

COUNTING HAMILTONIAN CYCLES IN SPARSE GRAPHS IN SUBEXPONENTIAL TIME, ASSUMING THE NEW #SAT ALGORITHMS

VAIBHAV TULSYAN

ABSTRACT. In October 2026 two manuscripts claimed refutations of the Exponential Time Hypothesis by deterministic algorithms for $\#k$ -SAT, each with a Lean formalisation of its top-level statement. Neither result applies directly to connectivity problems such as Hamiltonicity, because no linear-size CNF encoding of “this edge set is a single cycle” is known. This paper derives a corollary of the Anthropic manuscript: assuming three of its internal results (a network-to-clique reduction, an algorithm for cliques of all levels, and the existence of its “pieces”), the number of Hamiltonian cycles of a graph with n vertices and at most Dn edges can be computed exactly and deterministically in time $O(2^{\delta n})$ for every $\delta > 0$, and in time $2^{O(\max(m,n)/\sqrt{\log n})}$ for a graph with m edges. A rank-encoded variant of the construction gives $2^{O(n \log \bar{d}/\sqrt{\log n})}$ for average degree $\bar{d}$. Deterministic decision, search and Hamiltonian-path counting follow, and so do exact sparse TSP with polynomially bounded weights, exact counting of directed Hamiltonian cycles, and ATSP. The result is not a standalone breakthrough; it stands or falls with the claimed refutation. The ingredients are known: the folklore matrix-tree identity on 2-factors, the fermionic (Berezin) matrix-tree theorem of Caracciolo, Jacobsen, Saleur, Sokal and Sportiello, and the blob partitions of Cut&Count. What is new is the observation that connectivity can be encoded with $O(1)$ bits per edge through a Grassmann determinant whose only non-local part is a sign, and that this sign costs only $O(g^3)$ bits for g blobs, so the whole construction is a network in the sense of the manuscript, whose value is $n \cdot \#\text{HC}(G)$. Every theorem we claim is formally verified in Lean, end to end, on top of the manuscript’s own Lean development: the three counting theorems, the decision, search and path corollaries, sparse TSP, directed Hamiltonian cycles and ATSP, using only the standard axioms, with fresh kernel replays. The Lean sources are at <https://github.com/xennygrimmato/eth-hamiltonicity>. We also record weaker results that rest only on the machine-checked top-level theorems: the parity of $\#\text{HC}(G)$ for every graph in time $2^{O((m+n)/\sqrt{\log(m+n)})}$ (machine-checked in Lean, with the hypotheses discharged by the labs’ own Lean proofs), and one-sided randomized decision. We describe what is proved, what is checked numerically, what is formalised, what is assumed, and what the limitations are.

CONTENTS

1.	Introduction	2
2.	The network theorem of [A]	7
3.	Grassmann algebra	8
4.	Spanning trees and Hamiltonian cycles	9
5.	Blobs, pieces and the sign rule	11
6.	The network	14
7.	The algorithm and its running time	15
8.	Counting with logarithmic cost in the degree	17
9.	Decision, search and paths	20
10.	Results that rest only on the top-level theorems	21
11.	Extensions (sketched)	22

Date: October 2026.

2020 Mathematics Subject Classification. 68Q25, 68W40, 05C45, 05C30, 15A75.

Key words and phrases. Hamiltonian cycle, exact counting, Exponential Time Hypothesis, Grassmann algebra, matrix-tree theorem, Cut&Count, tensor networks.

12. Verification and formalisation status	24
13. Open problems	27
Acknowledgements	28
References	28

1. INTRODUCTION

1.1. Background. The Exponential Time Hypothesis (ETH) of Impagliazzo and Paturi [27] asserts that there is $\delta > 0$ such that 3-SAT on n variables cannot be solved in time $O(2^{\delta n})$. Together with the sparsification lemma [28] it underlies a large body of conditional lower bounds [33, 18]. Among them is the statement that, unless ETH fails, Hamiltonian Cycle has no $2^{o(n+m)}$ -time algorithm on graphs with n vertices and m edges [18, Ch. 14] (see also [33, Thm. 3.3], stated there as the absence of a $2^{o(n)}$ -time algorithm for Hamiltonian Path).

In October 2026 two manuscripts appeared, by OpenAI [35] and by Anthropic [15], each claiming a deterministic algorithm that counts the satisfying assignments of a k -CNF in subexponential time, and hence a refutation of ETH. Both are accompanied by Lean 4 formalisations of their top-level statements. We stress that these are *claimed* results that are, at the time of writing, still under review by the community. We write [A] for the Anthropic manuscript [15]. Its Main Theorem gives one deterministic algorithm that counts the models of a k -CNF with m clauses on n variables in time

$$2^{O(\min\{km/\sqrt{\log m}, k^2 n \log \log n/\sqrt{\log n}\})}$$

with an absolute constant in the O . The bound of [35] that is machine-checked is $2^{O_k(\rho(n))}$ with $\rho(N) = N(\log \log N)^2/\sqrt{\log N}$.

