CME 305: Discrete Mathematics and Algorithms

Instructor: Professor Aaron Sidford (sidford@stanford.edu)

January 30, 2018

Lecture 7 - Algebraic Methods for Matching[1]

In this lecture we move from unit 1 on combinatorial methods for combinatorial problems to unit 2 on algebraic, spectral, and random walk based methods. The main problem we consider this lecture is that of computing perfect matching in graphs (both bipartite and general).

1 Perfect Matchings

We begin by recalling the definition of matching and perfect matching. Given a graph $G = (V, E)$ we call a subset of the edges $F \subseteq E$ a *matching* if every vertex v in the graph induced by F has degree at most 1, i.e. it has at most one neighbor in the graph induced by F or equivalently is incident to at most one edge in F. Formally we have the following

Definition 1 (Matching)**.** For undirected $G = (V, E)$ a subset of the edges $F \subseteq E$ is a matching if and only if for all $v \in V$ it is the case that $|\{e \in F \,|\, v \in F\}| \leq 1$.

For every edge $\{i, j\} \in M$ for matching M we say that vertex i is *matched* to vertex j. We call a matching *perfect* if every vertex is incident to exactly one edge, i.e. every vertex is matched in the matching.

The main problem we consider in this lecture is how to compute a perfect matching in a graph. We'll focus on obtaining algorithms whose running times depend on $n = |V|$ the number of vertices in the graph, i.e. we will not try to obtain a better running time dependence based on the sparsity, i.e. the number of edges $m = |E|$ in the graph.

1.1 From Perfect Matchings to Maximum Matching

Before we discuss computing perfect matchings, we note that computing a perfect matching in a graph (or certifying that it does not have one) is nearly as hard as computing a maximum matching in a graph, i.e. a matching with the largest cardinality or number of edges in it. Note that the other direction is trivial. A graph has a perfect matching if and only if the maximum matching is perfect and thus computing a maximum matching is clearly harder. The following lemma shows that its asymptotic complexity though (in terms of the number of vertices in the graph) is nearly the same.

Lemma 2. *Given an algorithm which computes perfect matchings in n-node graphs in $O(\mathcal{T}(n))$ time we can produce an algorithm which computes a maximum matching in a graph in $O(T(2n)\log n)$.*[2]

Proof. Let $G = (V, E)$ be a graph with n-nodes for which we wish to compute a perfect matching. Let m_* be the size of the maximum matching in G, i.e. the number of edges in it. Now every edge in a matching matches two unique vertices and therefore the number of unmatched vertices in G is $n - 2m_*$.

Let's consider modifying the graph to match these vertices. Let G_k be the graph we start with G and add vertices $u_1, ..., u_k$ and add an edge $\{v, u_i\}$ for all $v \in G$ and $i \in [k]$ and add an edge $\{u_i, u_j\}$ for all $i \neq j \in [k]$.

[1]These lecture notes are a work in progress and there may be typos, awkward language, omitted proof details, etc. Moreover, content from lectures might be missing. These notes are intended to converge to a polished superset of the material covered in class so if you would like anything clarified, please do not hesitate to post to Piazza and ask.

[2]Here we are assuming that running times, e.g. $\mathcal{T}(n)$ always monotonically increase with n.

Now if $n + k$ is even and $k \geq n - 2m_*$ then G_k has perfect matching. To see this simply take a maximum matching in G, match the $n - 2m_*$ vertices that are not matched to the vertices u_i and for any extra u_i that are un-matched, match them arbitrarily using the $\{u_i, u_j\}$ edges.

On the other hand, if we find a matching of size α in G_k then at most k of these edges are incident to the u_i and therefore this yields a matching of size at least $\alpha - k$ in G. Consequently, since when $n + k$ is even a perfect matching would have $(n+k)/2$ edges we see that a perfect matching in G_k would yield a matching of size at least $\frac{n+k}{2} - k = \frac{n-k}{2}$ which can be at most m_*. Therefore, in order for G_k to have a perfect matching $\frac{n-k}{2} \leq m_*$ and when, this does not hold, i.e. $k < n - 2m_*$, we see that G_k does not have a perfect matching.

