Matchings in Graphs

Lecturer: *Nitin Saurabh*
Scribe: *Meena Mahajan*

Meeting: 9
11 March

Today we look at the status of another matchings-related question: decide whether a graph has exactly one perfect matching. We call this the **Unique Perfect Matching** problem, UPM in short. We will show:

- UPM is in P.
- For bipartite graphs, UPM is in deterministic NC.
- UPM, even restricted to bipartite graphs, is hard for non-deterministic logspace NLOG.

1 UPM is in P

The naive strategy for checking if a graph has exactly one perfect matching works:

Algorithm 1 Test-UPM-1

1: Input: A graph G.
2: Find a maximum matching M in G using any P-time algorithm.
3: M is not perfect, then reject.
4: **for** each $e \in M$ **do**
5: **if** $G \setminus \{e\}$ has a perfect matching **then**
6: reject.
7: **end if**
8: **end for**
9: Otherwise for each $e \in M$, e is essential in any perfect matching in G. So in fact M is the only perfect matching. Accept.

This algorithm requires $1 + n/2$ invocations of a perfect matching algorithm. There are more efficient ways of detecting UPM, but we will not discuss them here.

2 Bipartite UPM in NC

The algorithm of Section 1 is sequential because it calls a maximum/perfect matching routine in steps 1 and 3, and this routine is not known to have an NC algorithm in general. For special cases where maximum matching is known to be in NC, the method can be implemented in NC, since the $n/2$ calls to the routine in step 3 can be performed in parallel. For bipartite graphs, we do not know of an NC algorithm for maximum matching. Nevertheless, we can

design an NC algorithm for bipartite UPM, using a different approach. This algorithm is due to Kozen, vazirani and Vazirani [KVV85]. Thus, testing uniqueness is apparently easier than testing existence, since we are allowed to (in fact, expected to) also reject if there are multiple perfect matchings.

The NC algorithm uses as a subroutine a procedure that constructs in NC a perfect matching in a graph G, given the promise that G has exactly one perfect matching. Note that under this promise, we have already seen an NC algorithm for constructing the matching: namely, the Grigoriev-Karpinski algorithm from last class. (There, we assumed that the number of perfect matchings was bounded by a polynomial in n. Here, the polynomial is $n^0 = 1$.) Since the promise here is more stringent, there is a simpler algorithm due to Rabin and Vazirani [RV89], described below.

Let G be the given bipartite graph, on $2n$ vertices. Let the parition of the vertices be $U \cup V$, with $|U| = |V| = n$ and $E \subseteq U \times V$. Consider the $n \times n$ bipartite adjacency matrix B of G: rows are indexed by U and columns by V, and $B_{i,j} = 1$ if (u_i, v_j) is an edge, 0 otherwise. (If the vertices $U \cup V$ are ordered so that U appears before V, then the standard adjacency matrix A of G has the form $\begin{bmatrix} 0 & B \\ B^T & 0 \end{bmatrix}$. All the relevant information is contained in B.)

Every matching in G contributes a term, either $+1$ or -1, to $\det(B)$. Hence if there is just one matching, M, then $\det(B) = \pm 1$. And M can be retrieved as follows: For each edge $e = (u_i, v_j)$, if $e \in M$, then $G_{i,j} = G \setminus \{u_i, v_j\}$ has exactly one perfect matching and so its bipartite adjacency matrix $B_{i,j}$ has determinant ± 1. If $e \notin M$, then $G_{i,j}$ has no perfect matching (any perfect matching here could be extended to a perfect matching M' in G using e) and so $\det(B_{i,j}) = 0$. Thus we have the following algorithm:

Algorithm 2 Construct-UPM

1: Input: The bipartite adjacency matrix B of a bipartite graph G known to have exactly one perfect matching.
2: **for all** $e = (u_i, v_j)$ **do**
3: **if** $\det(B_{i,j} \in \{-1, +1\})$ **then**
4: $c_e = 1$
5: **else**
6: $c_e = 0$
7: **end if**
8: **end for**
9: $E' = \{e \in E \mid c_e = 1\}$ is the perfect matching.

