5 Lecture 5, October 10, 2014

5.1 Matchings

A *matching* in a graph $G = (V, E)$ is a set of vertex disjoint edges; the size of the matching is the number of edges.

Let $n = |V|$ and $m = |E|$. A *perfect matching* is one of size $n/2$. A *maximal matching* is one to which no edges can be added. A *maximum matching* is one of largest size.

How hard are the problems of finding such objects?

It is of course easy to find a maximal matching—sequentially. On the other hand, finding one on a parallel computer is a much more interesting problem, which I hope to return to later in the course.

Returning to sequential computation: Finding a maximum matching, or deciding whether a perfect matching exists, are interesting problems. In bipartite graphs, Hall's theorem and the augmenting path method give very nice and accessible polynomial time deterministic algorithms for maximum matching. In general graphs the problem is harder but there are deterministic algorithms running in time $O(\sqrt{n}m)$ [46, 25].

5.2 Bipartite Perfect Matching: Deciding Existence

The first problem we focus on here that of deciding whether a bipartite graph has a perfect matching. As noted there are nice deterministic algorithms for this problem but the randomized one is even simpler. Now we write $G = (V_1, V_2, E)$ with $E \subseteq V_1 \times V_2$. Form the $V_1 \times V_2$ "variable" matrix A which has $A_{ij} = x_{ij}$ if $\{i, j\} \in E$, and otherwise $A_{ij} = 0$.

Let q be some prime power and consider the x_{ij} as variables in $GF(q)$. The determinant of A, then, is a polynomial in the variables x_{ij}.

Lemma 21 $\text{Det}(A) \neq 0$ *iff G has a perfect matching.*

(This is a special case of a result of Tutte 25 that we will see later.) **Proof:** Every monomial in the expansion of the determinant corresponds to a permutation, which in turn corresponds to a perfect matching in the complete bipartite graph. That monomial is nonzero only if all edges are present in G, so a graph without perfect matchings has Det $= 0$; conversely, if some perfect matching is present, it puts a monomial in the determinant which identifies the perfect matching, and is not canceled by any other monomial. (Easily verified by just taking the assignment to the polynomial in which these variables are set to 1 and the others to 0.) $\square$

This suggests the following exceptionally simple algorithm: compute the polynomial and see if it is nonzero.

There's a problem with this idea! The determinant has exponentially many monomials. This is not a problem for computing determinants over a ring such as the integers, because even the sum of exponentially many integers only has polynomially more bits than the largest of those integers has. However, in this ring of multivariate polynomials, there are exponentially many *distinct* terms to keep track of if you want to write the polynomial out as a sum of monomials. Of course the determinant has a more concise representation (namely, as "Det(A)"), but we do not know how to efficiently convert that to any representation that displays transparently whether the polynomial is the 0 polynomial.

So we modify the original suggestion. Since we do know how to efficiently compute determinants of scalar matrices, let's substitute scalar values for the x_{ij}s. What values should we use? Random ones.

Revised Algorithm: Sample the x_{ij}'s u.a.r. in $GF(q)$; call the sampled matrix A_R. Compute Det(A_R); report "G has/hasn't a perfect matching" according to whether Det$(A_R) \neq 0$ / $= 0$.

Clearly the algorithm answers correctly if there is no perfect matching. What needs to be shown is that the probability of error is small if there is a perfect matching. (So this is an RP-type algorithm for "G has a perfect matching".)

Theorem 22 *The runtime of the algorithm is $\tilde{O}(n^\omega)$ where ω is the runtime exponent of matrix multiplication. The algorithm is error-free on graphs lacking a perfect matching, and the probability of error of the algorithm on graphs which have a perfect matching, is at most n/q.*

(The $\tilde{O}$ notation means "up to logarithmic factors", which here account for computing with poly-sized numbers.)

All we have to do, then, is use a prime power $q \geq 2n$ in order to have error probability $\leq 1/2$.

Incidentally, there is always a prime $2n \leq q < 4n$; this is called "Bertram's postulate". This fact isn't quite strong enough for us since we would need to be able to find a prime in the right size range efficiently, but that too can be done. We don't have to work this hard though, since we're satisfied with prime powers rather than primes. We can simply use the first power of 2 after $2n$.

We will prove Theorem 22 after introducing a general useful tool.

5.3 Polynomial Identity Testing

In the previous section we saw that testing for existence of a perfect matching in a bipartite graph can be cast as a special case of the following problem. We are given a polynomial $p(x)$, of total degree n, in variables $x = (x_1, \ldots, x_m)$, $m \geq 1$. (The total degree of a monomial is the sum of the degrees of the variables in it; the total degree of a polynomial is the greatest total degree of its monomials.) We are agnostic as to how we are "given" the polynomial, and demand only that we be able to quickly evaluate it at any scalar assignment to the variables. We wish to test whether the polynomial is identically 0, and our procedure for doing so is to evaluate it at a random point and report "yes" if the value there is 0. We rely on the following lemma.

Lemma 23 *Let $z(p) \subseteq q^m$ be the set of zeros of p. If $p \neq 0$ then $|z(p)| \leq nq^{m-1}$.*

This is a special case of the following more general statement which holds for any, even infinite, field κ.

Lemma 24 *Let p be a nonzero polynomial of total degree n in variables $x_1, \ldots, x_m$ over a field κ. Let $S_1, \ldots, S_m$ be subsets of κ with $|S_i| \leq s$ for all i. Then $|(S_1 \times \ldots \times S_m) \cap z(p)| \leq s^{m-1}n$.*

This is usually known as the Schwartz-Zippel lemma [57, 63], although the results in these two publications were not precisely equivalent, and moreover, there were at at least two other discoveries of some version of the result, by Ore [44] and by DeMillo and Lipton [16]. A generalization beyond polynomials is due to Gonnet [27].