Consequently, we can perform binary search on k in $\{0, ..., n\}$ where $n + k$ is even and find the smallest such k for which G_k has a perfect matching. As we have argued, this value of k will be $n - 2m_*$ and the edges in this perfect matching in G will be of size $\frac{n+n-2m_*}{2} - (n - 2m_*) = m_*$, i.e. it will be maximum. Note that the total running time of this procedure is the time to compute $O(\log n)$ perfect matchings on graphs with at most $n + k \leq 2n$ vertices. $\square$

It is not hard to see that a similar reduction can be derived in the case when G is bipartite.

1.2 Cut Characterizations of Matchings

Before we derive our algorithm for computing perfect matchings, here we have a small digression on when exactly a graph has a perfect matching. We will not strictly speaking need this result for our algorithms, but it is an important result in combinatorics and a discussion of perfect matchings would not be complete without it. Moreover, this characterization will highlight again the difference between perfect matchings in bipartite graphs and in general graphs.

In the following lemma we show that we can characterize when a bipartite graph has a perfect matching in terms of how many vertex neighbors a set has. Intuitively it says that a bipartite graph has a perfect matching if when we take any subset of vertices on one side, it is incident to at least as many vertices as the size of that set.

Before we give the lemma a little formalism is required. For graph $G = (V, E)$ and $S \subseteq V$ we let $N_G(S) \stackrel{\text{def}}{=} \{v \in V \mid \exists u \in S, \{u, v\} \in E\}$ denote the neighbors of S in G, i.e. the number of vertices connected to a vertex in S by an edge. We will drop the subscript G when it is clear from context. Furthermore, we say a graph $G = (V, E)$ is bipartite with bipartition $V = L \cup R$ if it is the case that $L \cap R = \emptyset$ and for all $\{i, j\} \in E$ we have $i \in L$ and $j \in R$ or $i \in R$ and $j \in L$.

Theorem 3 (Hall's Marriage Theorem)**.** *If $G = (V, E)$ is bipartite with bipartition $V = L \cup R$ then G has a perfect matching if and only if $|L| = |R|$ and for all $S \subseteq L$ it is the case that $|N(S)| \geq |S|$.*

Proof. Suppose G has a perfect matching. Then clearly $L = R$ since in a perfect matching every vertex on the left is matched with a unique vertex on the right and vice versa. Furthermore, for any set $S \subseteq L$ in a perfect matching each vertex of S is matched with a unique vertex on the right. Consequently, in a perfect matching $|N(S)| = |S|$ and since this graph has a perfect matching plus additional edges $|S| \leq |N(S)|$.

On the other hand suppose that $|L| = |R|$ and $|N(S)| \geq |S|$ for all $S \subseteq L$. Add a vertex s and a vertex t to G, and make every edge $\{u, v\} \in E$ with $u \in L$ and $v \in R$ a directed edge with infinite capacity. Furthermore add an edge (s, v) for all $v \in L$ with capacity 1 and add an edge (v, t) for all $v \in R$ with capacity 1. As we saw last lecture, a maximum s-t flow in this graph consists of length 3 s-t paths that induce matchings and the maximum flow value is $n = |L| = |R|$ if and only if the graph has a perfect matching. Now consider a minimum s-t cut C in this graph we can write $C = \{s\} \cup A \cup B$ for some $A \subseteq L$ and $B \subseteq R$. Now the edges cut by C are the edges from s to $L \setminus A$, the edges from A to $R \setminus B$, and the edges from B to t. However, since this is the minimum cut it should have finite capacity and therefore there are no edges from A to $R \setminus B$ and $u(C) = |L \setminus A| + |B|$. However, since this means that all edges from A go to B we have that $|B| \geq |N(A)| \geq |A|$ and since $|L \setminus A| = n - |A|$ we have $u(C) \geq n$ and therefore the minimum cut value is n, so the maximum flow value is n, and G has a perfect matching. $\square$

Unfortunately the same lemma does not extend directly to characterizing perfect matchings in general graphs. We could try a condition like, for all sets $S \subseteq V$ with $|S| \leq |V|/2$ we have $|N(S)| \geq |S|$ but the graph consisting of 2 independent cycles of length 3 meets this condition, yet does not have a perfect matching. Since if S is one of the cycles then all 3 vertices inside it are neighbors, and if S cuts one of the cycles then at least two vertices are neighbors. This definition could be further restricted by asking for $|S \setminus N(S)| \geq |S|$ but unfortunately, even a perfect matching does not meet this criterion.

