Lecture Outline 3
Topics in Graph Algorithms (CSCI5320-19S)

CAI Leizhen
Department of Computer Science and Engineering
The Chinese University of Hong Kong
lcai@cse.cuhk.edu.hk

January 22, 2019

Keywords: Matching, perfect matching, Berge's theorem, Hall's theorem, Tutte's theorem, König and Egervary's theorem, and Hungarian algorithm.

1. **Definitions**: Let $G = (V, E)$ be an undirected graph. $M \subseteq E$ is a *matching* if no two edges in M share vertices. For an edge $uv \in M$, u and v are *matched* under M or *M-matched*. For a vertex v, if v is incident with an edge of M, then v is *M-saturated* and M *saturates* v; otherwise v is *M-unsaturated*.

   A matching is a *perfect matching* if it saturates every vertex, and a *maximum matching* if it has the maximum size (maximum weight for weighted graphs).

   An *M-alternating path* is a path whose edges alternate between edges in M and edges not in M, and such a path is an *M-augmenting path* if it starts and ends with M-unsaturated vertices.

2. **Berge's Theorem** (1957). *A matching M of G is a maximum matching iff G has no M-augmenting path.*

   Proof. The sufficiency follows from the fact that an M-augmenting path can be used to obtain a new matching with one more edge. Conversely, suppose that M' is a larger matching in G and let $M \Delta M'$ be the symmetric difference of M' and M. Since the maximum degree of $G[M \Delta M']$ is two, $G[M \Delta M']$ is a disjoint union of M-alternating paths and cycles, and one M-alternating path is an M-augmenting path as $|M'| > |M|$. ∎

3. **Hall's Theorem** (1935). A bipartite graph $G = (X, Y; E)$ admits a matching saturating every vertex in X iff for every $S \subseteq X$, $|N(S)| \geq |S|$.

   Proof. The condition is obviously necessary and we need only show the sufficiency. For this purpose, we show that any maximum matching M of G saturates all vertices in X. Suppose to the contrary that X contains an M-unsaturated vertex x. Let $X' \subseteq X$, $Y' \subseteq Y$ be vertices reachable from x by M-alternating paths. Since M is a maximum matching, G has no M-augmenting path and hence any M-alternating from x terminates at a vertex inside X. It follows that $N(X') = Y'$. On the other hand, every vertex in

Y' is M-saturated and thus M matches Y' with $X' - x$, implying $|Y'| = |X'| - 1$. But then $|N(X')| = |X'| - 1 < |X'|$, contradicting the assumption of the theorem. Therefore X contains no M-unsaturated vertex. ∎

Algorithm: We now discuss a polynomial-time algorithm for determining whether a bipartite graph $G = (X, Y; E)$ admits a matching that saturates all vertices in X. Let M be a matching in G, and $u \in X$ an M-unsaturated vertex.

To find an M-augmenting path from u, we grow an *M-alternating tree* T, which is a tree with root u such that for every vertex v in T, the (u, v)-path in T is M-alternating.

Initially, T contains vertex u only. At any stage, either (Case 1) T contains an M-unsaturated vertex $y \in Y$, or (Case 2) all vertices in T except u are M-saturated.

For Case 1, the (u, y)-path P in T is an M-augmenting path, and we can use it to obtain a larger matching.

For Case 2, let $V_X = V(T) \cap X$ and $V_Y = V(T) \cap Y$. Then $V_Y \subseteq N(V_X)$. If $V_Y = N(V_X)$, then $|N(V_X)| = |V_X| - 1$ and G has no required matching by Hall's theorem. Otherwise, V_Y is strictly inside $N(V_X)$, i.e., some vertex $x \in V_X$ is adjacent to a vertex $y \in N(V_X) - V_Y$ and $xy \notin M$.

If y is M-unsaturated, then we can grow T by adding edge xy to reach Case 1. Otherwise y is M-saturated with mate z, and we can grow T by adding edges xy and yz and reach Case 2 again.

Question: What to do if X has more than one M-unsaturated vertices?

4. **Theorem** (König, Egervary 1931). For a bipartite graph G, the size of a maximum matching equals the size of a minimum vertex cover.