A refutation of ETH does not by itself yield fast algorithms for connectivity problems. ETH lower bounds for Hamiltonicity come from linear-size reductions *from* 3-SAT *to* Hamiltonian Cycle; an algorithm needs the reverse direction, an encoding of Hamiltonicity as a counting problem on $O(n)$ Boolean variables with short clauses. Connectivity is a global property and no such encoding is known. Accordingly, the consequences listed by both labs (Independent Set, Vertex Cover, Dominating Set, colouring, circuit lower bounds) are all *local* problems.

1.2. Results. Throughout, $G = (V, E)$ is a finite simple undirected graph with $n = |V|$ and $m = |E|$, and $\#HC(G)$ is the number of Hamiltonian cycles of G , each counted once as a subgraph (so $\#HC(K_4) = 3$); $\#HC(G) = 0$ if $n \leq 2$. Running times are counted as in [A]: operations on words long enough to hold an address, plus table queries. All logarithms are to base 2.

Our main results are corollaries of [A]: they are conditional on three results of [A], which we recall precisely in Section 2, and they inherit whatever doubt attaches to those results. If [A] is wrong, the theorems below lose their proofs.

Hypothesis (H_A). *The following statements of [A] hold as stated there: Lemma 8, case (i) (the reduction from general networks to cliques, Theorem 2.4); Theorem A(a),(b) (one algorithm for the (K_ℓ, λ) -cliques of all levels ℓ , Theorem 2.5); and Theorem B (pieces with parameters $(1/\sqrt{2}, c)$, Theorem 2.6).*

For Theorem 1.1 it suffices to have the pieces of [A, Lemma 2] (parameter 11/12) in place of Theorem B, or alternatively the main-text pair [A, Lemma 1, Proposition 3] (Theorem 2.7). Nothing else of [A] is used; in particular we do not use its CNF Main Theorem, which does not suffice for our purposes (the fermionic sign below is not local at the level of vertices).

Theorem 1.1 (Exact counting, fixed density). *Assume (H_A). For every real $\delta > 0$ and every $D \geq 1$ there is a deterministic algorithm that, given a graph G with n vertices and at most Dn edges, outputs $\#HC(G)$ within $O(2^{\delta n})$ operations.*

Theorem 1.2 (Exact counting, explicit rate). *Assume (H_A) . There are an absolute constant C^* and one deterministic algorithm that, given any graph G with $n \geq 2$ vertices and m edges, outputs $\#HC(G)$ within $2^{C^* \max(m,n)/\sqrt{\log n}}$ operations. In particular it runs in time $2^{O(Dn/\sqrt{\log n})}$ on graphs with $m \leq Dn$, where the O -constant is absolute and D may depend on n .*

When the density grows with n , a variant of the network gives a much better dependence on the degree. Write $\bar{d} := 2m/n$ for the average degree and Δ for the maximum degree.

Theorem 1.3 (Exact counting, maximum/average degree; machine-checked). *Assume (H_A) . There are an absolute constant $C^\#$ and one deterministic algorithm that, given any graph G with $n \geq 2$ vertices, average degree $\bar{d}$ and maximum degree Δ , outputs $\#HC(G)$ within*

$$2^{C^\# n \log(2e \max(\bar{d}, 2))/\sqrt{\log n}} \leq 2^{C^\# n \log(2e \max(\Delta, 2))/\sqrt{\log n}}$$

operations. In particular it runs in time $2^{o(n)}$ whenever $\log \bar{d} = o(\sqrt{\log n})$, for example for $\bar{d} = (\log n)^{O(1)}$ or $\bar{d} = 2^{(\log n)^{0.49}}$.

Theorem 1.3 is machine-checked: the Lean theorem `theorem3` gives one Turing machine that outputs $\#HC(G)$ in binary within $\lfloor C \cdot 2^{C n \log_2(2m/n+2)/\sqrt{\log_2(n+2)}} \rfloor$ steps for every graph G with n vertices and m edges, and `theorem3\_maxdeg` gives the same with $\log_2(\Delta+2)$; both use the standard axioms only (Section 12). For bounded $\bar{d}$ it gives the same order $2^{O(n/\sqrt{\log n})}$ as **Theorem 1.2**; it is stronger when $\bar{d} \rightarrow \infty$, because a pair of blobs then shares only the *set* of used crossing edges between them, rank-encoded, and the labels of those edges, so the total number of shared bits is about $2n \log(2e\bar{d})$ instead of $2 \log_2 5 \cdot m \approx 2.32 \bar{d}n$. It does not reach dense graphs: for $\bar{d} = \Theta(n)$ the bound is $2^{O(n\sqrt{\log n})}$, above 2^n (Section 1.7).

Corollary 1.4 (Decision, search and paths; machine-checked). *Assume (H_A) . Within time bounds of the same form as **Theorems 1.1** and **1.2** (with a larger constant C), one can deterministically decide whether G is Hamiltonian, output a Hamiltonian cycle if one exists, and, for $n \geq 2$, count, decide and find Hamiltonian paths. The search machine makes n^2+1 calls to the counting machine, deleting candidate edges in a fixed order; Hamiltonian paths of G correspond bijectively to Hamiltonian cycles of G plus an apex vertex.*

In Lean these are `decision\_TM`, `search\_TM` (fixed density) and `decision\_TM\_rate`, `search\_TM\_rate` (rate form), and for paths `hamPaths\_count\_TM`, `hamPaths\_decision\_TM`, `hamPaths\_search\_TM` and their `\_rate` versions, with the bijection `card\_hamPaths\_eq`; all use the standard axioms only.