Dealing with odd cycles causes much difficulty for computing perfect matchings in general graphs and it can actually be shown that there is no way to model the problem through a small, polynomial-sized linear program under reasonable assumptions. However, there is a characterization of when general graphs have perfect matchings that takes into account odd components more directly. We end this section with its statement, but will not prove it here. Instead we will consider a different set of algorithms (less combinatorial / cut based) to compute them.

Theorem 4 (Tutte's Theorem)**.** *An undirected graph G (which is not necessarily bipartite) has a perfect matching if and only if for all $U \subseteq V$ the graph induced on $V \setminus U$ has at most $|U|$ components with an odd number of vertices.*

2 Perfect Matchings in Bipartite Graphs (Algebraically)

Here we take a different approach to computing perfect matchings. Let's start with the case of a bipartite graph $G = (V, E)$ with bipartition $V = L \cup R$. We assume $|L| = |R| = n$, since otherwise G does not have a perfect matching, and let $L = \{l_1, ..., l_n\}$ and $R = \{r_1, ..., r_n\}$.

Now one nice way to think about perfect matchings in G is in terms of *permutations*, i.e. orderings of $[n]$. If we have a perfect matching $M \subseteq E$ then for all $i \in [n]$ the vertex l_i is matched with a unique r_j. Consequently, a perfect matching is simply a bijective mapping $\pi : [n] \to [n]$ with $\{l_i, r_{\pi(i)}\} \in E$ for all $i \in [n]$. Formally, we let P_n denote the set of permutations which we think of as mapping $\pi \in P_n$ from $[n]$ to $[n]$ with the property that $\pi(i) = \pi(j) \Leftrightarrow i = j$. With this notation we can restate whether or not a graph has a perfect matching as follows

Claim 5. Bipartite $G = (V, E)$ with bipartition $V = L \cup R$ and $L = \{l_1, ..., l_n\}$ and $R = \{r_1, ..., r_n\}$ has a perfect matching if and only if there is $\pi \in P_n$ with $\{l_i, r_{\pi(i)}\} \in E$ for all $i \in [n]$.

Consequently, to check whether or not a graph has a perfect matching, we simply need to optimize over the space of permutations. While there are precisely $n!$ permutations, i.e. this would take exponential time to check compactly, we can try to leverage the fact that there are linear algebraic primitives that can optimize over permutations more efficiently. In particular, we know that the determinant of a matrix is a natural operation for aggregating information about permutations as for any matrix $\mathbf{A} \in \mathbb{F}^{n \times n}$, where here $\mathbb{F}$ is a field (though it suffices to consider $\mathbb{F} = \mathbb{R}$ throughout this note), we have

$$\det(\mathbf{A}) = \sum_{\pi \in P_n} (-1)^{\mathtt{sgn}(\pi)} \mathbf{A}_{1,\pi(1)} \mathbf{A}_{2,\pi(2)}, ..., \mathbf{A}_{n,\pi(n)}$$

where $\mathtt{sgn}(\pi)$ is the signature or parity of π, i.e. the parity of $|x < y \in [n] | \pi(x) > y|$, or the parity of the number of transpositions from the identity permutation to make π.

