Lecture Notes November 18, 2026

Refutation of the Exponential Time Hypothesis

Lecturer: Hamed Hatami

Lecture overview

In this lecture, we present a deterministic algorithm that computes the permanent of an $n \times n$ matrix with $O(1)$ nonzero entries in each row and column using $2^{o(n)}$ arithmetic operations. Combined with sparsification and a standard reduction, this provides a $2^{o(n)}$ algorithm for 3SAT on n variables and therefore refutes the Exponential Time Hypothesis. We follow Ryan O'Donnell's reworking of the proof announced by OpenAI.

1 Introduction

Every problem in NP can be solved by exhaustive search over all candidate solutions. For 3SAT on n variables, this means testing all 2^n truth assignments. Since 3SAT is NP-complete, P $\neq$ NP is equivalent to the statement that 3SAT cannot be solved in polynomial time. Impagliazzo and Paturi [IP01] proposed a far stronger statement:

> **Exponential Time Hypothesis (ETH).** *There is a constant $\varepsilon > 0$ such that no deterministic algorithm solves 3SAT on n variables in time $O(2^{\varepsilon n})$.*

Since ETH is ultimately about exhaustive search, it is more natural to measure the complexity in the length of a solution rather than the input length; for 3SAT, this is the number of variables.

Although 3SAT is NP-complete, ETH does not automatically extend to every NP-complete problem. Indeed, lower bounds of the form $2^{\Omega(n)}$ are not preserved by arbitrary polynomial-time reductions. Nevertheless, for many important NP-complete problems, there are reductions to and from 3SAT that increase the solution length by at most a constant factor. Consequently, ETH could be equivalently formulated as the nonexistence of subexponential-time algorithms for many other problems, e.g. kSAT and graph k-colourability ($k \geq 3$), Independent Set, Clique, and Vertex Cover [IPZ01].

In October 2026, OpenAI announced a proof that ETH is false [Ope26], which implies that all of the problems listed above have subexponential-time algorithms. In fact, the proof gives more than a decision algorithm: it counts the satisfying assignments.

> *There is a deterministic $2^{o(n)}$-time algorithm that **counts** the satisfying assignments of a given n-variable 3SAT formula.*

Shortly after, O'Donnell [O'D26] recast the argument in a more natural form, as an algorithm for the permanent of sparse matrices. This suffices to refute ETH: by sparsification [IPZ01], one may assume the 3SAT formula has $O(n)$ clauses, and a standard reduction [DHM+14] then encodes the number of its satisfying assignments as the permanent of an $O(n) \times O(n)$ integer matrix with at most *three* nonzero entries in each row and column.

In these notes, we present a slightly modified version of the algorithm in [O'D26], in which the sparse matrix is decomposed using the group structure of $\mathbb{F}_2^m$.

The permanent. The permanent of an $n \times n$ matrix A with entries $a_{u,v}$ is defined as

$$\operatorname{per}(A) \coloneqq \sum_{\pi \in S_n} \prod_{u=1}^{n} a_{u,\pi(u)}. \tag{1}$$

We set the permanent of a nonsquare matrix to 0. When A is the bi-adjacency matrix of a bipartite graph, permanent counts the number of perfect matchings.

We call a matrix Δ-*sparse*, if there are at most Δ nonzero entries in each row and column.

Theorem 1.1 (Refutation of ETH via sparse permanents)**.** *For every integer $\Delta \geq 1$, the permanent of any Δ-sparse $n \times n$ integer matrix can be computed deterministically with*

$$2^{O_\Delta\left(n(\log\log n)^2/\sqrt{\log n}\right)}$$

arithmetic operations.

Proof overview. Let

$$d = \Theta\big(\sqrt{\log n}\big), \qquad \ell = \Theta\Big(\frac{\log n}{\log\log n}\Big).$$

We will construct a matrix whose entries are univariate polynomials of degree at most d,

$$A(t) = \sum_{i=0}^{d} A_i\, t^i,$$

such that $A(0) = A$, while for $k = 1, \dots, \ell$ the matrices $A(k)$ are well structured, so that their permanents, and the permanents of their square submatrices, are easy to compute.