The weighted version is the sparse travelling-salesman problem; it is machine-checked as well.

Theorem 1.5 (Sparse TSP; machine-checked). *Assume (H_A) . For every real $\delta > 0$ and every $D \geq 1$ there are a deterministic algorithm and a constant C such that, given a graph G with n vertices and at most Dn edges and integer edge weights $w: E \rightarrow \{0, \dots, W\}$, the algorithm outputs within $C \cdot 2^{\delta n} (W+2)^C$ steps the integer $n \sum_H 2^{bw(H)}$, $b = n^2 + n$, the sum running over the Hamiltonian cycles H of G . Its base- 2^b digit j is n times the number of tours of weight exactly j ; the minimum tour weight is the position of the lowest non-zero digit, and the number of optimal tours is that digit divided by n . In particular the sparse TSP is solved exactly, with its optimum counted, in time $2^{\delta n} \text{poly}(W)$ for every $\delta > 0$: subexponential for all weights $W = 2^{o(n)}$.*

In Lean this is `theoremTSP` and `theoremTSP\_decode`, built on [A]’s development with the standard axioms only (Section 12). What it does *not* give is as important: the running time grows with the number of edges, so dense and metric instances (complete graphs, Euclidean TSP) are untouched, and the $O^*(2^n)$ dynamic programme of Bellman and of Held and Karp [3, 26] remains essentially the record there (improved only to $2^{n-\Omega(\sqrt{n/\log(nW)})}$ $\text{poly}(W)$ for integer weights up to W [5], and, for bipartite instances, to $O(1.9999^n)$ assuming quadratic-time matrix multiplication [34]);

and weights $W = 2^{\Theta(n)}$ remain open (Section 1.7). The identity behind the theorem is the Kronecker substitution $z = 2^b$ in $n \sum_H z^{w(H)} = \sum_X z^{w(X)} \kappa(X)$, the weighted form of Theorem 5.7.

The next two results are weaker but depend only on the top-level theorems of the labs, which are themselves machine-checked; the first is machine-checked end to end on top of them.

Theorem 1.6 (Parity; machine-checked). *There is one deterministic algorithm that, for every graph G with $n \geq 2$ vertices and m edges, computes $\#HC(G) \bmod 2$ within $2^{O((m+n)/\sqrt{\log(m+n)})}$ steps, assuming the Main Theorem of [A]. Assuming instead the Lean theorem of [35], it does so in time $2^{O(\rho(5m+3n))}$ and polynomial space. Both statements are proved in Lean, and the hypotheses are discharged by the labs' own Lean proofs, rebuilt from source; the resulting theorems depend only on the standard axioms `propext`, `Classical.choice`, `Quot.sound`. The analogous statement holds for directed Hamiltonian cycles in digraphs of bounded in- and out-degree.*

Theorem 1.7 (Randomized decision).

- (a) (Machine-checked, assuming the Lean theorem of [35].) *For every Δ and every $\delta > 0$ there is a randomized algorithm with one-sided error that decides Hamiltonicity of graphs of maximum degree Δ in time $2^{\delta n}$. The same holds for digraphs of bounded in- and out-degree.*

An earlier version also stated a part (b), a randomized $2^{O_D(n(\log \log n)^2/\sqrt{\log n})}$ decision procedure for graphs with $|E| \leq Dn$. It is superseded by Theorem 1.2 (deterministic, and faster) and is no longer claimed; see Theorem 10.3.

Theorem 1.6 and Theorem 1.7(a) are from an earlier stage of this work and are superseded, as statements, by Theorems 1.1 and 1.4; we include them because they rest on far less. In (a), Lean checks soundness, the success fraction and the machine; only the outer loop of independent trials is not modelled.

Section 11 records further consequences of the same template, for example directed Hamiltonian cycles, TSP with polynomial weights, Steiner trees, feedback vertex sets and the Tutte polynomial on the line $y = 1$. These are sketched and checked numerically on small instances; they are *not* fully proved here.

1.3. Prior work. No subexponential algorithm for Hamiltonicity on general sparse graphs was known, consistent with ETH. Planar, bounded-genus and minor-free graphs admit $2^{O(\sqrt{n})}$ algorithms via treewidth. Table 1 lists the best previous bounds in the settings relevant here; all are of the form c^n with $c > 1$. The degree-dependent ones are $(2 - \varepsilon)^n$ with $\varepsilon \rightarrow 0$ as the degree grows; Björklund's $O^*(1.657^n)$ Monte Carlo decision bound [4] does not depend on the degree. The bound of [8] is for TSP (hence decision), by a trimmed Bellman–Held–Karp dynamic programme [3, 26]; it is not stated for counting. For small fixed degree, unconditional counting is much faster than $(2 - \varepsilon)^n$: for example, Hamiltonian cycles of graphs of maximum degree 3 can be counted in $O(2^{3n/8})$ time [24], or in $6^{n/6+o(n)}$ time through the pathwidth bound $\text{pw} \leq n/6 + o(n)$ for cubic graphs [25] and the 6^{pw} algorithm of [10].

