MTL 776: Graph Algorithms

Lecture : Matching in Graphs

Course Coordinator: Prof. B. S. Panda

Note: *The note is based on the lectures taken in the class. It is a draft version and may contain errors. It is being uploaded to help the students to refer for Minor I. Corrected version will be uploaded later.*

Definition 1.1 *A subset $M \subseteq E$ of a graph $G = (V, E)$ is called a* **matching** *if no two edges in M are adjacent. A vertex $v \in V$ is M***-saturated** *if v is incident on an edge in M; otherwise, v is called M***-unsaturated.** *The size, $|M|$, of the matching is the number of edges in M. A matching M is a* **maximal matching** *if no proper superset of M is a matching of G. A matching with maximum cardinality among all matchings of G is called a maximum matching of G.*

In this lecture, we will be interested in finding a maximum matching of a graph. The concept of augmenting path with respect to a matching is a key concept in characterizing the maximum matching.

Definition 1.2 *A path P is called an M-alternating path if the edges of P are alternately in M and not in M. An M-alternating path that begins and ends at unsaturated vertices is called an M-augmenting path.*

Theorem 1.3 *Let M_1 and M_2 be any two matchings in $G = (V, E)$. Let $H = (V, (M_1 \setminus M_2) \cup (M_2 \setminus M_1))$. Then each connected component of H is (i) an isolated vertex, (ii) an even cycle whose edges are alternately in M_1 and M_2, or (iii) a nontrivial path whose edges are alternately in M_1 and in M_2. Furthermore, each end vertex of the path is either M_1-saturated or M_2-saturated but not both.*

Proof: Let $v \in V(H)$. Since at most one edge of M_1 and at most one edge of M_2 can be incident on v, $d_H(v) \leq 2$. So each connected component is either an isolated vertex or a cycle or a path. Since no two edges of M_1 are incident on a vertex and no two edges of M_2 are incident on a vertex, the edges of each cycle and path of H are alternately in M_1 and M_2. So if a connected component of H is a cycle, then it is an even cycle whose edges are alternately in M_1 and M_2.

Consider a component of H that is a path P. Let x be an end vertex of P and xy be the edge in P. So, $xy \in (M_1 \setminus M_2) \cup (M_2 \setminus M_1)$. Since $(M_1 \setminus M_2) \cap (M_2 \setminus M_1) = \emptyset$, $e = xy$ lies in exactly one of the sets $(M_1 \setminus M_2)$ and $(M_2 \setminus M_1)$. Without loss of generality, assume that $e = xy \in (M_1 \setminus M_2)$. So $e \in M_1$ but $e \notin M_2$. So x is M_1-saturated. If x is M_2-saturated, then there exits an edge, say xz, such that $xz \in M_2$. So $d_H(x) \geq 2$. Since P is a connected component and x is an end vertex of P, $d_H(x) = 1$. This is a contradiction. So x is M_1-saturated but not M_2-saturated.

■

We are now in a position to characterize a maximum matching in a graph.

Theorem 1.4 *A matching M of a graph $G = (V, E)$ is a maximum matching if and only if G has no M-augmenting path.*

Proof: Necessity: Suppose that M is a maximum matching of G. We need to prove that G has no M-augmenting path. To the contrary, assume that, G has an M-augmenting path, say, P. Note that P is of odd length. Let $M' = (M \setminus (M \cap E(P))) \cup (E(P) \setminus M)$. That is, M' is formed from M by removing the edges of M that are present in $E(P)$ and adding all the edges of $E(P)$ which are not in M. Clearly $|M'| = |M| + 1$. Also M' is a matching. This contradicts the fact that M is a maximum matching. Hence, G can not have any M-augmenting path.

Sufficiency: Suppose that there is no M-augmenting path in G. We need to prove that M is a maximum matching. If possible, M is not a maximum matching. So there exits a matching, say M', such that $|M'| > |M|$. Consider the graph $H = (V, (M \setminus M') \cup (M' \setminus M))$. Since H contains more edges from M' than from M, H contains a connected component that contains more edges from M' than from M. So by Lemma 1.3, H contains an odd path that contains more edges from M' than from M and both end vertices of P are M-unsaturated. So, P is an M-augmenting path. This is a contradiction to the assumption that G has no M-augmenting path. Hence, M must be a maximum matching. ■

Now we are in a position to propose an algorithm to find a maximum matching in a graph.