The permanent $P(t) \coloneqq \operatorname{per}(A(t))$ is a polynomial of degree at most nd. Such a polynomial is determined by its value and its first $h-1$ derivatives at each of the points $1, \dots, \ell$, provided that $\ell h > nd$. Thus, it suffices to compute

$$P^{(j)}(k) \qquad (k \in [\ell],\ 0 \leq j < h), \qquad h = \frac{nd}{\ell} + 1 = \Theta\Big(\frac{n \log\log n}{\sqrt{\log n}}\Big).$$

These values will be easy to compute because of the structure of $A(k)$.

Why the degree must be small. Computing $P^{(j)}(k)$ involves permanents of submatrices of $A(k)$ obtained by deleting at most j rows and the same number of columns. For $j < h$ there are at most $\binom{n}{<h}^2$ such submatrices, and this number is subexponential in n only when $h = o(n)$. Since $h \approx nd/\ell$, it is therefore essential that the degree satisfies $d = o(\ell)$.

Under the rug. We measure running time by the number of arithmetic operations over $\mathbb{Q}$, and we do not track the sizes of the numbers involved. It is routine to check that every number appearing in the algorithm has bit length polynomial in n and in the bit length of the input.

Notation. For $x \in \{0,1\}^n$, we write $|x| = x_1 + \cdots + x_n$. For $S, T \subseteq [n]$, we write $A_{S\times T}$ for the submatrix of A on rows S and columns T.

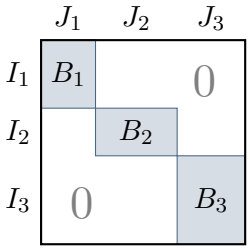
2 Easy permanents: small blocks plus low rank

The naive way of computing the permanent using (1) requires adding $n!$ products. However, Ryser's formula allows us to compute the permanent with only $2^{O(n)}$ arithmetic operations:

$$\operatorname{per}(A) = \sum_{x\in\{0,1\}^n} (-1)^{n-|x|} \prod_{u=1}^{n} \Big(\sum_{v=1}^{n} a_{u,v} x_v \Big). \tag{2}$$

Block-diagonal matrices. Suppose the rows are partitioned into $I_1, \ldots, I_p$ and the columns into $J_1, \ldots, J_p$. We say that B is *s-block-diagonal* with respect to these partitions if $|I_a|, |J_a| \le s$ for all a and

$$B_{uv} = 0 \quad \text{unless } (u,v) \in I_a \times J_a \text{ for some } a.$$



The permanent of such a matrix is the product of the permanents of its blocks,

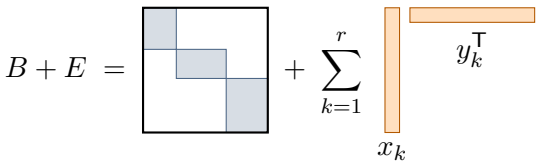
$$\operatorname{per}(B) = \prod_{a=1}^{p} \operatorname{per}\big(B_{I_a \times J_a}\big),$$

and each factor can be computed with $\operatorname{poly}(n)\, 2^s$ arithmetic operations by Ryser's formula.

Adding a low-rank matrix. Now let E be a matrix of rank r, written as a sum of rank-one matrices,

$$E = \sum_{k=1}^{r} x_k y_k^{\mathsf{T}}, \qquad E_{uv} = \sum_{k=1}^{r} x_k(u)\, y_k(v). \tag{3}$$

The matrix $B + E$ has the following shape:



We show that the permanent of $B + E$ is easy to compute when s and r are both small. It is a good exercise to see why this is so when $B = 0$, or when B consists of a single block. For more blocks, the algorithm processes them one at a time, by dynamic programming.