Two lines of earlier work are closely related to our technique. Cut&Count [20] reduces connectivity to counting modulo 2: a 2-factor X is paired with the $2^{\text{cc}(X)-1}$ cuts that no edge of X crosses, so non-Hamiltonian 2-factors cancel in pairs. This is the route to Theorems 1.6 and 1.7. Björklund's determinant sums [4] pair cycle covers with determinants to enforce connectivity; the identity of Theorem 4.1 below is of the same folklore kind. The fermionic, or Grassmann, form of the matrix-tree theorem that we use is a special case of the generalisation of Kirchhoff's theorem to non-Gaussian Grassmann integrals by Caracciolo, Jacobsen, Saleur, Sokal and Sportiello [14]. Rank bounds for the matchings-connectivity matrix (over $\mathbb{F}_2$ [19], and over $\mathbb{Q}$ [17], building on Raz and Spieker's bound for its bipartite variant [36]) show that, across one cut, connectivity costs only $O(b)$ bits over $\mathbb{Q}$ rather than $b \log b$; our construction can be read as making this compose over many cuts simultaneously.

TABLE 1. Previous bounds and the bounds of this paper (all of the latter conditional; see the text). Here $\rho(N) = N(\log \log N)^2 / \sqrt{\log N}$.

Setting	Previously best known	This paper
undirected, general, decision	$O^*(1.657^n)$ randomized [4]; deterministic $O^*(2^n)$ [3, 26, 32, 30, 2], $2^{n-\Omega(\sqrt{n/\log n})}$ [5] (both also count)	no gain on dense graphs
bounded degree Δ , decision/TSP	$(2 - \varepsilon_\Delta)^n$ [8]	deterministic $2^{\delta n}$, exact count
maximum degree 3	decision/TSP $O(2^{n/3})$ [24], $O^*(1.2312^n)$ [39]; counting $O(2^{3n/8})$ [24]	deterministic $2^{\delta n}$, exact count
counting, pathwidth pw	$6^{\text{pw}} n^{O(1)}$ [10, Table 1], tight under SETH [17, Thm. 1.1]	$2^{O(m/\sqrt{\log n})}$
counting, average degree $\bar{d}$	$2^{n-\Omega(n/\bar{d})}$ deterministic [9] (stated for digraphs; covers graphs)	$2^{O(n \log \bar{d}/\sqrt{\log n})}$ (Theorem 1.3; machine-checked)
counting, directed, average degree d	$2^{n-\Omega(n/d)}$ [9]	see Section 11 (sketch)
parity, directed, general	$O^*(1.619^n)$ [6]	$2^{O(\rho(n))}$ for bounded degree (Lean)
parity, undirected, all graphs	–	$2^{O((m+n)/\sqrt{\log(m+n)})}$ (Lean)

On the hardness side, Dell, Husfeldt, Marx, Taslaman and Wahlén [22] and Curticapean [16] proved #ETH lower bounds for the permanent, perfect matchings and the Tutte polynomial on sparse graphs, and Brand, Dell and Roth [12] completed the #ETH dichotomy for the Tutte plane by treating the line $y = 1$. Most of these bounds concern local problems and are already contradicted by the labs’ theorems through local encodings; the line $y = 1$ (counting forests) is not local, and we return to it in Section 11.

1.4. Overview of the technique. Section 4 starts from the identity

$$(1.1) \quad n \cdot \#\text{HC}(G) = \sum_{X \text{ a 2-factor of } G} \kappa(V, X),$$

where $\kappa(V, X)$ is the number of spanning trees of (V, X) : a 2-factor has n spanning trees if it is a Hamiltonian cycle and none otherwise. The number of spanning trees is the top coefficient of a product of even elements $1 + q_e$ of a Grassmann algebra, one for each edge (Theorem 4.3). This replaces the colouring filters of Cut&Count, which provably need about $\log n$ bits per vertex to be exact (Theorem 10.2).

We partition V into g blobs. Each internal edge is handled inside its blob. Each crossing edge is either unused or used with one of five monomials of $1 + q_e$; merging two monomials that live in the same blob gives five labels, i.e. $\log_2 5$ bits per crossing edge (Table 2). Moving the pieces of the crossing edges into blob order produces a sign. We show that this sign factors as a product of blob-local signs and an interleaving sign $(-1)^{\sum_{a < c < b < d} \pi_{ab} \pi_{cd}}$, where π_{ab} is the parity of the odd labels on the bundle of edges between blobs a and b (Theorems 5.3 and 5.5). The interleaving sign is paid for by $O(g^3)$ parity-copy bits (Theorem 5.6). The outcome is a network in the sense of [A] with arbitrary integer node tables and about $2 \log_2 5 \cdot m + g^3$ shared bits, whose value is $n \cdot \#\text{HC}(G)$ (Theorem 6.3). [A]’s network theorem then evaluates it (Section 7). One trap is that [A]’s cost

depends on the largest number of bits at one node, so a vertex of degree $\Theta(n)$ must be removed first; we branch on its two cycle edges ([Theorem 7.1](#)). [Section 8](#) replaces the per-edge digits by a rank code of the set of used crossing edges between two blobs, which costs only logarithmically in the degree and needs no branching ([Theorem 1.3](#)).