Algorithm 1: Algorithm Maximum Matching

Input : A graph $G = (V, E)$.
Output: A matching M of G.
$M = \{e\}$ for some $e \in E(G)$;
while *there is an M-augmenting path in G* **do**
    Find an M-augmenting path P of G;
    $M = (M \setminus E(P)) \cup (E(P) \setminus M)$;
Output(M);

Theorem 1.5 *Let M be a matching of a graph $G = (V, E)$, v be an M-unsaturated vertex, and M_1 be a matching obtained by augmenting M along an M-augmenting path P. If there is no M-augmenting path with v as an end vertex, then there is no M_1-augmenting path with v as an end vertex.*

So in the above algorithm, if at i^{th} stage, there is no M-augmenting path starting at an M-unsaturated vertex v, then in subsequent stages, the unsaturated vertex v will not be considered for finding an augmenting path. v will remain as unmatched vertex with respect to the final maximum matching M found by the algorithm.

1.1 Maximum Matching in Bipartite Graphs

In this section, we describe two polynomial time algorithms to find an M-augmenting path in a bipartite graph.

1.1.1 finding M-augmenting Path

Let M be a matching of a bipartite graph $G = (X \cup Y, E)$. Direct all edges in G, taking direction from X to Y for all unmatched edges, and from Y to X for all matched edges. Now G has an M-augmenting path if and only if there is a directed path from an M-unsaturated vertex $x \in X$ to an M-unsaturated vertex y in Y. These augmenting paths can be found by performing a breadth first-search (BFS) on a modified graph that adds a source vertex s, a sink vertex t, and directed edges between s and M-unsaturated vertices in X, as well as directed edges between M-unsaturated vertices in Y and t. BFS runs in O(m) time, which yields a directed path from s to t if such a path exists. An M-augmenting path can be obtained from such a path easily.

1.1.2 Alternating tree Method

Start with a vertex v that is M-unsaturated to see if there is any M-augmenting path with v as an end vertex. If there is no M-augmenting path with v as an end vertex, then v will not be considered in subsequent steps. Using BFS, we construct a rooted tree T rooted at v such that all the paths in T with v as an end vertex are M-alternating. The tree T is called an alternating tree. We construct the tree T as follows. If there is any M-unsaturated vertex u adjacent to v, then v,u is an M-augmenting path and we stop. Otherwise, all the adjacent vertices of v are M-saturated. Place v at level 0 and all the vertices adjacent to v, say $u_1, u_2, \ldots, u_k$, at level 1. We add the edges $vu_i, 1 \leq i \leq k$ to the tree T and set $P(u_i) = v, 1 \leq i \leq k$. Since u_i is M-saturated for $1 \leq i \leq k$, there exist vertices, say $x_1, x_2, \ldots, x_k$ such that $u_i x_i \in M, 1 \leq i \leq k$. Place the vertices $x_1, x_2, \ldots, x_k$ in level 2 and add the edges $u_i x_i, 1 \leq i \leq k$ in T and set $P(x_i) = u_i, 1 \leq i \leq k$. The second level of the alternating tree is constructed. Assume that the alternating tree has been constructed through level r, where r is even and no M-augmenting path starting at v has been found so far. For every vertex x at level r, check whether any of the neighbors of x is M-unsaturated. If so, we have found an M-augmenting path. Otherwise, for each vertex x at level r of T, check each neighbor y of x. If y is already in the tree, then ignore. Otherwise, add xy to the tree and place y in level $r+1$. Find the vertex z such that $yz \in M$. Add z in level $r+2$ and add the edge yz in T. Continue this process till either an M-augmenting path is detected or we cannot add any level to the tree. In the first case, we augment M using the M-augmenting path starting at v. In the second case, we construct an alternating tree rooted at another M-unsaturated vertex, say w. If no M-unsaturated vertex remains, then M is a maximum matching.

We give below the pseudo code of finding an M-augmenting path in a bipartite graph.

Algorithm 2: Algorithm Augmenting path

```
Input  : A bipartite graph G = (X ∪ Y, E) and a matching M.
Output: An M-augmenting path P if exists.
if there is no M-unsaturated vertex in X then
    Output(There is no M-augmenting path in G);
else
    for each v ∈ V(G) do
        Mark[v]=Unvisited;
        P[v]=-1;
        Saturated[v]=1 if v ∈ X and v is M-unsaturated;
    for each unsaturated vertex x in X do
        if Mark[x]== Unvisited then
            enqueue(Q,v);
        while Q ≠ empty do
            v = Dequeue(Q);
            Mark[v]=Visited;
            for each unvisited vertex u ∈ N(v) do
                if u is M-unsaturated then
                    output (the path from u to root); Exit;
                else
                    P[u] = v; P[w] = u, where uw ∈ M;
                    Enqueue(Q, w);
    Output(there is no M-augmenting path);
```

1.2 Vertex Cover

A subset $S \subseteq V$ is called a vertex cover if every edge in G is incident on a vertex in S. A vertex cover of minimum cardinality in G is called minimum cardinality vertex cover.