Proposition 2.1. *Let B be s-block-diagonal, and let E have rank at most r. Then $\operatorname{per}(B+E)$, or the permanent of any specified square submatrix of $B+E$, can be computed with*

$$(r+2)^{O(s)}\, n^{O(r+1)}$$

arithmetic operations

Proof. Write $E = \sum_{k=1}^{r} x_k y_k^{\mathsf{T}}$. Then

$$\operatorname{per}(B+E) = \sum_{\pi \in S_n} \prod_{u=1}^{n} \Big(B_{u\pi(u)} + \sum_{k=1}^{r} x_k(u)\, y_k\big(\pi(u)\big) \Big).$$

Expanding the product, each row u is matched either through B or through some $x_k y_k^{\mathsf{T}}$. Let S_0, T_0 be the rows and columns matched through B, and S_k, T_k those matched through $x_k y_k^{\mathsf{T}}$. Once the sets are fixed sets, summing over π gives $\operatorname{per}(B_{S_0 \times T_0})$ for the B-part, and, since there are $|S_k|!$ ways of pairing S_k with T_k, we obtain

$$\operatorname{per}(B+E) = \sum_{\substack{[n]=S_0 \sqcup \cdots \sqcup S_r \\ [n]=T_0 \sqcup \cdots \sqcup T_r \\ |S_k|=|T_k|\ \forall k}} \operatorname{per}\big(B_{S_0 \times T_0}\big) \prod_{k=1}^{r} \Big(|S_k|! \prod_{u \in S_k} x_k(u) \prod_{v \in T_k} y_k(v) \Big). \tag{4}$$

We break this sum further according to how the sets S_k and T_k interact with the blocks of B. Write

$$S_{a,k} = I_a \cap S_k, \qquad T_{a,k} = J_a \cap T_k \qquad (1 \le a \le p,\ 0 \le k \le r).$$

Since B is block diagonal, $\operatorname{per}(B_{S_0 \times T_0})$ is the product of the permanents $\operatorname{per}(B_{S_{a,0} \times T_{a,0}})$, which vanish unless $|S_{a,0}| = |T_{a,0}|$ for all a. The only parts of (4) that involve several blocks at once are the factorials $|S_k|!$ and the constraint $|S_k| = |T_k|$, both depending only on the sizes

$$|S_k| = \sum_{a=1}^{p} |S_{a,k}|, \qquad |T_k| = \sum_{a=1}^{p} |T_{a,k}|.$$

For each block a, we record these sizes in the *count vectors*

$$\alpha^a = \big(|S_{a,0}|, |S_{a,1}|, \ldots, |S_{a,r}|\big), \qquad \beta^a = \big(|T_{a,0}|, |T_{a,1}|, \ldots, |T_{a,r}|\big).$$

We therefore define the contribution of block a with counts $(\alpha', \beta') \in \mathbb{Z}_{\ge 0}^{r+1} \times \mathbb{Z}_{\ge 0}^{r+1}$ as

$$F_a(\alpha', \beta') = \sum_{\substack{I_a = S_{a,0} \sqcup \cdots \sqcup S_{a,r} \\ J_a = T_{a,0} \sqcup \cdots \sqcup T_{a,r} \\ |S_{a,k}|=\alpha'_k,\ |T_{a,k}|=\beta'_k\ (0 \le k \le r)}} \operatorname{per}\big(B_{S_{a,0} \times T_{a,0}}\big) \prod_{k=1}^{r} \Big(\prod_{u \in S_{a,k}} x_k(u) \prod_{v \in T_{a,k}} y_k(v) \Big).$$

Grouping the terms of (4) by the total counts $\alpha = (|S_0|, |S_1|, \ldots, |S_r|)$, and using $|S_k| = |T_k|$, gives

$$\operatorname{per}(B+E) = \sum_{\substack{\alpha \in \mathbb{Z}_{\ge 0}^{r+1} \\ \alpha_0 + \cdots + \alpha_r = n}} \Big(\prod_{k=1}^{r} \alpha_k! \Big) \sum_{\substack{\alpha^1 + \cdots + \alpha^p = \alpha \\ \beta^1 + \cdots + \beta^p = \alpha}} \prod_{a=1}^{p} F_a(\alpha^a, \beta^a).$$