Thus, the determinant of a matrix is a natural way to aggregate information about permutations on a matrix. This suggests a natural way to use the determinant to check if a graph has a perfect matching. We simply make a matrix $\mathbf{A}(G)$ where we set entry $\mathbf{A}(G)_{ij}$ to be non-zero if and only if $\{l_i, r_j\} \in E$. If we do this, then the only non-zero terms in the sum will be those corresponding to perfect matchings. However, there could be many such summands and therefore to figure out what to set the entries to, we will simply make the non-zero entries of $\mathbf{A}(G)$ variables for now and compute the determinant symbolically. Formally, we introduce variables $x_{ij} \in \mathbb{F}$ for all $i, j \in [n]$ and some field $\mathbb{F}$ and define

$$\mathbf{A}(G)_{ij} = \begin{cases} x_{i,j} & \text{if } (l_i, r_j) \in E \\ 0 & \text{otherwise} \end{cases}$$

with this we have that

$$\begin{aligned} \det(\mathbf{A}(G)) &= \sum_{\pi \in P_n} (-1)^{\mathbf{sgn}(\pi)} \mathbf{A}(G)_{1,\pi(1)} \mathbf{A}(G)_{2,\pi(2)}, ..., \mathbf{A}(G)_{n,\pi(n)} \\ &= \sum_{\pi \in P_n | \pi \text{ corresponds to a perfect matching in } G} (-1)^{\mathbf{sgn}(\pi)} x_{1,\pi(1)} x_{2,\pi(2)} \cdots x_{n,\pi(n)} \end{aligned}$$

Consequently, $\det(\mathbf{A}(G))$ is simply a multivariate polynomial in the x_{ij} with terms corresponding to perfect matchings. To reason about this we introduce a little terminology regarding multivariate functions.

Definition 6 (Multivariate Polynomials)**.** We say that $f \in \mathbb{F}[x_1, ..., x_n]$ is a multivariate polynomial over field $\mathbb{F}$ if for some $\alpha_{i_1,...i_n} \in \mathbb{F}$ and all $x_1, ...x_n \in \mathbb{F}$ we have

$$f(x_1, ..., x_n) = \sum_{i_1, i_2, ..., i_n \in \mathbb{Z}_{\geq 0}} \alpha_{i_1, i_2, ..., i_n} x_1^{i_1} x_2^{i_2} \cdots x_n^{i_n} .$$

We call the maximum value of $i_1 + i_2 + \cdots + i_n$ for which $\alpha_{i_1,i_2,...,i_n} \neq 0$ the *degree of* f denoted $\deg(f)$. We define the degree of the 0 polynomial to be 0 as well. We say two polynomials are equal if all the terms are the same.

With this terminology in place we see that we have established another nice characterization of perfect matchings in bipartite graphs.

Lemma 7. *For bipartite* $G = (V, E)$ *we have* $\det(\mathbf{A}(G)) \in \mathbb{F}[x_{11}, ..., x_{nn}]$ *is a multivariate degree at most* n *polynomial with* $\det(\mathbf{A}(G)) \neq 0$ *if and only if* G *contains a perfect matching.*

Proof. We have already argued that

$$\det(\mathbf{A}(G)) = \sum_{\pi \in P_n | \pi \text{ corresponds to a perfect matching in } G} (-1)^{\mathbf{sgn}(\pi)} x_{1,\pi(1)} x_{2,\pi(2)} \cdots x_{n,\pi(n)} .$$

Consequently it is a degree at most n polynomial by definition (since only n of the x_{ij} appear in a term). Furthermore, we see that for a particular $\pi \in P_n$ the term with variables $x_{1,\pi(1)} x_{2,\pi(2)} \cdots x_{n,\pi(n)}$ only appears for that particular π. Consequently, there is no interaction between these terms and by definition of a polynomial being non-zero it is non-zero if and only if it has a perfect matching. □

With this established, all that remains is to show how to compute $\det(\mathbf{A}(G))$. Unfortunately, even just writing down the polynomial $\det(\mathbf{A}(G))$ can be prohibitively expensive. If G has every edge possible, i.e. it is the bipartite complete graph between two vertex sets of size n, then every permutation corresponds to a perfect matching and $\det(\mathbf{A}(G))$ is a polynomial with $n!$, i.e. an exponential, number of terms. In this case, symbolically as a polynomial, we are not really saying much, we are just enumerating permutations a different way.