   Proof. Obviously, the size of a minimum vertex cover is an upper bound of the size of a maximum matching. Let $X' \cup Y'$, where $X' \subseteq X$ and $Y' \subseteq Y$, be a minimum vertex cover of G. Consider induced subgraph $G_1 = G[X' \cup (Y - Y')]$. For any $S \subseteq X'$, $|N_{G_1}(S)| \geq |S|$ by the assumption that $X' \cup Y'$ is a minimum vertex cover of G. By Hall's theorem, G_1 admits a matching M_1 saturating every vertex in X'. Similarly, $G_2 = G[(X - X') \cup Y']$ admits a matching M_2 saturating every vertex in Y'. It follows that $M_1 \cup M_2$ is a matching of size $|X' \cup Y'|$. ∎

5. **Tutte's Theorem** (1947) A graph $G = (V, E)$ admits a perfect matching iff for every $S \subseteq V$, $o(G - S) \leq |S|$, where $o(G - S)$ denotes the number of odd components in $G - S$.

   Note that an odd component is a component with an odd number of vertices.

6. **Weighted cover**

   Let G be a complete bipartite graph $K_{n,n}$ with vertex set $X \cup Y$ and weight $w(xy) \in N$ for each edge xy.

   Weighted cover f of G: an assignment $f : X \cup Y \rightarrow N$ such that $f(x) + f(y) \geq w(xy)$ for each edge xy.

   Equality subgraph G_f for f: formed by edges xy of G satisfying $f(x) + f(y) = w(xy)$.

   Cost $c(f)$ of weighted cover f: $c(f) = \sum_{v \in X \cup Y} f(v)$.

Lemma. For every perfect matching M and weighted cover f of G, we have $c(f) \geq w(M)$. Furthermore, $c(f) = w(M)$ iff M consists of edges xy with $f(x) + f(y) = w(xy)$, and in such case M is a maximum-weight matching and f a minimum cost weighted cover.

7. **Hungarian algorithm** (Kuhn 1955 and Munkres 1957)

Find a maximum-weight matching and a minimum cost weighted cover in a weighted complete bipartite graph $G = K_{n,n}$ in $O(n^4)$ time.

Idea: Maintain a weighted cover f, iteratively reduce the cost of cover until the equality subgraph G_f contains a perfect matching. Initially, $f(x) = \max_{y \in Y} w(xy)\}$ for each $x \in X$ and $f(y) = 0$ for each $y \in Y$.

Iteration: $G \leftarrow G_f$;
Find a maximum matching M and minimum vertex cover V' in G;
If M is a perfect matching, DONE.
Otherwise $X' \leftarrow X \cap V'$, $Y' \leftarrow Y \cap V'$,
and $\epsilon \leftarrow \min\{f(x) + f(y) - w(xy) : x \in X - X', y \in Y - Y'\}$;
For each $x \in X - X'$, $f(x) \leftarrow f(x) - \epsilon$;
For each $y \in Y'$, $f(y) \leftarrow f(y) + \epsilon$.

8. **Applications**

Chinese Postman (Guan 1962): Find a closed walk of minimum weight that visits all edges of a weighted graph G.

The problem is equivalent to the following: duplicate some edges of G to form an Eulerian multigraph G' so that the total weight of duplicated edges is minimized. An Euler tour in G' gives us a solution for G.

Question: Why does an optimal walk visit each edge at most twice?

Determine edges to be duplicated: Let E' be the set of duplicated edges in the multigraph G'. Then E' can be decomposed into edge-disjoint paths in G whose ends are odd vertices of G. Furthermore, these edge-disjoint paths are shortest paths between odd vertices (why?).

Construct a weighted complete graph G^* from G whose vertices are odd vertices in G. The weight of edge uv in G^* equals the (u, v)-distance in G. Note that the number of vertices in G^* are even (why?)

A minimum weight perfect matching of G^* yields shortest paths that correspond to duplicated edges of G'.

Question: Why are these shortest paths edge-disjoint?

Rooted subtree isomorphism: Given two rooted trees S and T, determine whether S is a rooted subtree of T, i.e., there is a subgraph isomorphism that maps the root of S into the root of T.

Let $\{S_i\}$ (resp., $\{T_j\}$) be subtrees of S (resp., T) whose roots are children of roots of S (resp., T). For each S_i and T_j, recursively determine whether S_i is a rooted subtree of T_j.

Construct a bipartite graph G whose vertices are all S_i's and T_j's, and whose edges S_iT_j indicate that S_i is a rooted subtree of T_j. Then G has a matching saturating all vertices S_i in G iff S is a rooted subtree of T.

Growing path: Alice and Bob alternately choose a distinct vertex to grow a path in a graph G. The person who cannot make a move loses.

The second player has a winning strategy when G contains a perfect matching.

Question: Does the first player have a winning strategy when G has no perfect matching?