Now we use this algorithm to test uniqueness. Note that the algorithm can be run on any bipartite graph, but without the UPM promise, the edge set E' returned may not be a perfect matching at all. If E' is not a perfect matching, then certainly G is not a UPM graph. Even if E' is a perfect matching, it may not be unique. We can try to check this using steps 4-8 of Algorithm 1. But that requires an NC algorithm for existence. Instead we do the following: create a weighted graph H where edges in E' have weight x (x is a variable)

and other edges have weight 1. Let D be the bipartite adjacency matrix of H. Then the terms of $\det(D)$ are no longer ± 1. A matching using k edges from E' gives rise to a term of the form $\pm x^k$. Thus terms of $\det(D)$ are of the form $\pm x^k$ for $0 \leq k \leq n$. Clearly, if E' is the only perfect matching in G, then the only term in $\det(D)$ is $\pm x^n$. The tricky part is showing the converse.

Lemma 1 $\det(D) = \pm x^n \iff E'$ *is the unique perfect matching in* G.

Proof: The $\Leftarrow$ direction is obvious; we consider the $\Rightarrow$ direction. Without loss of generality, assume that the vertices in U and V are renumbered so that $E' = \{(u_1, v_1), (u_2, v_2), \ldots, (u_n, v_n)\}$, giving rise to the identity permutation. So $\det(D) = x^n$. Assume, to the contrary, that G has perfect matchings other than E'. We need to show that the terms of these matchings do not cancel each other in the determinant.

The disjoint union (symmetric difference) of any two perfect matchings is a collection of vertex-disjoint even alternating cycles, each of length at least 4. Let $\mathcal{M}$ be the set of all perfect matchings other than E' in G. Let k be the smallest number such that for some $M \in \mathcal{M}$, $M \oplus E'$ has a cycle of length $2k$.

Claim 2 *The terms of all perfect matchings agreeing with* E' *in exactly* $n - k$ *edges have the same sign.*

Proof: Let M be a perfect matching agreeing with E' in exactly $n-k$ edges. Then $M \oplus E'$ has alternating cycles with $2k$ edges, and by our choice of k, all these edges must lie on a single cycle. Let the cycle be as follows, starting from the E' edge: $u_{i_1} v_{i_1} u_{i_2} v_{i_2} \ldots u_{i_k} v_{i_k}$. Renumber the vertices so that $i_1 = 1, i_2 = 2, \ldots, i_k = k$. (Renumbering does not change the sign of a permutation; the sign depends only on the cycle structure of the permutation.) Now we see that

$$M = \{(u_2, v_1), (u_3, v_2), \ldots, (u_k, v_{k-1}), (u_1, v_k)\} \cup \{(u_j, v_j) \mid k < j \leq n\}$$

Thus the permutation corresponding to M has the cycle decomposition

$$\langle\ (k, k-1, \ldots, 2, 1), (k+1), \ldots, (n)\ \rangle$$

Hence its sign depends only on k; it is $+1$ if k is odd and -1 otherwise. ■

Ths, if $\mathcal{M}$ is not empty, then k is well-defined, and $\det(D)$ has a term $a_k x^{n-k}$, where $|a_k|$ is the number of perfect matchings agreeing with E' in exactly $n-k$ edges. This contradicts $\det(D) = \pm x^n$. ■

This now gives us the algorithm for testing if G is in UPM.

The only step we have glossed over here is how to compute, in NC, the determinant of a matrix over integers, or where entries are in $\mathbb{Z} \cup \{x\}$.

One question to ask at this stage is, why doesn't this work for non-bipartite graphs? Is there not an analogue of Lemma 1 using the skew-symmetric Tutte matrix, and setting

Algorithm 3 test-UPM-2

```
Input: Bipartite graph G
Run Construct-UPM(G) to get an edge set E'.
if E' is not a perfect matching then
    reject.
end if
Replace entries in B corresponding to E' with the variable x to get D.
Compute the n + 1 coefficients of the univariate degree n polynomial det(D).
if det(D) ≠ ±x^n then
    reject
else
    accept
end if
```

the E' variables to x and the others to 1? Indeed in [KVV85] this is claimed to work, but the claim was later retracted. In fact there are known cases where the determinant of the Tutte matrix, so initialised, turns out to be $\pm x^{2n}$ even though there is more than one perfect matching; see [HTM06]. What goes wrong in the proof of the claim is that the determinant has squares of perfect matchings as well as cross terms (products of two perfect matchings), and the cross-terms can nullify the perect matchings.

