IEOR 8100: "Matchings" **Fall 2012**

Lecture 4: Matching Algorithms for Bipartite Graphs

Professor: Clifford Stein *Scribes: Jelena Marašević*

Let $G = (V, E)$ be a bipartite graph, and let $n = |V|$, $m = |E|$. Recall that the linear program for finding a maximum matching on G, and its dual (which finds a vertex cover) are given by:

$$\begin{aligned} \text{maximize} \quad & \sum_{e \in E} x_e \\ \text{subject to} \quad & \sum_{e \in \delta(v)} x_e \leq 1, \quad \forall v \in V \\ & x_e \geq 0, \quad \forall e \in E \end{aligned} \qquad \begin{aligned} \text{minimize} \quad & \sum_{v \in V} y_v \\ \text{subject to} \quad & y_u + y_v \geq 1, \quad \forall (u, v) \in E \\ & y_v \geq 0, \quad \forall v \in V \end{aligned}$$

Optimal solution to the primal LP is a $0-1$ m-length vector, in which $x_e = 1$ is assigned to those edges that belong to the maximum matching M, and $x_e = 0$ is assigned to all the remaining edges. Similarly, optimal solution to the dual LP assigns $y_v = 1$ to those vertices that constitute the vertex cover C, and 0 otherwise.

4.1 Basic algorithm for bipartite matching

Before delving into the algorithm for bipartite matching, let us define several terms that will be used in the rest of this notes. Suppose we are given a bipartite graph $G = (V, E)$ and a matching M (not necessarily maximal). We say that, with respect to the matching M:

- $v \in V$ is a **free vertex**, if no edge from M is incident to v (i.e, if v is not matched).
- P is an **alternating path**, if P is a path in G, and for every pair of subsequent edges on P it is true that one of them is in M and another one is not.
- P is an **augmenting path**, if P is an alternating path with a special property that its start and end vertex are free.

For example, in graph G shown in the Fig 4.1, with all the edges from the matching M being marked bold, vertices a_1, b_1, a_4, b_4, a_5 and b_5 are free, $\{a_1, b_1\}$ and $\{b_2, a_2, b_3\}$ are two examples of alternating paths, and $\{a_1, b_2, a_2, b_3, a_3, b_4\}$ is one example of an augmenting path.

Main idea for the algorithm that finds a maximum matching on bipartite graphs comes from the following fact: *Given some matching M and an augmenting path P, $M' = M \oplus P$ is a matching with $|M'| = |M| + 1$.* Here, '$\oplus$' denotes the symmetric difference set operation (everything that belongs to both sets individually, but doesn't belong to their intersection). This is not hard to see if we observe that every augmenting path

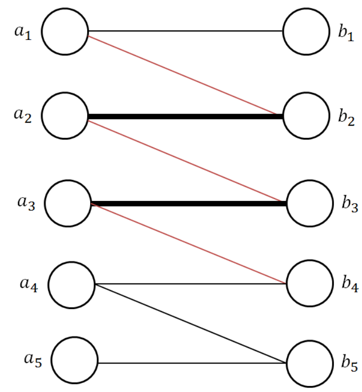

Figure 4.1: A matching on a bipartite graph.

P, as it is alternating and it starts and ends with a free vertex, must be odd length and must have one edge more in its subset of unmatched edges ($P \backslash M$) than in its subset of matched edges ($P \cap M$). For example, on a graph shown in Fig. 4.1, a better matching can be obtained by taking red edges instead of bold edges. In this example, P is given by red and bold edges, and operation of replacing the old matching M by a new one $M' = M \oplus P$ is called the **augmentation over path** P.

At this point the idea for finding a maximum matching becomes apparent–start with any matching in G (including the empty one), and repeatedly find an augmenting path (if there exists one) and augment over it, until there are no augmenting paths left. To argue that this approach is correct, we must prove the following theorem:

Theorem 4.1 *For a given bipartite graph G, a matching M is maximum if and only if G has no augmenting paths with respect to M.*

Proof: ($\Rightarrow$) We prove this by contrapositive, i.e., by showing that if G has an augmenting path, then M is not a maximum matching. But this holds true due to the fact we argued above, as if there is some augmenting path P, we can take $M' = M \oplus P$ and obtain a matching of size $|M| + 1$, so M cannot be a maximum matching.

($\Leftarrow$)We prove this part also by contrapositive, starting with the assumption that M is a non-maximum matching, and then showing that this implies the existence of some augmenting path. Let M^* denote some maximum matching in G. Observe the symmetric difference between M and M^*, namely, $Q = M \oplus M^*$. As both M and M^* are matchings, each vertex in Q can have degree not greater than 2, because it can be adjacent to at most one edge from each matching. If a vertex has degree equal to 2, then it is adjacent to one edge from M and one edge from M^*. Therefore, all the cycles and paths are alternating (note that cycles can only be even-length). Now notice that as $|M^*| > |M|$ and $Q = M^* \oplus M = (M^* \cup M) \backslash (M^* \cap M)$, it is also true that $|Q \cap M^*| > |Q \cap M|$, meaning that the graph induced on G by Q must have objects that contribute more to M^* than to M. As we decomposed our graph into even-length cycles and paths, those objects can only be odd-length paths that start and end with edges from M^*. Therefore, there must be at least one alternating path, starting and ending with an edge from M^*. But this type of path is an augmenting path in G with respect to the matching M, which closes our proof. ■

Pseudocode for bipartite matching can now be written as:

BIPARTITE-MATCHING(G)
$M = \emptyset$
repeat
  $P =$(AUGMENTING-PATH(G, M))
  $M = M \oplus P$
until $P = \emptyset$

The question that has still remained unanswered is how to find augmenting paths. Before answering it, let us introduce some additional notation that will be used throughout. Split the set of vertices V into sets A and B, such that: $A \cup B = V$, $A \cap B = \emptyset$ and $\forall e = (u, v) \in E : u \in A \Leftrightarrow v \in B$, that is, A and B are "left" and "right" set of vertices of a bipartite graph G.

