

Lecture 4: Matching in Bipartite Graphs

A graph $G(V, E)$ is **bipartite** if we can partition V into two sets A and B such that all edges are incident to one vertex in A and one vertex in B (i.e. for all $(u, v) \in E$ either $u \in A, v \in B$ or $u \in B, v \in A$). We usually write $G(A, B, E)$.

A **matching** M is a set of edges such that every vertex is incident to at most one edge in M . In other words, it is a set of “independent” edges that share no endpoints in common. A **perfect matching** is a matching such that saturates all the vertices. Matching M is a perfect matching iff $|A| = |B| = |M|$. M is a **maximal** matching if we cannot greedily increase the size of M , i.e. $\forall e \in E, M \cup e$ is not a matching. M is a **maximum** matching if there are no possible matchings of greater size.

Given a bipartite graph $G(A, B, E)$, how can we tell if it admits a perfect matching? Philip Hall in 1935 was gave a simple necessary and sufficient condition:

Theorem 1 (Hall’s Marriage Theorem) *Let $G(A, B, E)$ be a bipartite graph such that $|A| = |B| = n$. G has a perfect matching if and only if*

$$\forall S \subseteq A, |S| \leq |N(S)| \quad (1)$$

where $N(S)$ is the neighborhood of S , i.e. $N(S) = \{v \in B \mid \exists u \in S : (u, v) \in E\}$.

Proof:

“ \Rightarrow ”: It’s east to see that if $\exists S \subset A$ such that $|S| > |N(S)|$, G cannot have a perfect matching.

“ \Leftarrow ”: By induction on the size of the graphs. The case $n = 1$ is trivial. We split the induction step up into two cases:

Case 1: $\forall S \subset A, S \neq A, |S| < |N(S)|$

Pick an arbitrary vertex $u \in A$ and match it to one of its neighbors v . Remove (u, v) and all edges incident to them from G . Let $G_1(A_1, B_1, E_1)$ be the resulting graph. Since we had a strict inequality, $\forall S \subset A_1, |S| \leq |N_{G_1}(S)|$, and the conditions of the marriage theorem hold for G_1 . Applying the inductive hypothesis to G_1 and adding on (u, v) yields a perfect matching in G .

Case 2: $\exists S \subset A, S \neq A, |S| = |N(S)|$

Define $G_1(A_1, B_1, E_1)$ such that $A_1 = S, B_1 = N_{G_1}(S)$ and E_1 is the set of edges between A_1 and B_1 in G . Define $G_2(A_2, B_2, E)$ such that $A_2 = A - S, B_2 = B - N_G(S)$ and define E_2 similarly to E_1 .

- Let $S_1 \subset A_1$. By construction $N_{G_1}(S_1) = N_G(S_1)$. Then $|S_1| \leq |N_G(S_1)| = |N_{G_1}(S_1)|$.
- Let $S_2 \subset A_2$. If $|S_2| > |N_{G_2}(S_2)|$, then $|A_1 \cup S_2| > |N_G(A_1 \cup S_2)|$ by construction of A_1 , which contradicts Hall's condition for G . Thus $|S_2| \leq |N_{G_2}(S_2)|$.

Both subgraphs G_1 and G_2 therefore satisfy Hall's condition, so by the induction hypothesis G has a perfect matching. ■

This proof gives a nice combinatorial result, but is unfortunately non-constructive. It gives no hint as to how to efficiently find a perfect matching or to determine whether a perfect matching exists. In order to develop an polynomial time algorithm for maximum matching, we first develop an alternate characterization.

Definition: Alternating/Augmenting path with respect to a matching M .

Let M be a matching in a bipartite graph G . A path $P = u_1, v_1, u_2, \dots, v_{q-1}, u_q, v_q$ is called an alternating (or augmenting) path with respect to M iff

- u_1 and v_q are unmatched in M .
- $(v_i, u_{i+1}) \in M$ for all $i \in \{1, \dots, q-1\}$.

This leads to a nice characterization of maximum matchings:

Theorem 2 *Berge's theorem*

Let $G(A, B, E)$ be a bipartite graph and M be a matching. Then M is maximum if and only if there is no alternating path in G with respect to M .

Proof:

“ \Rightarrow ”: Let's assume M is maximum and that P is an alternating path with respect to M . Since every other edge of P is in M , we can match u_i to v_i for all $i \in \{1, \dots, q\}$, in place of the previously matched edges in M . This gives us a matching M' of cardinality $|M| + 1$, which contradicts that M is maximum.

“ \Leftarrow ”: Assume matching M is such that no alternating path in G exist with respect to M , and there exists a maximum matching M' in G such that $|M'| > |M|$.

From the first part of this proof, we know there are no alternating paths with respect to M' . Now consider the multigraph \tilde{G} with vertex set V and an edge set $M \cup M'$ (if an edge is in both M and M' , we put it in twice). Since M and M' are both matchings, vertex v in \tilde{G} has degree of 0 (if it was unmatched in both matchings), 1 (if it was matched only in one of the two matchings) or 2 (if it was matched in both matchings).

This means that all the connected components of \tilde{G} are either cycles or paths. It is easy to see that if a connected component is a cycle, given that M and M' are both matchings, it has an even number of edges. And hence the same number of edges from M and M' . If a component is a path of even length we have the same situation. A path of odd length would imply an alternating path for either M or M' in G (since the first and last edges in the path both belong to either M or M'). Thus, there are no paths of odd length proving that $|M| = |M'|$, hence M is maximum. ■

To see how Berge's theorem leads directly to an efficient algorithm for finding maximum matchings in bipartite graphs we define the concept of an almost alternating path.

Definition: Almost alternating paths.

Let M be a matching in a bipartite graph G . We say that a path $Q = u_1, v_1, u_2, \dots, v_{q-1}, u_q$ is an almost alternating path with respect to M iff

- u_1 is unmatched in M .
- $(v_i, u_{i+1}) \in M$ for all $i \in \{1, \dots, q-1\}$.

A single unmatched vertex by itself can be considered an almost alternating path.

From the definition of an almost alternating path, it is obvious that such paths start and end in the same set of vertices (A or B). Given an almost alternating path $Q = u_1, v_1, u_2, \dots, v_{q-1}, u_q$, note that if $v \in N(u_q)$ and v is not matched, then $P = Q \cup \{v\}$ is an alternating path. Using this fact we can devise an algorithm, known as the Hungarian algorithm, to find a maximum matching.

The high-level idea behind the Hungarian algorithm is to search the set of vertices reachable from a starting unmatched vertex through almost alternating paths. The goal is to find an almost alternating path that can be extended to an alternating path. As there can be an exponential number of almost alternating paths starting from a given vertex, a brute force search would not work well in general. The key observation is that there are only polynomial many almost alternating path *endpoints*, and these are what are important for finding alternating points. So the Hungarian algorithm finds alternating paths simply by searching among potential almost alternating path endpoints and augments the matching using these paths.