Computing the sum. There are too many choices of $(\alpha^1, \dots, \alpha^p)$ to enumerate them efficiently. Instead, we compute the inner sum by dynamic programming over the blocks. Let $D_0(\alpha, \beta) = \mathbf{1}_{\{\alpha=\beta=0\}}$ and

$$D_a(\alpha, \beta) = \sum_{\alpha' \le \alpha,\ \beta' \le \beta} D_{a-1}(\alpha - \alpha', \beta - \beta')\, F_a(\alpha', \beta') \qquad (1 \le a \le p),$$

where $\le$ is coordinatewise. By induction, $D_a(\alpha, \beta)$ is the inner sum restricted to the first a blocks, so

$$\operatorname{per}(B + E) = \sum_{\alpha} \Big(\prod_{k=1}^{r} \alpha_k! \Big) D_p(\alpha, \alpha).$$

Cost. In the dynamic programming, the tables $D_a(\alpha, \beta)$ and $F_a(\alpha, \beta)$ each have at most $(n+1)^{2(r+1)}$ entries, and each step of the recursion for D_a costs at most $(n+1)^{4r+4}$ operations.

The formula for F_a has at most $(r+1)^{|I_a|+|J_a|} \le (r+1)^{2s}$ summands in total, and each requires a permanent $\operatorname{per}(B_{S_{a,0} \times T_{a,0}})$, which takes $2^{O(s)}$ operations by (2). The total number of operations is

$$\operatorname{poly}(n)\, 2^s (r+1)^{2s} (n+1)^{4r}.$$

For a square submatrix, run the same algorithm on the surviving rows and columns. □

3 Deforming a sparse matrix

We have seen that permanents of matrices that are s-block-diagonal plus low rank are easy to compute. We now return to the plan: we want a polynomial matrix that *deforms* A into matrices of this form at the points $t = 1, \dots, \ell$.

By subadditivity of rank, a sum of a few low-rank matrices still has low rank. A sum of s-block-diagonal matrices, however, may have no useful block structure, since their blocks may be unrelated. For a sum to keep the small-blocks plus low-rank structure, the summands must be block diagonal with respect to a *common* pair of partitions $\{I_a\}$ and $\{J_a\}$.

Every Δ-sparse matrix A is a sum of at most Δ many 1-sparse matrices, but this decomposition is not useful, since the block structures of the summands are unrelated. Instead, we carefully construct a decomposition into more, but still few, 1-sparse matrices, so that at each of the points $t = 1, \dots, \ell$ all summands can be deformed to become block diagonal with respect to a *common partition*. To this end, we colour the rows and columns of A by elements of $G = (\mathbb{F}_2^k)^\ell$. This lets us write A as a sum of at most $|G|$ many 1-sparse matrices, each supported on a copy of a translation matrix $T_\gamma(x, y) = \mathbf{1}_{\{x+y=\gamma\}}$, with rows and columns indexed by their colours. The ℓ points match the ℓ coordinates of G: we deform each translation matrix so that at the point $t = i$ its blocks are determined by the ith *coordinate of the colours alone*, and not by γ.

Translation matrices. The key tool is a simple observation: on most of $\{0,1\}^\ell$, the parity of $z_1 + \cdots + z_\ell$ agrees with a polynomial of degree about $\sqrt{\ell}$.

> **Proposition 3.1.** *Let $0 < \varepsilon < 1$ and $d = 2\sqrt{\ell \log(2/\varepsilon)}$. There is a rational polynomial p of degree at most d such that*
>
> $$(-1)^{z_1 + \cdots + z_\ell} = p\big(z_1 + \cdots + z_\ell\big)$$

for all but at most an ε fraction of the points $z \in \{0,1\}^\ell$.

