating tree is a tree whose root is an unmatched vertex. All paths in an alternating tree are alternating paths with respect to M. If we manage to add an unmatched vertex, other than the root, to an alternating tree, we have identified an

augmenting path. Vertices at the even levels of alternating tree are labeled EVEN while vertices at the odd levels are labeled ODD. Vertices that are currently in no alternating tree are unlabeled.

Initially all nodes are unlabeled. We choose an unmatched vertex r, label it EVEN, make it the root of a new tree and declare it to be unexplored. As long as there is an unexplored EVEN vertex in the tree, we choose one such vertex u and examine all the edges touching it. For every edge $(u, v) \in E$, if v is unmatched, we have found an augmenting path. If v is unlabeled and matched, we let v' be the vertex matched to v, i.e., $(v, v') \in M$. We add v and v' to the tree and label them ODD and EVEN, respectively, and declare v' to be unexplored. (Check that if v is unlabeled then v' is also unlabeled.) If v is already labeled ODD, we do nothing. (We simply found an alternative odd length alternating path from an unmatched vertex to v. Note that v may be part of a previously constructed tree.) The remaining case is that v is already labeled EVEN. Vertex v cannot belong to a previously constructed tree, as the edge (u, v) would have been explored from u. Thus, v belongs to the same tree as u. As both u and v are EVEN, an odd cycle was found. This can only happen in non-bipartite graphs. Dealing with this last case poses a non-trivial challenge that we face in Section 5. Let us assume, for the time being, that this 'problematic' case does not occur. When all the edges of u are examined, u is declared to be explored. If there are no more unexplored EVEN vertices in the tree, a new unmatched vertex is chosen and a new alternating tree is constructed. The process ends when there are no more unmatched vertices.

4 Minimum vertex cover in bipartite graphs

Let $G = (V, E)$ be an undirected graph. A set $C \subseteq V$ is said to be a *vertex cover* if and only if $e \cap C \neq \phi$, for every $e \in E$. In other words, C is a vertex cover if and only if every edge $(u, v) \in E$ has at least one of its endpoints in C. In the *minimum vertex cover* problem we are given an undirected graph $G = (V, E)$ and asked to find a vertex cover of minimum size. For general graphs, the problem is NP-complete. (Note that C is a vertex cover if and only if $V - C$ is an independent set.) In this section we show, perhaps surprisingly, that the minimum vertex cover problem in bipartite graphs can be efficiently solved. It is solved, in fact, by the algorithm sketched in the previous section.

If the input graph $G = (V, E)$ is bipartite, then the problematic case encountered in the previous section does not occur and we obtain a linear time algorithm for finding an augmenting path with respect to M, if one exists.

Lemma 4.1 *If $G = (V, E)$ is a bipartite graph and M is not a maximum matching in G, then the algorithm sketched in the previous section finds an augmenting path with respect to M.*

Suppose now that M is a maximum matching in the bipartite graph $G = (V, E)$. Suppose that $V = A \cup B$, where $A \cap B = \phi$ and that all edges connect vertices in A with vertices of B. As every augmenting path in G connects an unmatched vertex in A with an unmatched vertex in B,

it is enough to construct alternating trees rooted at vertices of A. Let F be the forest obtained. (Edmonds [Edm65] calls F an *Hungarian forest.*) The following lemma is easily verified:

Lemma 4.2 *The forest F obtained has the following properties:*

1. *All EVEN vertices in F belong to A.*
2. *All ODD vertices in F belong to B.*
3. *If $u \in F$ is a leaf, then $u \in A$.*
4. *If $(u, v) \in E$ and $u, v \in F$, then $u \in B$ or $v \in B$.*
5. *If $(u, v) \in E$ and $u \in F$ and $v \notin F$, then $u \in B$.*
6. *If $(u, v) \in E$ and $u, v \notin F$ and $u \in A$, then u is matched.*

We next show that the size of a maximum matching in a bipartite graph is equal to the size of the minimum vertex cover in the graph.

Theorem 4.3 *Let $G = (V, E)$ be a bipartite graph. If M is a maximum matching in G and C is a minimum vertex cover, then $|M| = |C|$.*

Proof: It is clear that $|M| \leq |C|$, as any vertex cover must contain at least one endpoint of each edge of the matching M.

Suppose now that M is a maximum matching in $G = (V, E)$. Suppose that $V = A \cup B$, where $A \cap B = \phi$ and that all edges connect vertices in A with vertices of B. Let F be the forest of alternating trees rooted at the unmatched vertices in A as above. We construct a vertex cover C' such that $|C'| = |M|$ as follows. Let $M' = M \cap F$ be the edges of M that are contained in F, and let $M'' = M - F$ be the edges of M that are not contained in F. The vertex cover C' is obtained by taking the endpoints of the edges of M' that belong to B and the endpoints of the edges of M'' that belong to A. Clearly $|C'| = |M|$. It follows easily from Lemma 4.2 that C' is indeed a vertex cover, as required. □

The proof of Theorem 4.3 gives us an efficient algorithm for finding a minimum vertex cover of a bipartite graph $G = (V, E)$: Find a maximum matching in G, construct an (Hungarian) forest F and then a minimum vertex cover C. We note that the last two steps require only linear time.

5 Blossom shrinking

We now return to the problem of finding a maximum matching in non-bipartite graphs.

A *flower* with respect to a matching M is composed of a *stem*, which is an alternating path of even length from an unmatched vertex r, called the *root*, to vertex b, and an 'alternating' cycle of odd length that passes through b, called a *blossom*. The last edge on the stem belongs to M. The two

edges of the blossom touching b are not in M. Other than that, every second edge on the blossom belongs to M. The vertex b is said to be the *base* of the blossom.