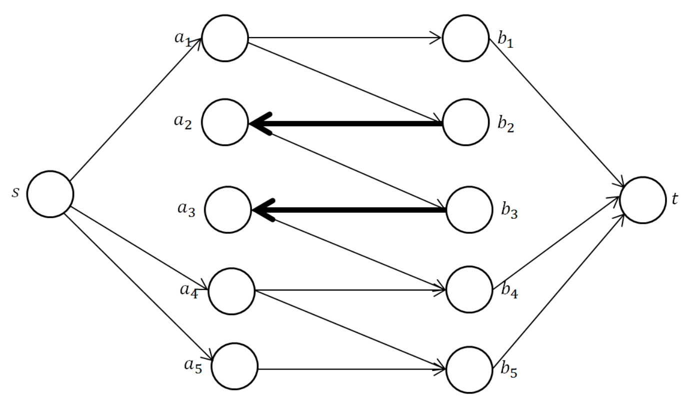

Figure 4.2: Finding an augmenting path.

Direct all edges in G, taking direction from A to B for all unmatched edges, and from B to A for all matched edges. Now all the directed paths in G are alternating, and a free vertex in B can be reached from a free vertex in A only via augmenting path. These paths can be found by performing a bread-first-search (BFS) on a modified graph that adds a source vertex s, a sink vertex t, and directed edges between s and free vertices in A, as well as directed edges between free vertices in B and t, as in Fig. 4.2. BFS runs in $O(m)$ time, which yields a total running time of $O(mn)$ for BIPARTITE-MATCHING, as matching can be of size at most $\frac{n}{2} = O(n)$.

AUGMENTING-PATH(G, M)
Direct unmatched edges $A \rightarrow B$, matched $B \rightarrow A$
Add s, t and connect them to free vertices in A and B, respectively
Run BFS on G and return a shortest path P from s to t
return $P \backslash \{s, t\}$

4.2 Vertex cover

Having solved maximum bipartite matching, next interesting problem would be to find its dual, a vertex cover, from it. It turns that this is possible to do in an efficient manner, and pseudocode below describes the algorithm for finding it.

Bipartite-Vertex-Cover(G, M)
$L = \emptyset$
Run dept-first-search from each of the free vertices in A
Add each vertex discovered by DFS in previous step to L
return $C^* = (A \backslash L) \cup (B \cap L)$

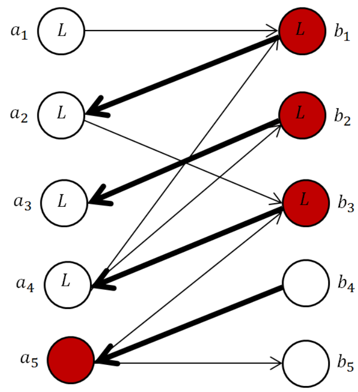

Figure 4.3: Vertex cover.

To argue the correctness of this algorithm, we need to prove the following claim:

Claim 4.2 *If M^* is a maximum matching on G, and L is the set of all the vertices reachable from A with respect to M^*, then $C^* = (A \backslash L) \cup (B \cap L)$ is a vertex cover in G, and $|C^*| = |M^*|$.*

Proof: Assume that C^* is not a vertex cover. Then there must exist at least one edge $(a, b) \in E$ not incident to any vertex in C^*. As $C^* = (A \backslash L) \cup (B \cap L)$, this edge cannot have its endpoints in neither $(A \backslash L)$ nor in $(B \cap L)$. Therefore, it must be $a \in A \cap L$ and $b \in B \backslash L$. Observe the two possible cases:

1. $(a, b) \in M^*$. As $a \in A \cap L$, it must have been reached in a dept-first-search. But a can only be reached via matched edge, so b must have also been added to L, which is a contradiction.
2. $(a, b) \notin M^*$. As $a \in L$ by the assumption, and (a, b) is unmatched, b is reachable from a, so b gets added to L by the algorithm, which is a contradiction, as $b \notin L$.

So far we have proved that C^* is a vertex cover; now we need to show that is also a minimum one. To do so, it is enough to show that $|C^*| \geq |M^*|$, as $|C^*| \leq |M^*|$ holds by the weak duality. Consider the following observations:

i) No vertex in $A \backslash L$ is free, because each DFS starts from a free vertex in A, and each free vertex from A must be added to L by the algorithm.

ii) No vertex in $B \cap L$ is free, otherwise we would have found an augmenting path, which is impossible, as the matching is maximum.

iii) No edge (a, b), such that $a \in A \backslash L$, $b \in B \cap L$, can belong to the maximum matching M^*. If there was such edge, then the search that adds b to L would also discover a and add it to L, which is a contradiction, as $a \in A \backslash L$.

By i) and ii), each vertex in C^* is incident to a matched edge. By iii), no two different vertices in C^* can be incident to the same matched edge. Therefore, C^* is at least as big as M^*, which concludes the proof. ■