1.5. What is new here. Little of the above is new in isolation, and we want to be precise about the credits. The identity (1.1), matrix-tree on 2-factors, is folklore. The fermionic matrix-tree theorem, which writes the number of spanning trees as a Berezin integral of $\prod_e (1 + q_e)$, is a special case of the generalised matrix-tree theorem of Caracciolo, Jacobsen, Saleur, Sokal and Sportiello [14]. Partitioning the vertices into blobs and paying for connectivity across the cuts is the pattern of Cut&Count [20]. Björklund’s determinant sums [4] (FOCS 2010, SIAM J. Comput. 2014) pair cycle covers with determinants to enforce connectivity, a closely related pattern, though there the determinant is evaluated over a field of characteristic 2 and the aim is a randomized $O^*(1.657^n)$ bound. The engine that evaluates the final object is [A]’s network theorem; we add nothing to it. The one new step is the observation that connectivity can be encoded with $O(1)$ bits per edge ($\log_2 5$ per crossing edge) by a Grassmann determinant whose only non-local part is the fermionic reordering sign, and that this sign factors so that the non-local remainder costs $O(g^3)$ bits for g blobs. That is exactly what is needed for the construction to fit the network model of [A], with its linear bit budget and arbitrary node tables; the Cut&Count filters do not ([Theorem 10.2](#)), and neither do the $b \log b$ -bit dynamic-programming states over a cut. Everything else is bookkeeping.

1.6. Status of the results. The proofs of [Sections 3 to 7](#) and [9](#) are complete, conditional on (H_A) . They were checked numerically in exact integer arithmetic, and an independent referee re-derived the construction with fresh code and found it sound, conditional on [A]. The mathematical core is machine-checked in Lean. This covers the identity $\sum_\ell \prod_a \sigma_a \chi_a I_a = n \cdot \#HC(G)$ behind [Theorems 5.7](#) and [6.3](#) and the parameter bounds of [Theorem 6.4](#), with the network stated in [A]’s own Lean format. The blob-table procedure is formalised as well, and [Theorems 1.1](#) and [1.2](#) are both formally verified in Lean end to end, as statements about Turing machines built on [A]’s Lean development [38], with only the standard axioms (`theorem1`, `theorem2`); a fresh kernel replay with `leanchecker` has passed for both `theorem1` and `theorem2`. The sparse TSP theorem ([Theorem 1.5](#)) is formally verified at the same level (`theoremTSP`, `theoremTSP\_decode`). [Theorem 1.3](#) is the exception: its proof ([Section 8](#)) is written out and was checked numerically and by an independent AI referee, who found it sound, but it is not formalised. [Section 12](#) gives the details. The bounds are purely asymptotic: as for the SAT algorithms themselves, the crossover with brute force is astronomically large.

1.7. Limitations. The following should be read together with [Section 12](#).

- (a) *Everything is conditional.* The main theorems rest on the claimed ETH refutation of [A], through (H_A) , and on [A]’s Lean formalisation faithfully capturing its printed theorems. Expert review of both is pending. Lean guarantees that [A]’s proofs establish [A]’s formal statements, not that those statements say what the printed ones say.
- (b) *The Lean proofs here were written and reviewed by AI.* They were produced by AI agents and reviewed by an AI coordinator. No human has yet checked that the formal statements say what the printed theorems say. A human reader who wants to do so needs to read about five definitions: `hamCycles` (the set of Hamiltonian cycles being counted, and that it counts each cycle once as a subgraph); `GraphProblems.code` (the input encoding of a graph, and that it is the adjacency matrix); `Outputs` (what it means for the machine to output $\#HC(G)$ in binary); the statements of `theorem1` and `theorem2` themselves, in particular that the step bound is $\lfloor C \cdot 2^{\delta n} \rfloor$ resp. $\lfloor C \cdot 2^{C(m+n)/\sqrt{\log_2(n+2)}} \rfloor$ with C independent of G ; and [A]’s `Net` format together with the hypothesis bundle through which our network is handed to [A]’s theorems, to confirm that nothing beyond [Theorems 2.4 to 2.6](#) is assumed.