Proposition 1.6 *If M is matching in G and C is a vertex cover in G, then $|M| \leq |C|$.*

Proof: Let M be a matching and C be a vertex cover of G. Since no two edges of M are adjacent, no two edges of M will be incident on the same vertex of C. Since C is a vertex cover, C contains at least $|M|$ vertices which are incident on the edges of M. Hence, $|M| \leq |C|$. ■

Consider $G = C_3$, the cycle of length three. Note that the maximum cardinality matching in G is of cardinality 1 and the the minimum cardinality of a minimum vertex cover is of cardinality two. So, the converse of the above proposition is not true in general. However, if G is a bipartite graph, then the converse of the above proposition is always true. This statement is known as Konig-Egervery theorem as this result was proved independently by Konig and Egervery in 1931.

Theorem 1.7 *If G is a bipartite graph, then the cardinality of a maximum matching of G is equal to the cardinality of a minimum vertex cover of G.*

Proof: Let M be a maximum matching of $G = (X \cup Y, E)$. Let U be the set of all M-unsaturated vertices in X. Let $S = \{x \in (X \cup Y) | x$ is reachable by an M-alternating path from a vertex in $U\}$.

First note that for each edge $xy \in M$, $x \in S$ if and only if $y \in S$. We construct a set C consisting of exactly one of the end vertices of each of the edges in M as follows. For an edge $xy \in M$ with $x \in X$ and $y \in Y$, we take the vertex y in C if $x, y \in S$. Otherwise, we take the vertex x in C. So $|C| = |M|$.

We next show that C is a vertex cover. Let $ab \in E(G)$ with $a \in X$. If $a \notin S$, then a is not an M-unsaturated vertex. Let ax be an edge of M. So $x \notin S$. Hence $a \in C$ by construction of C. Next suppose that $a \in S$. If $a \in U$, then $b \in S$ and $bx \in M$. Hence $b \in C$. If $a \notin U$, then $ab \in M$ and $b \in S$. So $b \in C$. Hence C is a vertex cover.

■

When can a bipartite graph $G = (X \cup Y, E)$ with $|X| \leq |Y|$ has a matching that saturates all the vertices in X? The following theorem known as Hall's Theorem answers this question and was obtained by Hall in 1935.

Theorem 1.8 *Let $G = (X \cup Y, E)$ be a bipartite graph such that $|X| \leq |Y|$. If $|N(S)| \geq |S|$ for all $S \subset X$, the G has a matching that saturates all the vertices of X.*

Proof: Necessity: Suppose that G has a matching M that saturates every vertex in X. Let $S \subset X$. Now each vertex of S is adjacent to an edge of M and no two edges in M are adjacent. So $|N(S)| \geq |S|$.

Sufficiency: Assume that $|N(S)| \geq |S|$ for all $S \subset X$. Let M be a maximum matching of G. If possible M does not saturate some vertex in X. Let $u \in X$ be an M-unsaturated vertex. Let S be the set of all vertex of G which are reachable by an M-alternating path from u. Let $U = X \cap S$ and $V = Y \cap S$. Now each vertex in $U \setminus \{u\}$ is matched by an edge, say uv, such that $v \in V$. So $|V| = |U| - 1$ and $V \subseteq N(U)$. For each $v \in N(U)$, G contains an M-alternating u-v path. So $N(U) \subseteq V$. Hence, $N(U) = V$ and $|N(U)| = |V| = |U| - 1 < |U|$. This is a contradiction to the fact that $|N(S)| \geq |S|$ for all $S \subset X$. Hence, M must saturates all the vertices of X. ■

1.3 Maximum cost matching in Weighted bipartite graphs

In this section, we will discuss the problem of finding a maximum cost matching in a weighted bipartite graph. A matching M of $G = (V, E)$ is called a perfect matching if it saturates all the vertices of G. First we will show that the maximum cost matching in weighted bipartite graph can be reduced to the problem of finding a maximum cost perfect matching in a weighted complete bipartite graph having same number of vertices in each part.

Given a weighted bipartite graph $G = (X, Y, E)$ with $w : E \to R$, we construct a weighted complete bipartite graph $G' = (X', Y', E')$ and $w' : E' \to R$ as follows.

Without loss of generality assume that $|X| \geq |Y|$. Let $k = |X| - |Y|$. Let $X' = X$ and $Y' = Y \cup \{z_1, z_2, \ldots, z_k\}$. Consider the complete bipartite graph $G' = (X', Y', E')$, where $E' = \{xy | x \in X', y \in Y'\}$. Let $w' : E' \to R$ be such that $w'(xy) = 0$ if $x \in X', y \in Y'$ but $xy \notin E$ and $w'(xy) = w(xy)$ if $xy \in E$. The following result shows that **MCMWBG (Maximum Cost Matching in Weighted Bipartite Graphs)** reduces to **MCPMWCBG (Maximum Cost Perfect Matching in Weighted Complete Bipartite Graphs)**.