Before giving the proof, a word on a subtle point: what does it mean for a polynomial p to be "nonzero"? There are two reasonable notions. We are considering a polynomial over a field κ, which is to say, the coefficients of all the monomials in the polynomial lie in κ.

- **Strong:** $p(x) \neq 0$ for some $x \in \kappa$.
- **Weak:** Some monomial coefficient is nonzero.

Clearly the strong condition implies the weak one, but the converse is not true—consider, for example, the polynomial $x^2 + x$ over the field $\mathbb{Z}/2$.

(However, the conditions are equivalent over infinite fields such as $\mathbb{R}$ or $\mathbb{C}$. In fact, you will be able to conclude that from Lemma 24.)

When we assume that a polynomial is nonzero, we mean it *in the weak sense*. Since the nonzero-ness assumption appears in the hypothesis of Lemma 24, this means the lemma is stronger; and in applications we usually rely on this.

Proof: of Lemma 24: First consider the univariate case, $m = 1$. This follows by induction on n because if $n \geq 1$ and $p(a) = 0$ then p can be factored as $p(x) = (x - a) \cdot g(x)$ for some g of degree $n - 1$.

Next we handle $m > 1$ by induction. Write p in the form $p(x) = \sum_0^n x_1^i p_i(x_2, \ldots, x_m)$, and let i be largest such that $p_i \neq 0$. The degree of p_i is at most $n - i$, so by induction,

$$s^{m-2} \cdot (n - i) \geq |(S_2 \times \ldots \times S_m) \cap z(p_i)| =: r$$

For $(x_2, \ldots, x_m) \in z(p_i)$ we allow as a worst case that all choices of x_1 yield a zero of p. For $(x_2, \ldots, x_m) \notin z(p_i)$, p restricts to a nonzero polynomial of degree i in the variable x_1, so by the case $m = 1$ there are at most i choices of x_1 yielding a zero of p.

Combining these we have

$$\begin{aligned} |z(p)| &\leq s \cdot r + i \cdot (s^{m-1} - r) \\ &\leq s \cdot s^{m-2} \cdot (n - i) + i \cdot (s^{m-1} - s^{m-2} \cdot (n - i)) \\ &= s^{m-1} \cdot n - s^{m-2} \cdot i \cdot (n - i) \\ &\leq s^{m-1} \cdot n. \end{aligned}$$

□

Comment: This lemma gives us an efficient randomized way of testing whether a polynomial is identically zero, and naturally, people have wondered whether there might be an efficient deterministic algorithm for the same task. So far, no such algorithm has been found, and it is now known that any such algorithm would have very strong implications in complexity theory [36].

5.4 Back to deciding the existence of a perfect matching in a bipartite graph

Proof: of Theorem 22: This is a corollary of the above theorem, noting that the determinant of A is a multivariate polynomial of degree n. □

5.5 Perfect Matchings in General Graphs: Deciding Existence

Deterministically, deciding the existence of a perfect matching in a general graph is notably harder than the same problem in a bipartite graph. (As noted, we have poly-time algorithms, but not nearly so simple ones.) With randomization, however, we can adapt the same approach to work with almost equal efficiency.

We must define the Tutte matrix of a graph $G = (V, E)$. Order the vertices arbitrarily from $1, \ldots, n$ and set

$$T_{ij} = \begin{cases} 0 & \text{if } \{i, j\} \notin E \\ x_{ij} & \text{if } \{i, j\} \in E \text{ and } i < j \\ -x_{ji} & \text{if } \{i, j\} \in E \text{ and } i > j \end{cases}$$

Theorem 25 (Tutte [61]) $\mathrm{Det}(T) \neq 0$ *iff T has a perfect matching.*

Proof: If T has a perfect matching, assign $x_{ij} = 1$ for edges in the matching, and 0 otherwise. Each matching edge $\{i, j\}$ describes a transposition of the vertices i, j. Since every row and column has only the single nonzero entry corresponding to the matching edge it is part of, the matrix is the permutation matrix (with some signs) of the involution that transposes the vertices on each edge. Since a transposition has sign -1 and there is a single -1 in each pair of nonzero entries, we have $\mathrm{Det}(T) = 1$.

Conversely suppose $\mathrm{Det}(T) \neq 0$ as a polynomial. Consider the determinant as a signed sum over permutations. The net contribution to the determinant from all permutations having an odd cycle is 0, for the following reason. In each such cycle identify the "least" odd cycle by some criterion, e.g., ordering the cycles by their least-indexed vertex. Then flip the direction of the least odd cycle. This map is an involution on the set of permutations. It carries the permutation to another, which contributes the opposite sign to the determinant, since the sign of all edges in the cycle flipped.

Therefore there are permutations of the vertices supported by T (i.e., each vertex is mapped to a destination along one of the edges incident to it, that is, $\pi(i) = j \Rightarrow T_{ij} \neq 0$) having only even cycles. The even cycles of length 2 are matching edges, and in any even cycle of length greater than 2, we can use every alternate edge; altogether we obtain a perfect matching. □

In exactly the same way as for the bipartite case, this yields:

Theorem 26 *The runtime of the algorithm to determine existence of a perfect matching in a graph on n vertices, is $\tilde{O}(n^\omega)$ where ω is the runtime exponent of matrix multiplication. The algorithm is error-free on graphs lacking a perfect matching, and the probability of error of the algorithm on graphs which have a perfect matching, is at most n/q.*

By self-reducibility this immediately yields an $\tilde{O}(n^{\omega+2})$-time algorithm for *finding* a perfect matching.

We'll see a beautiful method in the next lecture that has the following advantage: it works in RNC, that is to say, the algorithm is randomized, uses polynomially many processors in parallel, and terminates in polylogarithmic time.