The permanent and the determinant

Uriel Feige
Department of Computer Science and Applied Mathematics
The Weizman Institute
Rehovot 76100, Israel
uriel.feige@weizmann.ac.il

May 18, 2022

1 Introduction

Given an order n matrix A with entries a_{ij}, its *permanent* is

$$per(A) = \sum_{\sigma} \prod_{i=1}^{n} a_{i\sigma(i)}$$

where σ ranges over all permutations on n elements. Its determinant is

$$det(A) = \sum_{\sigma} (-1)^{\sigma} \prod_{i=1}^{n} a_{i\sigma(i)}$$

where $(-1)^{\sigma}$ is $+1$ for even permutations and -1 for odd permutations. A permutation is even if it can be obtained from the identity permutation using an even number of transpositions (where a transposition is a swap of two elements), and odd otherwise.

Letting A_{-ij} denote the matrix obtained by removing the ith row of A and the jth column of A, we have the Laplace expansion (along row i) of the permanent/determinant of a matrix A of order n:

$$per(A) = \sum_{j=1}^{n} a_{ij} \cdot per(A_{-ij}) \text{ and } det(A) = \sum_{j=1}^{n} (-1)^{i+j} a_{ij} \cdot det(A_{-ij})$$

The determinant can be computed in polynomial time by Gaussian elimination, and in time n^{ω} by fast matrix multiplication. On the other hand, there is no polynomial time algorithm known for computing the permanent. In fact, Valiant showed that the permanent is complete for the complexity class $\#P$, which makes computing it as difficult as computing the number of solutions of NP-complete problems (such as SAT, Valiant's reduction was from Hamiltonicity).

For 0/1 matrices, the matrix A can be thought of as the adjacency matrix of a bipartite graph (we refer to it as a *bipartite adjacency matrix* – technically, A is an off-diagonal block of the usual adjacency matrix), and then the permanent counts the number of perfect matchings. Computing the permanent of integer matrices can be reduced in polynomial time to that of computing the permanent of 0/1 matrices.

The permanent and the determinant can both be viewed as multilinear polynomials, where the entries a_{ij} of the respective matrices represent variables. Hence computing the permanent or determinant can be viewed as evaluating a multinomial at a point given by the respective matrix (the value of every variable a_{ij} of the multinomial is equal to the entry of A in location ij). Valiant showed how the evaluation of any explicit integer multinomial (one that may have exponentially many monomials, but there is a polynomial time algorithm that given a monomial computes its integer coefficient) can be reduced in polynomial time to computing the permanent of some polynomial size matrix.

2 Computing the permanent

The natural algorithm takes time roughly $n!$. Ryser showed a quicker way to compute the permanent, using the exclusion-inclusion formula.

For a nonempty set S of columns, let $R_i(S)$ denote the sum of items in columns S of row i. Then:

$$per(A) = \sum_S (-1)^{n-|S|} \prod_{i=1}^{n} R_i(S)$$

Computing the permanent using the above formula takes time $2^n \cdot n^{O(1)}$.

To see that Ryser's formula is correct, consider the representation of the permanent as a multinomial, and observe that Ryser's formula gives the correct coefficient for every monomial. For monomials that correspond to permutations, this coefficient is 1 (they contribute to Ryser's formula only when S is the set of all columns). For monomials that involve variables from $c < n$ columns, the number of times that they appear in Ryser's formula is

$$\sum_{j=0}^{n-c} (-1)^{n-c+j} \binom{n-c}{j} = 0$$

(Equality with 0 can be seen by the equivalence with fact that with $n - c$ coin tosses, the probability that an even number of them come up heads is equal to the probability that an odd number of them come up heads. It all depends on the last coin toss.)

More details can be found in [2] (Chapter 11).

3 Relations with matchings

To see whether a bipartite graph has an even or odd number of perfect matching, compute the permanent modulo 2 of its bipartite adjacency matrix. This is equivalent to computing the determinant modulo 2, and hence can be done in time $O(n^\omega)$.