Proof. Let p interpolate $p(m) = (-1)^m$ at the at most $d+1$ integers $m \in [\ell/2 - d/2,\ \ell/2 + d/2]$, so $\deg p \le d$. The equality fails only if $|z_1 + \cdots + z_\ell - \ell/2| > d/2$, which by Hoeffding's inequality happens for at most a $2e^{-d^2/(2\ell)} \le \varepsilon$ fraction of points z. □

For integers $k, \ell \ge 0$, let $G \coloneqq (\mathbb{F}_2^k)^\ell$. We write $a = (a_1, \dots, a_\ell)$ for the elements of G, with $a_i \in \mathbb{F}_2^k$. For $\gamma \in G$, the *translation matrix* T_γ is the $G \times G$ permutation matrix

$$T_\gamma(a,b) = \mathbf{1}_{\{a+b=\gamma\}},$$

with addition in G. It pairs each a with $a + \gamma$.

For $i \in [\ell]$, we say that a $G \times G$ matrix K is *i-nice* if $K(a,b) = 0$ whenever $a_i \ne b_i$. Such a matrix is block diagonal with respect to the partitions

$$I_\sigma = J_\sigma = \{a \in G : a_i = \sigma\} \qquad (\sigma \in \mathbb{F}_2^k).$$

These partitions do not depend on γ, which makes them compatible across different γ.

Lemma 3.2 (Deformation for translation matrices)**.** *Let $0 < \varepsilon < 1$ and $d = 2\sqrt{\ell \log(2/\varepsilon)}$. For every $\gamma \in G$ there is a matrix polynomial $T_\gamma(t)$ of degree at most d such that*

$$T_\gamma(0) = T_\gamma, \qquad T_\gamma(i) = K_{\gamma,i} + E_\gamma \quad (i \in [\ell]),$$

where each $K_{\gamma,i}$ is i-nice and $\operatorname{rank} E_\gamma \le \varepsilon |G|$.

Proof. For $\xi, z \in G$ write $\xi \cdot z = \sum_{j=1}^\ell \xi_j \cdot z_j \in \mathbb{F}_2$, where $\xi_j \cdot z_j$ is the standard inner product on $\mathbb{F}_2^k$. By Fourier inversion,

$$T_\gamma(x,y) = \mathbf{1}_{\{x+y=\gamma\}} = \frac{1}{|G|} \sum_{\xi \in G} (-1)^{\xi\cdot\gamma}\, (-1)^{\xi\cdot(x+y)} = \frac{1}{|G|} \sum_{\xi \in G} (-1)^{\sum_{j=1}^\ell |\xi_j \cdot \gamma_j|}\, (-1)^{\xi\cdot(x+y)}.$$

For uniform $\xi \in G$, the bits $|\xi_j \cdot \gamma_j|$ with $\gamma_j \ne 0$ are independent and uniform, while those with $\gamma_j = 0$ vanish. Applying Proposition 3.1 to the bits with $\gamma_j \ne 0$ gives a polynomial p of degree at most d such that

$$e(\xi) \coloneqq (-1)^{\sum_{j=1}^\ell |\xi_j \cdot \gamma_j|} - p\left(\sum_{j=1}^\ell |\xi_j \cdot \gamma_j| \right)$$

is supported on at most $\varepsilon |G|$ points. Define the $G \times G$ polynomial matrix

$$T_\gamma(t)(x,y) \coloneqq \frac{1}{|G|} \sum_{\xi \in G} \left[p\left(\sum_{j=1}^\ell (1 - t/j)\, |\xi_j \cdot \gamma_j| \right) + e(\xi) \right] (-1)^{\xi\cdot(x+y)}.$$

We verify the three claims of the lemma. The entries of $T_\gamma(t)$ have degree at most d in t, and at $t = 0$ the bracket equals $(-1)^{\xi\cdot\gamma}$, so $T_\gamma(0) = T_\gamma$.

For the low-rank part, let E_γ be the sum of the terms with $e(\xi)$. Since $(-1)^{\xi\cdot(x+y)} = (-1)^{\xi\cdot x}(-1)^{\xi\cdot y}$, each term is a rank-one matrix, so $\operatorname{rank} E_\gamma \le |\operatorname{supp}(e)| \le \varepsilon |G|$.