So how can we leverage this characterization? One idea is to actually pick an assignment of values to the x_{ij}, i.e. pick some random $r_{ij} \in \mathbb{F}$ for all $i, j \in [n]$. Note that evaluating the multivariate polynomial $\det(\mathbf{A}(G))$ for an assignment of $x_{ij} = r_{ij}$ for some values r_{ij} is the same as computing $\det(\mathbf{A}(G)[\{r_{ij}\}])$ where $\mathbf{A}(G)[\{r_{ij}\}]$ is the matrix where x_{ij} is set to be equal to r_{ij}. Thus we can evaluate this multivariate polynomial in the same time that it takes to compute the determinant of a matrix with explicit values. As we will see next class, this can be done in time polynomial in n.

Now for all assignments $x_{ij} = r_{ij}$ it is the case that if G does not have a perfect matching that $\det(\mathbf{A}(G)[\{r_{ij}\}]) = 0$ as the polynomial itself is 0. However, what is the probability that $\det(\mathbf{A}(G)[\{r_{ij}\}] = 0$ when the polynomial is nonzero, i.e. the graph has a perfect matching. The following lemma, which we will prove next lecture, shows that this probability can be made fairly small.

Lemma 8 (Schwartz Zippel Lemma). *Let $f \in \mathbb{F}[x_1, ..., x_n]$ be a non-zero polynomial of degree d over field $\mathbb{F}$. Let S be a finite subset of $\mathbb{F}$ and suppose $r_1, ..., r_n$ are chosen independently at random from S then*

$$\Pr\left[f(r_1, r_2, ..., r_n) = 0\right] \leq \frac{d}{|S|} \, .$$

Consequently, if we pick $r_{ij} \in \{1, 2, ..., O(n^c)\}$ then with high probability if our bipartite graph has a perfect matching then $\det(\mathbf{A}(G))[\{r_{ij}\}] \neq 0$ whereas if the graph does not have a perfect matching then $\det(\mathbf{A}(G))[\{r_{ij}\}] = 0$. Consequently, by computing determinants of integer valued matrices (which we can do in polynomial time), we can test whether or not a bipartite graph has a perfect matching in polynomial time. In the next lecture we show how to use this to obtain a polynomial time algorithm for computing a bipartite matching in full.

3 Perfect Matchings in General Graphs

In the remainder of this lecture we show that similar techniques can be used to test whether or not a general graph has a perfect matching. For the remainder of this section let $G = (V, E)$ be a general graph and let $V = \{1, ..., n\}$ be its vertices. As before, we will draw a correspondence between permutations $\pi \in P_n$ and perfect matchings in G and then show how to use the determinant of a matrix to check whether or not there is a permutation corresponding to a perfect matching.

Our correspondence is fairly natural. First, for $\pi \in P_n$ with think of $\pi(i)$ as suggesting which vertex vertex v_i should be matched to. For $\pi \in P_n$ let G_π be a directed graph associated with π defined as follows: G has vertices $\{1, ..., n\}$ and an edge $(i, \pi(i))$ for all $i \in [n]$. For this section, I am going to overload notation a little and say $\pi \in E$ or $G_\pi \in G$ if it is the case that $\{v_i, v_{\pi(i)}\} \in E$ for all $i \in [n]$ or equivalently, the undirected graph associated with G_π is a subgraph of G. With this notation we'll say that π corresponds to a perfect matching if $G_\pi \in G$ and G_π has a perfect matching.

So when does G_π have a perfect matching? Note that every vertex in G_π has out-degree one, i.e. one edge leaving it, and in-degree one, i.e. one edge pointing to it. This follows from the fact that π is a permutation and $\pi(i) = j$ for a unique i. Consequently, G_π is simply a collection of disjoint cycles. To see this, note that if we pick a vertex and keep following edges, we will never encounter a vertex twice, except for the vertex we started at (since in-degrees are one). However, clearly a cycle has a perfect matching if and only if its length is even (as edges in a perfect matching would have to alternate). Putting this together we get the following:

Lemma 9. *Graph $G = (V, E)$ with $V = \{1, ..., n\}$ has a perfect matching if and only if some πcorresponds to a perfect matching, which happens if and only if $\pi \in E$ and all cycles in G_π have even length.*