To see whether the number of perfect matchings is divisible by 3, compute the permanent modulo 3. However, this problem is difficult (NP-hard under randomized reductions).

To see whether a bipartite graph has a perfect matching, Lovasz suggests the following randomized algorithm that works in time roughly n^ω. Observe that there is a perfect matching iff the determinant, with formal variables replacing the 1 entries in the matrix, is not the 0-multinomial. Let $p > n^2$ be prime, replace the 1 entries by random entries in

$\{0, \ldots p-1\}$ and compute the determinant modulo p. If there is no perfect matching, the answer is 0. If there is a perfect matching, the answer is nonzero with probability at least $(p-n^2)/p$. This last fact follows from the Schwartz-Zippel lemma.

Lemma 1 *Let p be prime, and assume that all computations are performed in Z_p (namely, all coefficients are in $\{0, \ldots p-1\}$ and all computations are done modulo p). Let $P(x_1, \ldots, x_k)$ be a multilinear polynomial over k variables that is not identically 0. Let x be a random assignment to all variables, where the the value of each variable is chosen independently at random from $\{0, \ldots p-1\}$. Then the probability that $P(x) \neq 0$ is at least $(1-1/p)^k \geq (p-k)/p$.*

Proof: The proof is by induction on k. For the base case $k = 1$, $P(x)$ is a linear polynomial $ax + b$. If $a = 0$ then $b \neq 0$, and the polynomial is never 0. If $a \neq 0$, then $P(x) = 0$ only for $x = -b/a$ modulo p, and there is a unique such x in $\{0, \ldots p-1\}$.

For the inductive step, assume the lemma for k and prove for $k+1$. Write $A(x_1, \ldots x_{k+1}) = x_{k+1}A_1(x_1, \ldots x_k) + A_2(x_1, \ldots x_k)$. Pick random values for $x_1, \ldots, x_k$ first. By induction, with probability at least $(1-p)^k$, $A_1(x_1, \ldots x_k) \neq 0$. If so, then when picking x_{k+1} at random, with probability $(p-1)/p$, $x_{k+1} \neq A_2(x_1, \ldots x_k)/A_1(x_1, \ldots x_k)$. Hence $P(x_1, \ldots, x_{k+1}) \neq 0$ with probability at least $(1-1/p)^{k+1}$. $\square$

The approach described above can be extended to testing whether a non-bipartite graph has a perfect matching, and also to actually find a perfect matching, essentially in time n^ω. See [1] for more information.

4 Counting spanning trees

Recall that counting matchings is NP-hard. Here we sketch the famous matrix-tree theorem of Kirchoff, that shows how to efficiently count spanning trees.

Recall that the Laplacian L_G of a graph G is the matrix $D_G - A_G$, where D_G is a diagonal matrix with the degrees of the vertices along its diagonal, and A_G is the adjacency matrix of G.

Theorem 2 *For an n vertex graph G, let L_G denote its Laplacian matrix, and let $\hat{L}$ denote an order $n-1$ matrix derived from L_G by choosing an arbitrary index $1 \leq i \leq n$, and removing row i and column i from L_G. Then the number of spanning trees in G is exactly $det(\hat{L})$.*

There are several proofs for Theorem 2. The one we provide is similar to the one in [2] (Chapter 34).

We shall use the Binet-Cauchy expansion of the determinant.

Lemma 3 *For r by c matrices A and B with $r < c$ the following holds:*

$$det(AB^T) = \sum_S det(A_S)det(B_S)$$

where A_S is the block S of columns from A, with $|S| = r$.

We defer the proof of Lemma 3 to later and proceed to describe the algorithm for counting spanning trees.

Given a graph G with n vertices and m edges, direct its edges arbitrarily. Consider the n by m incidence matrix M of the directed graph with vertices as rows, edges as columns, and $+1$ and -1 entries for incoming and outgoing edges.