Theorem 1.9 *M' is a maximum cost perfect matching in the weighted complete bipartite graph $G' = (X', Y', E')$ with $w : E' \to R$, then $M' \cap E$ is a maximum cost matching in $G = (X, Y, E)$ with $w : E \to R$. Furthermore, cost(M') = cost(M).*

In view of the above theorem, we next concentrate on the **MCPMWCBG** problem.

We discuss the famous **Kuhn-Munkres** algorithm, which is also known as Hungarian Method, for **MCPMWCBG** problem.

Definition 1.10 *A function $l : X \cup Y \to R$ is called a feasible vertex labeling of a weighted complete bipartite graph $(G = (X, Y, E), w)$ if $l(u) + l(v) \geq w(uv)$ for all $uv \in E$.*

Note that $l(u) = \max\{w(uv) | v \in Y\}$ and $l(v) = 0, v \in Y$ is a fvl(feasible vertex labeling).

Let $E_l = \{uv \in E | l(u) + l(v) = w(uv)\}$ and $H_l = (X, Y, E_l)$ with the weight function w, and G_l is the corresponding unweighted graph of H_l.

Theorem 1.11 *Let l be a feasible vertex labeling of a weighted complete bipartite graph $(G = (X, Y, E), w)$. If H_l contains a perfect matching M', them M' is a maximum cost matching in G.*

Proof: Now $cost(M') = \sum_{e \in M'} w(e) = \sum_{x \in X \cup Y} l(x)$. Let M be any matching in G. Now $cost(M) = \sum_{xy \in M} w(xy) \leq \sum_{x \in X \cup Y} l(x) = cost(M')$. Hence, M' is a maximum cost matching in G. ∎

Next, we present the maximum cost perfect matching algorithm.

Theorem 1.12 *l' defined in the* **Algorithm Perfect Matching** *is a feasible vertex labeling and $G_{l'}$ contains M' and T.*

1.4 Maximum matching in General graphs

Finally, we concentrate on finding a maximum cardinality matching in general graphs. Recall that a matching M of G is a maximum matching if and only if there is no M-augmenting path in G. We used alternating

Algorithm 3: Algorithm Maximum Cost Perfect Matching

Input : A weighted complete bipartite graph $(G = (X, Y, E), w)$.
Output: A maximum cost perfect matching M of G.
Find a feasible vertex labeling l of G;
while *true* **do**
  Construct G_l;
  Find a maximum cardinality matching M' of G_l;
  if *M' is a perfect matching of G_l* **then**
    output (M);
    break;
  Let T be an M'-alternating tree rooted at an M'-unsaturated vertex $x \in X$;
  Find $ml = \min\{l(u) + l(v) - w(uv) | u \in X \cap V(T), v \in Y \setminus V(T)\}$;
  $l'(u) = l(u) - ml$ if $u \in X \cap V(T)$, $l'(v) = l(v) + ml$ if $v \in Y \cap V(T)$, $l'(v) = l(v)$ otherwise;
  $l = l'$

tree method for finding an M-augmenting path in G if such a Path exists. However, the same method does not work for general graphs. This is because of the presence of a structure called a flower which we will define below.

Definition 1.13 *A flower with respect to a matching M is a stem S, which is an M-alternating even path from an M-unsaturated vertex, say r to a vertex b, together with a blossom B, which is an M-alternating cycle of odd length containing the only vertex b from S. So the edges of B incident on b are edges not from M. The vertex b is called the base of the blossom B.*

If x and y are two vertices adjacent to b, then consider the graph G consisting of the flower F whose stem is $S = r, a, b$, and whose blossom is $B = b, x, x_1, y_1, y, b$, together with the edge y_2y and the matching $M = \{ab, xx_i, yy_1\}$. Now the alternating tree method started at r will not be able to detect the M-augmenting path $r, a, b, x, x_1, y_y, y, y_2$, but will be able to detect the flower F.

We will modify the alternating tree method to find a maximum matching in general graphs. If a flower is found, the blossom of the flower is contracted to a single vertex to obtained new graph G' with the property that G has an M-augmenting path if and only if G' has an M-augmenting path. We continue the alternating tree method in G'. If we get an M-alternating path, then we augment M. If we find a flower, then we contract the blossom of the flower to get another graph G''. If no M-augmenting path is found and no flower is found in G', the we declare that M^* which is obtained from M by adding the edges of the blossoms in an appropriate way, is a maximum matching in G.