Proof. We have shown the equivalence of π corresponding to a perfect matching and $\pi \in E$ with all cycles in G_π having even length. Furthermore, we have shown that when this holds, G has a perfect matching. Lastly, if G has perfect matching $\{v_1, v_2\}, \{v_3, v_4\}, \ldots$ note that the permutation where for all odd i, $\pi(v_i) = v_{i+1}$ and for all even i, $\pi(v_i) = v_{i-1}$. $\square$

Note also, that given π corresponding to a perfect matching, we can easily find one in nearly linear time. To use this, we simply need to give a matrix where the determinant aggregates over permutations in the graph with even length cycles. We show how to do this in the following lemma.

Lemma 10. *$G = (V, E)$ with vertices $V = \{1, \ldots, n\}$ has a perfect matching if and only if for variables x_{ij} for $ij \in [n]$ the Tutte matrix $\boldsymbol{T}(G)$ defined for all $i, j \in [n]$ by*[3]

$$\boldsymbol{T}(G)_{ij} = \begin{cases} 0 & \text{if } \{i,j\} \notin E \\ x_{ij} & \text{if } i < j \text{ and } \{v_i, v_j\} \in E \\ -x_{ji} & \text{if } i > j \text{ and } \{v_i, v_j\} \in E \end{cases}$$

satisfies $\det(T(G)) \neq 0$.

Proof. As before we have that

$$\begin{aligned} \det(\boldsymbol{T}(G)) &= \sum_{\pi \in P_n} (-1)^{\mathtt{sgn}(\pi)} T(G)_{1,\pi(1),} T(G)_{2,\pi(2)} \cdots T(G)_{n,\pi(n)} \\ &= \sum_{\pi \in P_n, \pi \in E} (-1)^{\mathtt{sgn}(\pi)} (\pm x_{1,\pi(1)})(\pm x_{2,\pi(2)}) \cdots (\pm x_{n,\pi(n)}) \, . \end{aligned}$$

Where the $\pm$ in the above formula depend on whether $i < \pi(i)$ or $i > \pi(i)$. Now suppose there is a $\pi \in P_n$ with $\pi \in E$ that has an odd cycle. Suppose we look at the same permutation with the order of that cycle reversed, denote π'. Reversing the order of a cycle can always be done with an even number of transpositions, i.e. reversing the order of a cycle does not affect its sign, and thus $\mathtt{sgn}(\pi') = \mathtt{sgn}(\pi)$. However, reversing the order of the permutation reverses each of the $\pm$ values in the product and therefore reverses the sign of the term. Consequently, we have argued that for every $\pi \in E$ with an odd cycle there is unique $\pi' \in E$ that gives the same term with the sign reversed. Consequently, all the terms for odd cycles cancel and we have

$$\det(\boldsymbol{T}(G)) = \sum_{\pi \in P_n, \pi \in E, G_\pi \text{has only even cycles}} (-1)^{\mathtt{sgn}(\pi)} (\pm x_{1,\pi(1)})(\pm x_{2,\pi(2)}) \cdots (\pm x_{n,\pi(n)}) \, .$$

Consequently, if $\det(\boldsymbol{T}(G)) \neq 0$ then G has a permutation corresponding to a perfect matching and G has a perfect matching. On the other hand, if G has a perfect matching, denote it $\{v_1, v_2\}, \{v_3, v_4\}, \ldots$ then if we look at the permutation where for all odd i, $\pi(v_i) = v_{i+1}$ and for all even i, $\pi(v_i) = v_{i-1}$ then the variables in the term will be $(-x^2_{v_1 v_2})(-x_{v_3,v_4})^2 \cdots$. As this is the only occurence of this term $\det(\boldsymbol{T}(G)) \neq 0$. $\square$

Consequently, again by Schwartz Zippel, picking a random assignment of the x_{ij}lets us test if G has a perfect matching in the time needed to compute a determinant. In the next lecture we show how to use this to obtain polynomial time algorithms for computing perfect matchings.

[3] Note that the rule used to determine the sign of x_{ij} does not matter so long as it appears once with each sign.