Restrict attention to a submatrix of M induced by a set S of $n-1$ columns (edges), and denote this submatrix by M_S. As there are n vertices, either the set S is a spanning tree of G, or the subgraph of G whose edges are S has at least two connected components. In the latter case, the rank of M_S is at most $n-2$, because for each connected component, summing up its rows gives the 0 vector (as each edge contributes both $+1$ and -1), and hence these rows induce a linear dependency. If S forms a spanning tree, then the rank of M_S is $n-1$, because in this case there is no subset of $k < n$ rows that forms a linear dependency (in every such subset there is an edge for which only one of the endpoints remains, and the corresponding entry cannot be cancelled out by other rows in the subset). Remove an arbitrary row from M_S to get a matrix B. If S is a spanning tree the rank of B is $n-1$, and otherwise the rank is smaller than $n-1$. Hence B has full rank and nonzero determinant iff the edges of S form a spanning tree. Being a matrix with at most one $+1$ and at most one -1 entry in each column, the matrix is totally unimodular, and hence its determinant is either 0 or ± 1. It follows that $det(BB^T) = det(B)det(B^T) = 1$ if the columns form a spanning tree (because then B has full rank and its determinant cannot be 0), and 0 otherwise.

Hence removing one row from from M (we denote the resulting matrix by $\hat{M}$) and having S range over all blocks of $n-1$ columns, we have proved that the number of spanning trees is exactly $\sum_B det(BB^T)$. But by Lemma 3 (with $r = n-1$ and $c = m$) this last expression is exactly equal to $\det(\hat{M}\hat{M}^T)$, which can be computed in polynomial time.

Observe that MM^T is exactly the Laplacian $L(G)$ of the graph (degrees along the diagonal, -1 in entry $L_{i,j}$ if there is an edge (i,j)). Observe further than $\hat{M}\hat{M}^T$ can be obtained from the Laplacian by removing one row and one column (both with the same index). This proves Theorem 2.

We now prove the Binet-Cauchy formula (Lemma 3).

Proof: Recall that we assumed that $r < c$. It is instructive to note that for $c = r$ the Binet-Cauchy formula is the known equality $det(AB^T) = det(A)det(B)$, and for $r > c$ we have that the rank of AB^T is at most $c < r$, and hence $det(AB^T) = 0$. (The right hand side of the Binet-Cauchy formula is undefined in this case.)

Returning to $r < c$, let Δ be an order c diagonal matrix with formal variables x_i along the diagonal. We show that

$$det(A\Delta B^T) = \sum_S det(A_S)det(B_S) \prod_{i \in S} x_i$$

are identical as formal polynomials, and the Binet-Cauchy formula follows by setting $x_i = 1$ for all i.

Observe that $A\Delta B^T$ is an order r matrix with entries that are linear forms in the x_i variables. Hence $det(A\Delta B^T)$ is a homogeneous polynomial of degree r. For monomials with fewer than r distinct variables, their coefficient must be 0. This can be seen by substituting arbitrary values in these variables and 0 in the rest. The rank of $A\Delta B^T$ in this case

becomes smaller than r, and hence the polynomial restricted only to these variables is the 0 polynomial. The coefficients of other monomials (say, defined over a set S of r variables) can be determined by substituting 1 for the S variables and 0 for other variables. Then $A\Delta B^T = A_S(B_S)^T$ and the coefficient of $\prod_{i \in S} x_i$ is indeed $det(A_S)det(B_S)$. $\square$

5 Homework

1. How does the statement (and proof) of Lemma 1 change if the multinomial is not multilinear, but instead, has total degree at most d? (That is, if in every monomial, the sum of degrees of the variables is at most d.)

2. Cayley's formula says that the number of spanning trees of the complete graph on n vertices is n^{n-2}. Cayley's formula has many proofs. Use the matrix-tree theorem to prove Cayley's formula. (For the proof, it may be useful to recall that the determinant of a matrix equals the product of its eigenvalues, and that adding the identity matrix to a matrix increases every eigenvalue by 1.)

References

[1] Nicholas J. A. Harvey. Algebraic Algorithms for Matching and Matroid Problems. SIAM Journal on Computing, 39(2):679-702, 2009.

[2] J.H. van Lint and R.M. Wilson. A Course in Combinatorics. Cambridge University Press, 1992.