- (c) *Sparse graphs only.* The network of [Section 6](#) has about $2 \log_2 5$ bits per edge, so [Theorem 1.2](#) beats 2^n only when $m = o(n\sqrt{\log n})$. The rank-encoded network of [Section 8](#) has about $2 \log(2e\bar{d})$ bits per vertex, so [Theorem 1.3](#) extends this, roughly, to average degree $\bar{d} = 2^{o(\sqrt{\log n})}$. Dense graphs are still not covered: at $\bar{d} \approx \Delta \approx n$ the bound of [Theorem 1.3](#) is $2^{\Theta(n \log n / \sqrt{\log n})}$, which exceeds 2^n . General TSP and the Held–Karp $O^*(2^n)$ bound are untouched.
- (d) *“Previous algorithms were exponential” refers to general sparse graphs.* Planar, bounded-genus, minor-free and bounded-treewidth graphs already had subexponential or single-exponential-in-treewidth algorithms, and on planar graphs the $2^{O(\sqrt{n})}$ bound is better than ours. The gain is on sparse graphs with no such structure, for example bounded-degree expanders.
- (e) *Asymptotic only.* The crossover with brute force is astronomically large; nothing here is of practical use.
- (f) *The extensions are sketched.* [Section 11](#) is checked numerically on small instances and is not proved at the level of the rest of the paper.
- (g) *Citations.* All citations were checked against the primary sources on 30 Oct 2026. In particular, the $6^{\text{pw}} n^{O(1)}$ counting bound in [Table 1](#) is $\tilde{O}(6^{\text{pw}} \text{pw}^{O(1)} n^2)$ in [[10](#), Table 1], and its SETH-tightness is [[17](#), Thm. 1.1]; the other constants in that table are as stated in the cited works.
- (h) *No systematic novelty search has been done.* The credits in the previous subsection are to the works we know; a Grassmann-determinant encoding of connectivity in a bit-bounded model may well exist elsewhere, and we have not searched the literature systematically.

1.8. Organisation. [Section 2](#) recalls the results of [A] that we use. [Section 3](#) collects facts about Grassmann algebras. [Section 4](#) proves the fermionic matrix-tree identity for Hamiltonian cycles. [Section 5](#) splits it over blobs and proves the sign rule. [Section 6](#) builds the network and bounds its parameters. [Section 7](#) proves [Theorems 1.1](#) and [1.2](#), [Section 8](#) proves [Theorem 1.3](#) with a rank-encoded variant of the network, and [Section 9](#) proves [Theorem 1.4](#). [Section 10](#) treats the results that rest only on the top-level theorems. [Section 11](#) sketches extensions, [Section 12](#) describes the verification status, and [Section 13](#) lists open problems.

2. THE NETWORK THEOREM OF [A]

We recall the definitions and results of [A] that we use, in the notation of [A] except where a letter clashes with ours. Conventions of [A]: $\log = \log_2$, $[x = u] \in \{0, 1\}$ and $[K] = \{1, \dots, K\}$.

Definition 2.1 ([A, Definition 1]). A function $F: X \times X' \rightarrow \mathbb{Z}$ *factors through c bits* if there are $f: X \times \{0, 1\}^c \rightarrow \mathbb{Z}$ and $f': X' \times \{0, 1\}^c \rightarrow \mathbb{Z}$ with $F(z, z') = \sum_{\theta \in \{0, 1\}^c} f(z, \theta) f'(z', \theta)$. In particular, for any $h: X \rightarrow \{0, 1\}^c$ and $h': X' \rightarrow \{0, 1\}^c$ the function $[h(z) = h'(z')] = \sum_{\theta} [h(z) = \theta] [h'(z') = \theta]$ factors through c bits.

Definition 2.2 ([A, Definition 2]). Let $P, B, \tau \geq 1$ be integers. A *network on J with parameters P, B, τ* consists of a finite set J of nodes; for each node a a finite set S_a of settings and a function $W_a: S_a \rightarrow \mathbb{Z}$; and for each unordered pair $\{a, b\}$ of distinct nodes a function $\Psi_{a,b}: S_a \times S_b \rightarrow \mathbb{Z}$ (with $\Psi_{b,a}(z_b, z_a) = \Psi_{a,b}(z_a, z_b)$), an integer $c_{a,b} = c_{b,a} \geq 0$ and a factorisation $(f_{a,b}, f_{b,a})$ of $\Psi_{a,b}$ through $c_{a,b}$ bits. The settings of a are encoded as bit strings of one fixed length $P_a \leq P$, and

- (i) $\sum_{b \neq a} c_{a,b} \leq P$ for every $a \in J$;
- (ii) all values of all $W_a, \Psi_{a,b}, f_{a,b}$ are below 2^B in absolute value;
- (iii) an algorithm receives P, B and the numbers $P_a, c_{a,b}$ as input, and can decide membership in S_a and compute any value of $W_a, \Psi_{a,b}, f_{a,b}$ with at most τ operations.

The *value* of the network is $Z = \sum_{z \in \prod_a S_a} \prod_{a \in J} W_a(z_a) \prod_{\{a,b\}} \Psi_{a,b}(z_a, z_b)$, and its *total* is $U = \sum_{(a,b): a \neq b} c_{a,b}$, which counts every shared bit from both ends. By (i), $U \leq |J|P$.