For the nice part, let $K_{\gamma,i} = T_\gamma(i) - E_\gamma$. At $t = i$ the weight $1 - i/i$ of block i vanishes, so the argument of p does not depend on ξ_i, and i-niceness follows by summing over the ith coordinate:

$$\sum_{\xi_i \in \mathbb{F}_2^k} (-1)^{\xi_i \cdot (x_i + y_i)} = 2^k \, \mathbf{1}_{\{x_i = y_i\}}. \qquad \square$$

Sparse matrices. To decompose A into copies of translation matrices, we first need a suitable colouring of its rows and columns.

> **Lemma 3.3** (Colouring). *Consider a bipartite graph with maximum degree Δ and n-vertex parts U and V. For every $4\Delta^2\sqrt{n} \le Q \le n$, there are colourings $a\colon U \to [Q]$ and $b\colon V \to [Q]$ such that*
>
> *(i) each colour appears at most $D := \lceil 2n/Q \rceil$ times in each part;*
>
> *(ii) distinct edges (u, v) receive distinct pairs $(a(u), b(v))$.*

Proof. We colour greedily. At any point, at most $n/D \le Q/2$ colours have reached the cap D in a part, so (i) forbids at most $Q/2$ colours.

First we colour the vertices in U so that vertices with a common neighbour receive different colours and (i) holds. This is possible since at most $\Delta^2 \le Q/4$ vertices share a neighbour with any $u \in U$.

Next we colour the vertices in V so that $v, v' \in V$ receive different colours whenever they have neighbours u, u' with $a(u) = a(u')$. This is possible since it forbids at most $\Delta^2 D \le Q/4$ colours for every vertex.

For (ii), let $(u, v) \ne (u', v')$ be edges. If $v = v'$, then u, u' share a neighbour, so $a(u) \ne a(u')$; if $v \ne v'$ and $a(u) = a(u')$, the rule for V gives $b(v) \ne b(v')$. $\square$

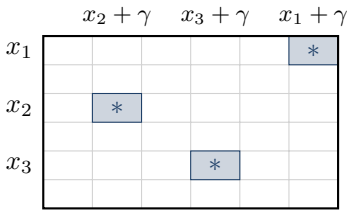
Next lemma shows that deforming each group through its translation matrix, we can obtain the deformation of A used in the proof of Theorem 1.1.

> **Lemma 3.4** (Deforming a sparse matrix). *Let A be an $n \times n$ rational matrix with at most Δ nonzero entries in each row and column, let $k, \ell \ge 1$ be integers with $4\Delta^2\sqrt{n} \le 2^{k\ell} \le n$, and let $0 < \varepsilon < 1$. There is a rational matrix polynomial $A(t)$ of degree at most $2\sqrt{\ell \log(2/\varepsilon)}$ such that*
>
> $$A(0) = A, \qquad A(i) = B_i + E \quad (i \in [\ell]),$$
>
> *where $\operatorname{rank} E \le \varepsilon 4^{k\ell}$ and each B_i is block diagonal with blocks of at most $3n/2^k$ rows and columns.*

Proof. Apply Lemma 3.3 to the support bigraph of A, using $G = (\mathbb{F}_2^k)^\ell$ as the set of colours, and let $a(u)$ and $b(v)$ be the colours of row u and column v. For $\gamma \in G$, let

$$\mathcal{E}_\gamma := \{(u, v) : A_{uv} \ne 0,\ a(u) + b(v) = \gamma\},$$

and let A_γ be the matrix obtained from A by keeping only the entries in $\mathcal{E}_\gamma$, so that $A = \sum_{\gamma \in G} A_\gamma$. The entries in $\mathcal{E}_\gamma$ have colour pairs of the form $(x, x + \gamma)$. Since distinct entries of $\mathcal{E}_\gamma$ have distinct colour pairs, their row colours are all distinct, as are their column colours. Thus $\mathcal{E}_\gamma$ is a matching, and the row and column colours, places $\mathcal{E}$ on the non-zero entries of $T_\gamma(x, y) = \mathbf{1}_{\{x+y=\gamma\}}$, in $G \times G$.



The stars mark the entries of $\mathcal{E}_\gamma$. On the matched rows and columns, they form a permutation matrix.