When the algorithm of Section 3 finds an edge $(u, v) \in E$ such that both u and v are EVEN, it, in fact, finds a flower. Edmonds suggests *contracting* the blossom B of the flower into a new vertex, labeling this new vertex EVEN, and continuing. We let G_B be the graph obtained by contracting B, and $M_B = M - B$ the matching induced by M in this new graph. We usually let b denote the vertex in G_B obtained by contracting B. (Note that b also denotes the base of the stem in the original graph.)

Lemma 5.1 *Let $G = (V, E)$ be an undirected graph and let M be a matching in G. Let B be a blossom in G. If G_B contains an augmenting path with respect to M_B, then G contains an augmenting path with respect to M.*

Proof: Let P be an augmenting path in G_B with respect to M_B. If P does not pass through b, then P is also an augmenting path in G with respect to M, and we are done.

Suppose, therefore, that P does pass through b. Assume, at first, that P starts (or ends) at b. Let (b, c) be the first edge on B and let P_c be the part of P that starts at c. Clearly $(b, c) \notin M_B$. Also, there is a vertex $v \in B$ such that $(v, c) \in E$ and $(v, c) \notin M$. In the blossom B, we can find an even length alternating path Q from b to v. This path ends with an edge of M. The path $Q, (v, c), P_c$ is then an augmenting path in G with respect to M, as required.

Finally, assume that b is not the first or last vertex on P. Let $(a, b), (b, c) \in P$ be edge edges of P that touch b. Assume that $(a, b) \in M_B$ and $(b, c) \notin M_B$. Let P_a be the part of P up to a, and let P_c be the part of P from c. As $(a, b) \in M_b$, the edge (a, b) must also be present in the original graph, as the only edge of the matching that enters a blossom enters it at its base. As before, there is a vertex $v \in B$ such that $(v, c) \in E$ and $(v, c) \notin M$. We can again find an even alternating path Q in the blossom from b to v. The path $P_a, (a, b), Q, (v, c), P_c$ is then an augmenting path in G with respect to M, as required. □

We shall see later that the converse to Lemma 5.1 also holds. That is, if G contains an augmenting path with respect to a matching M and G_B is obtained by shrinking a blossom B, then G_B contains an augmenting path with respect to M_B.

This suggests the following algorithm for finding an augmenting path. Start to construct the alternating trees, as in Section 3. If a blossom is encountered, contract it and continue. As at most n blossoms may be encountered while looking for an augmenting path, a very naive implementation of this algorithm, which constructs all the alternating trees from scratch whenever a blossom is encountered, requires only $O(mn)$ time per augmentation, giving an $O(mn^2)$-time maximum matching algorithm. A slightly less naive, but still fairly straightforward, finds an augmenting path in $O(n^2)$ or even $O(m\alpha(m, n))$ time, yielding $O(n^3)$-time and $O(mn\,\alpha(m, n))$-time maximum matching algorithms. For a very clear description of this more efficient algorithm, complete with C++ code, see Section 7.7 of LEDA book [MN99]. Various versions of the algorithm are also described in the books [Law76], [Tar83] and [AMO93].

The maximum matching problem is general graph can actually be solved in $O(m\sqrt{n})$ time (see [MV80], [GT91], [Vaz94]) but this is beyond the scope of this course.

6 Correctness of the blossom shrinking algorithm

In this section we give a direct proof to the converse of Lemma 5.1.

Lemma 6.1 *Let $G = (V, E)$ be an undirected graph and let M be a matching in G. Let B be a blossom in G. If G contains an augmenting path with respect to M, then G_B contains an augmenting path with respect to M_B.*

Proof: To be given in class ... □

Lemma 6.1 would also follow from the results of the next section.

7 Odd vertex covers

Let $G = (V, E)$ be an undirected graph. An *odd vertex cover* of G is composed of a subset $C \subseteq V$ of vertices and a collection of subsets $B_1, B_2, \ldots, B_k \subseteq V$ such that for every edge $(u, v) \in E$ either $u \in C$ or $v \in C$, or there is an index $1 \leq i \leq k$ for which $u, v \in B_i$. The *size* of such a cover is defined to be $|C| + \sum_{i=1}^{k} \lfloor \frac{|B_i|}{2} \rfloor$. We may assume that the sets $B_1, B_2, \ldots, B_k$ are of odd size.

Theorem 7.1 *Let $G = (V, E)$ be an undirected graph. The size of a maximum matching in G is equal to the size of a minimum odd vertex cover of G.*

Proof: To be given in class ... □

References

[AMO93] R.K. Ahuja, T.L. Magnanti, and J.B. Orlin. *Network flows – Theory, algorithms and applications.* Prentice Hall, 1993.

[Edm65] J. Edmonds. Paths, trees, and flowers. *Canadian Journal of Mathematics*, 17:449–467, 1965.

[GT91] H.N. Gabow and R.E. Tarjan. Faster scaling algorithms for general graph matching problems. *Journal of the ACM*, 38(4):815–853, 1991.

[Law76] E.L. Lawler. *Combinatorial optimization: networks and matroids.* Holt, Rinehart and Winston, 1976.

[MN99] K. Mehlhorn and S. Näher. *LEDA – A platform for combinatorial and geometric computing*. Cambridge University Press, 1999.

[MV80] S. Micali and V. V. Vazirani. An $O(\sqrt{|V|} \cdot |E|)$ algorithm for finding maximum matching in general graphs. In *Proc. of 21st FOCS*, pages 17–27, 1980.

[Tar83] R.E. Tarjan. *Data structures and network algorithms*. SIAM, 1983.

[Vaz94] V.V. Vazirani. A theory of alternating paths and blossoms for proving correctness of the $O(\sqrt{V}E)$ general graph maximum matching algorithm. *Combinatorica*, 14(1):71–109, 1994.