Definition 2.3 ([A, Definition 3]). Let $K \geq 2$ and $\lambda, B \geq 1$. In the (K, λ) -clique with tables $W_1, \dots, W_K$, a setting of node w is a tuple $(z_w(w'))_{w' \neq w}$ of λ -bit strings, $c_{w,w'} = \lambda$ and $\Psi_{w,w'} = [z_w(w') = z_{w'}(w)]$. The tables W_w , with $|W_w| < 2^B$, are queried at unit cost.

Imported Lemma 2.4 ([A, Lemma 8, case (i)]). Let $K \geq 2$, $g = 3K^8$ and $\zeta = 1/(2K)$. Take a network on $[g]$ with parameters P, B, τ and total U , and put

$$L = \left\lceil \frac{U + gP/K}{K^2} \right\rceil, \quad B' = \lceil 2gP/K \rceil + g^2B.$$

If an algorithm outputs the value of every (K, L) -clique with $|W_w| < 2^{B'}$ within T operations, then the value of the network is computed within

$$(2.1) \quad (K^g + T \cdot 2^{2gP/K} \cdot \tau) \cdot (g(P + B))^{C_1}$$

operations, by one procedure, for an absolute constant C_1 .

The bound (2.1) is weaker than the printed form of [A, Lemma 8] by at most a polynomial factor, which is irrelevant below; it is the form proved in the Lean development of [A] as `ExplicitCost.groupingExplicit`.

Imported Theorem 2.5 ([A, Theorem A]). Suppose there are pieces with parameters (ρ, c) in the sense of [A, Definition 7], and let $\alpha' = \log(1/\rho)$. Fix an integer $x_0 \geq 18$ with $2^{x_0} \geq c + 5$ and define

$$K_0 = 2^{x_0}, \quad \zeta_\ell = 2^{-\ell-5}, \quad g_\ell = 2^{3\ell+x_0} K_\ell, \quad K_{\ell+1} = g_\ell^2, \quad e_0 = \frac{1}{2}, \quad e_{\ell+1} = (\rho + 3\zeta_\ell)e_\ell + 3/K_\ell.$$

- (a) $\log K_\ell = (3x_0 + 6)2^\ell - 6\ell - 2x_0 - 6$, so $K_\ell < K_{\ell+1} \leq K_\ell^{10}$; and $\frac{1}{2}\rho^\ell \leq e_\ell \leq 0.85\rho^\ell$, where $\rho^\ell \leq ((3x_0 + 6)/\log K_\ell)^{\alpha'}$.
- (b) One deterministic algorithm, given $\ell \geq 0$, $\lambda, B \geq 1$ and query access to tables with $|W_w| < 2^B$, outputs the value of the (K_ℓ, λ) -clique within

$$(2.2) \quad 2^{e_\ell K_\ell^2 \lambda} \cdot (K_\ell(\lambda + B))^{C_0 K_\ell^2}$$

operations, with C_0 depending only on c .

Imported Theorem 2.6 ([A, Theorem B and Lemma 2]). There are pieces with parameters $(1/\sqrt{2}, c)$ for an absolute constant c . Moreover, [A, Lemma 2] gives pieces with parameters $(11/12, c)$ for every integer $c \geq 8$ large enough.

Remark 2.7 (Alternative route for [Theorem 1.1](#)). [A, Lemma 1] states: if (K, e) is achievable and $g = 3K^8$, every network on $[g]$ is evaluated within $C_1 2^{e(U+gP/K)+2gP/K} (P+B)^{C_1 \tau}$ operations, with $C_1 = C_1(K, e)$; and [A, Proposition 3] gives, for every $\eta > 0$, infinitely many K with (K, η) achievable. This pair can replace (H_A) in the proof of [Theorem 1.1](#).

Remark 2.8. [A]’s Lean development proves [Theorem 2.4](#) and [Theorem 2.5\(b\)](#) as `ExplicitCost.groupingExplicit` and `ExplicitCost.recursionSolves`, and [Theorem 2.6](#) as `PiecesCounted.halfPieces`. There they are instantiated only for the networks that come from CNFs, and Lean’s machine model differs from the printed one (operations are polynomial-time steps or queries, memory is bounded, and K, g, P, B are given in unary). Applying these results to the network of [Section 6](#) is done here on paper only; see [Section 12](#).

3. GRASSMANN ALGEBRA

Let Γ be a finite set of *generators* with a total order $\prec$. The exterior algebra $\Lambda(\Gamma)$ over $\mathbb{Z}$ is the free $\mathbb{Z}$ -module with basis $\{e_S : S \subseteq \Gamma\}$ and product

$$e_S \cdot e_T = 0 \text{ if } S \cap T \neq \emptyset, \quad e_S \cdot e_T = (-1)^{\#\{(s,t) \in S \times T : t \prec s\}} e_{S \cup T} \text{ otherwise.}$$

This is the exterior algebra of $\mathbb{Z}^\Gamma$ [11, Ch. III, §7]; it is associative with unit $e_\emptyset = 1$. Writing γ for $e_{\{\gamma\}}$ we have $\gamma\gamma' = -\gamma'\gamma$ and $\gamma^2 = 0$, and e_S is the product of the elements of S in increasing $\prec$ -order. A product of generators therefore does not depend on $\prec$, and all statements below, being about sums and products of generators, are independent of $\prec$; changing $\prec$ only re-signs the basis.