Let U_γ and V_γ be the $n \times G$ matrices with

$$U_\gamma(u, a(u)) = A_{uv}, \qquad V_\gamma(v, b(v)) = 1 \qquad \text{for each } (u,v) \in \mathcal{E}_\gamma,$$

and all other entries 0. Each row and column of U_γ and of V_γ has at most one nonzero entry. Since $T_\gamma(x,y) = \mathbf{1}_{\{y=x+\gamma\}}$,

$$(U_\gamma T_\gamma V_\gamma^{\mathsf{T}})(u,v) = U_\gamma(u,a(u))\, V_\gamma(v, a(u)+\gamma) = A_{uv}\mathbf{1}_{\{(u,v)\in\mathcal{E}_\gamma\}}.$$

Hence $U_\gamma T_\gamma V_\gamma^{\mathsf{T}} = A_\gamma$, and

$$A = \sum_{\gamma \in G} U_\gamma T_\gamma V_\gamma^{\mathsf{T}}.$$

Let $T_\gamma(t)$ be the deformation of T_γ from Lemma 3.2, and set

$$A(t) \coloneqq \sum_{\gamma \in G} U_\gamma T_\gamma(t) V_\gamma^{\mathsf{T}}.$$

Then $A(0) = A$, and the entries of $A(t)$ have degree at most $2\sqrt{\ell \log(2/\varepsilon)}$. At $t = i$, write $T_\gamma(i) = K_{\gamma,i} + E_\gamma$ as in Lemma 3.2, and put

$$B_i = \sum_{\gamma \in G} U_\gamma K_{\gamma,i} V_\gamma^{\mathsf{T}}, \qquad E = \sum_{\gamma \in G} U_\gamma E_\gamma V_\gamma^{\mathsf{T}},$$

so that $A(i) = B_i + E$ and by the subadditivty of the rank

$$\operatorname{rank} E \le \sum_{\gamma \in G} \operatorname{rank} E_\gamma \le \varepsilon |G|^2 = \varepsilon 4^{k\ell}.$$

For the blocks of B_i: the (u,v) entry of $U_\gamma K_{\gamma,i} V_\gamma^{\mathsf{T}}$ is a multiple of $K_{\gamma,i}(a(u), b(v))$, which vanishes when $a_i(u) \neq b_i(v)$ by i-niceness. This holds for every γ with the same condition $a_i(u) = b_i(v)$, so the sum B_i is block diagonal with respect to the partitions

$$I_\sigma = \{u : a_i(u) = \sigma\}, \qquad J_\sigma = \{v : b_i(v) = \sigma\} \qquad (\sigma \in \mathbb{F}_2^k).$$

Exactly $|G|/2^k$ colours have ith symbol σ, and by (i) each colour is used at most $\lceil 2n/|G| \rceil$ times in each part, so

$$|I_\sigma|,\ |J_\sigma| \le \frac{|G|}{2^k}\left\lceil \frac{2n}{|G|} \right\rceil \le \frac{3n}{2^k}. \qquad \square$$

4 Computing the derivatives

Let $P(t) = \operatorname{per}(A(t))$, where $A(t)$ is an $n \times n$ matrix whose entries are polynomials of degree at most $d \le n$. We want to compute the derivatives $P(i), P'(i), \ldots, P^{(h-1)}(i)$.

Proposition 4.1. *Let $1 \le h \le n/2$. Suppose that the permanent of every square submatrix of $A(i)$ can be computed in T arithmetic operations. Then $P^{(b)}(i)$ for all $0 \le b < h$ can be computed in number of arithmetic operations*

$$T\,2^{O(h\log(n/h))}.$$

Proof. For $S, T \subseteq [n]$ with $|S| = |T|$, let $P_{S,T}(t) := \operatorname{per}\big(A(t)_{\bar{S}\times\bar{T}}\big)$ be the permanent after deleting the rows in S and the columns in T. Differentiating the permanent term by term gives

$$P'_{S,T}(t) = \sum_{\substack{u\notin S\\ v\notin T}} A'_{uv}(t)\, P_{S\cup\{u\},\,T\cup\{v\}}(t), \tag{5}$$

and differentiating (5) another $b-1$ times gives

$$P^{(b)}_{S,T}(i) = \sum_{\substack{u\notin S\\ v\notin T}} \sum_{j=1}^{b} \binom{b-1}{j-1} A^{(j)}_{uv}(i)\, P^{(b-j)}_{S\cup\{u\},\,T\cup\{v\}}(i). \tag{6}$$