3 NLOG reduces to Bipartite UPM

We now show that UPM, and even Bipartite UPM, is at least as hard as NLOG. A canonical complete problem for NLOG is Directed Acyclic Reachability: Given a directed acyclic graph G and vertices s, t in it, determine whether there is a path from s to t in G. Since NLOG is known to be closed under complementation, another canonical complete problem is Directed Acyclic Unreachability:

$$\text{Directed-Acyclic-UnReachability} = \left\{ \langle G, s, t\rangle\} \mid \begin{array}{l} G \text{ is a directed acyclic graph;} \\ s,\ t \text{ are vertices in } G; \\ \text{there is no path from } s \text{ to } t \text{ in } G. \end{array} \right\}$$

We show that this reduces to bipartite UPM.

Lemma 3 *There is a log-space many-one reduction from Directed Acyclic UnReachability to Bipartite Unique Perfect Matching.*

Proof: Let G, s, t be the given unreachability instance. Without loss of generality, we can assume that there are no edges entering s and no edges leaving t. Construct an undirected graph $H = (V_H, E_H)$ as follows:

$$\begin{aligned} V_H &= \{u_{in} \mid u \in V_G \setminus \{s\}\} \cup \{u_{out} \mid u \in V_G \setminus \{t\}\} \\ E_H &= \{(u_{in}, u_{out}) \mid u \in V_G \setminus \{s, t\}\} \cup \{(u_{out}, v_{in}) \mid (u, v) \in E_G\} \end{aligned}$$

Note that H is a bipartite graph; all the "out" nodes are in one part, and all the "in" nodes in another.

Claim 4 *Paths in G from s to t are in bijection with perfect matchings in H.*

Proof: $\Rightarrow$: Let ρ be a path in G from s to t. Let the vertices on the path be $(s =)u^0, u^1, \ldots, u^l (= t)$ for some l. The corresponding perfect matching in H is

$$M_\rho = \{(u^0_{out}, u^1_{in}), (u^1_{out}, u^2_{in}), \ldots, (u^{l-1}_{out}, u^l_{in})\} \cup \{(v_{in}, v_{out}) \mid v \text{ not on } \rho\}$$

$\Leftarrow$: Let M be a perfect matching in H. Then there is some u^1 such that $(s_{out}, u^1_{in}) \in M$ and so $(s, u^1) \in E_G$. Hence there has to be some u^2 such that $(u^1_{out}, u^2_{in}) \in M$ and so $(u^1, u^2) \in E_G$. Continuing in this way, since the graph is acyclic, we must eventually reach a vertex u^l_{in} such that there is no vertex u^l_{out}. But t is the only such vertex. So we can reconstruct a path ρ_M from s to t.

Now note that the above two constructions put together actually give a bijection. ■

Now we construct graph H' by adding to H the edge (t_{in}, s_{out}). This certainly creates one new perfect matching:

$$M = \{(v_{in}, v_{out}) \mid v \in V_G \setminus \{s, t\}\} \cup \{(s_{out}, t_{in})\}$$

It does not create any other perfect matchings: if any other perfect matching M' uses (s_{out}, t_{in}), then $M \oplus M'$ contains a cycle in H not touching s_{out} or t_{in}, and from this cycle we would be able to extract a directed cycle in G, contradicting its acyclicity.

Thus we conclude

$$\begin{aligned}
(\text{number of perfect matchings in } H') &= 1 + (\text{number of perfect matchings in } H) \\
&= 1 + (\text{number of paths in } G \text{ from } s \text{ to } t) \\
&= 1 \text{ if and only if } G \text{ has no path from } s \text{ to } t
\end{aligned}$$

and hence H' is in UPM if and only if $(G, s, t) \in$ Directed-Acyclic-UnReachability. ■

References

[HTM06] T M Hoang, T Thierauf, and M Mahajan. On the bipartite unique perfect matching problem. In *Proceedings of the 33rd International Colloquium on Automata, Languages and Programming (ICALP)*, pages 453–464. Springer, 2006. Lecture Notes in Computer Science LNCS 4051.

[KVV85] D Kozen, U Vazirani, and V Vazirani. NC algorithms for comparability graphs, interval graphs, and testing for unique perfect matching. In *Proceedings of FST&TCS Conference, LNCS Volume 206*, pages 496–503. Springer-Verlag, 1985.

[RV89] M Rabin and Vijay Vazirani. Maximum matchings in general graphs through randomization. *Journal of Algorithms*, 10(4):557–567, 1989.