We compute $P^{(b)}_{S,T}(i)$ for all (S,T) with $|S| = |T|$ and all $b \ge 0$ with $|S| + b < h$, in increasing order of b: for $b = 0$ by assumption, and for $b \ge 1$ by (6), whose right-hand side involves only pairs (S',T') and orders $b' < b$ with $|S'| + b' \le |S| + b < h$. Taking $S = T = \varnothing$ gives $P(i), \dots, P^{(h-1)}(i)$.

Since $h \le n/2$, there are at most $\binom{n}{\le h-1}^2 \le (en/h)^{2h} = 2^{O(h\log(n/h))}$ pairs (S,T), each requiring one submatrix permanent and $\operatorname{poly}(n)$ further operations. This gives the stated bound. □

5 Recovering the permanent

Proof of Theorem 1.1. Let n be sufficiently large. We apply Lemma 3.4 with

$$q = 2^k = \Theta\big((\log n)^2\big), \qquad \ell = \Theta\Big(\frac{\log n}{\log\log n}\Big), \qquad \varepsilon = q^{-4},$$

where k and ℓ are chosen so that $4\Delta^2\sqrt{n} \le q^\ell = O(\sqrt{n})$. This gives a deformation $A(t)$ of A of degree

$$d = O\big(\sqrt{\ell\log(1/\varepsilon)}\big) = O(\sqrt{\log n}).$$

Each B_i is s-block-diagonal with $s = O(n/(\log n)^2)$, and E has rank $r = O(n/(\log n)^4)$. By Proposition 2.1, the permanent of every square submatrix of $A(i)$ can therefore be computed with

$$(r+2)^{O(s)}\, n^{O(r+1)} = 2^{O((s+r)\log n)} = 2^{O(n/\log n)}$$

arithmetic operations.

Let $P(t) = \operatorname{per}(A(t))$, a polynomial of degree at most nd, and let

$$h = \Big\lfloor \frac{nd}{\ell} \Big\rfloor + 1 = O\Big(\frac{n\log\log n}{\sqrt{\log n}}\Big),$$

so that $\ell h > nd$. By Proposition 4.1, computing $P(i), \dots, P^{(h-1)}(i)$ for all $i \in [\ell]$ takes

$$2^{O(n/\log n)} \cdot 2^{O(h\log(n/h))} = 2^{O(n(\log\log n)^2/\sqrt{\log n})}$$

arithmetic operations, since $\log(n/h) = O(\log\log n)$. These values determine P: Writing $P(t) = \sum_{j\le nd} c_j t^j$, the values $P^{(b)}(i)$ are linear forms in the coefficients c_j. Since $\ell h > nd$, the resulting system has a unique solution, and solving it recovers P, and hence $P(0) = \operatorname{per}(A)$. □

References

[DHM+14] Holger Dell, Thore Husfeldt, Dániel Marx, Nina Taslaman, and Martin Wahlén, *Exponential time complexity of the permanent and the Tutte polynomial*, ACM Transactions on Algorithms **10** (2014), no. 4, 21:1–21:32.

[IP01] Russell Impagliazzo and Ramamohan Paturi, *On the complexity of k-SAT*, Journal of Computer and System Sciences **62** (2001), no. 2, 367–375.

[IPZ01] Russell Impagliazzo, Ramamohan Paturi, and Francis Zane, *Which problems have strongly exponential complexity?*, Journal of Computer and System Sciences **63** (2001), no. 4, 512–530.

[O'D26] Ryan O'Donnell, *Some exposition of "The Exponential Time Hypothesis Is False"*, October 2026, Manuscript.

[Ope26] OpenAI, *The exponential time hypothesis is false*, October 2026, Manuscript.