Parity. $\Lambda = \Lambda^0 \oplus \Lambda^1$, where Λ^p is spanned by the e_S with $|S| \equiv p \pmod{2}$. Elements of $\Lambda^0 \cup \Lambda^1$ are *homogeneous* (so 0 is homogeneous of both parities), and Λ^0 is a subring.

Fact 3.1 (Graded commutativity). *For $x \in \Lambda^p$ and $y \in \Lambda^q$ we have $xy = (-1)^{pq}yx$. In particular Λ^0 is central.*

Proof. For disjoint S, T , every pair in $S \times T$ is counted in exactly one of $\#\{t \prec s\}$ and $\#\{s \prec t\}$, so $e_{SE}T = (-1)^{|S||T|}e_{TES}$. Extend bilinearly. $\square$

Lemma 3.2 (Koszul sign rule). *Let $y_1, \dots, y_M$ be homogeneous of parities $p_1, \dots, p_M$ and let σ be a permutation of $[M]$. Then*

$$y_{\sigma(1)}y_{\sigma(2)} \cdots y_{\sigma(M)} = (-1)^{N(\sigma)} y_1y_2 \cdots y_M, \quad N(\sigma) = \sum_{\substack{s < t \\ \sigma^{-1}(s) > \sigma^{-1}(t)}} p_s p_t,$$

the sum running over the pairs whose relative order σ reverses.

Proof. Induction on the number of reversed pairs. If $\sigma \neq \text{id}$, some adjacent positions $j, j+1$ have $\sigma(j) > \sigma(j+1)$. Exchanging these two factors multiplies the product by $(-1)^{p_{\sigma(j)}p_{\sigma(j+1)}}$ by [Theorem 3.1](#); it removes exactly the pair $\{\sigma(j), \sigma(j+1)\}$ from the set of reversed pairs and changes no other relative order. $\square$

Subalgebras and top coefficients. For $\Gamma' \subseteq \Gamma$, $\Lambda(\Gamma') \subseteq \Lambda(\Gamma)$ is the span of the e_S with $S \subseteq \Gamma'$. Let $|\Gamma'|$ be even and $\omega = \pm e_{\Gamma'}$. For $x \in \Lambda(\Gamma')$ we write $[\omega](x)$ for the coefficient c such that $x - c\omega$ lies in the span of the e_S with $S \subsetneq \Gamma'$. (This is the Berezin integral of x , normalised by ω .)

Lemma 3.3 (Block factorisation of the top coefficient). *Let $\Gamma = \Gamma_1 \sqcup \cdots \sqcup \Gamma_g$ with every $|\Gamma_a|$ even (blocks may be empty). Let $\omega_a = \pm e_{\Gamma_a}$, $\Omega = \omega_1\omega_2 \cdots \omega_g$ and $x_a \in \Lambda(\Gamma_a)$. Then*

$$[\Omega](x_1x_2 \cdots x_g) = \prod_{a=1}^g [\omega_a](x_a).$$

Proof. Since $\Omega = \pm e_\Gamma$, $[\Omega](\cdot)$ is defined. Write $x_a = [\omega_a](x_a)\omega_a + r_a$, where r_a is a combination of e_S with $S \subsetneq \Gamma_a$. Expanding the product, the term with no r_a is $\prod_a [\omega_a](x_a) \cdot \Omega$. Every other term is a combination of products $e_{S_1} \cdots e_{S_g}$ with $S_b \subseteq \Gamma_b$ for all b and $S_a \subsetneq \Gamma_a$ for some a ; such a product is 0 or $\pm e_{\bigcup S_b}$ with $\bigcup S_b \neq \Gamma$, and does not contribute to $[\Omega](\cdot)$. $\square$

4. SPANNING TREES AND HAMILTONIAN CYCLES

Fix $G = (V, E)$ with $n \geq 3$ and a root $r \in V$. A *2-factor* is a set $X \subseteq E$ such that every vertex has degree exactly 2 in (V, X) . We write $\kappa(V, X)$ for the number of spanning trees of the graph (V, X) .

Lemma 4.1. $n \cdot \#\text{HC}(G) = \sum_X \kappa(V, X)$, *the sum over all 2-factors X of G .*

Proof. A 2-factor X makes (V, X) a disjoint union of cycles covering V . If (V, X) is connected it is a single n -cycle, i.e. a Hamiltonian cycle, whose spanning trees are obtained by deleting one of its n edges; so $\kappa(V, X) = n$. If (V, X) is disconnected, $\kappa(V, X) = 0$. Conversely, the edge set of a Hamiltonian cycle is a connected 2-factor, and distinct Hamiltonian cycles have distinct edge sets. $\square$

Generators. Let $\Gamma = \{\bar{\psi}_v, \psi_v : v \in V \setminus \{r\}\}$, in any order. We put $\bar{\psi}_r = \psi_r := 0$ (the zero element), $\omega_v := \bar{\psi}_v \psi_v \in \Lambda^0$, and for $S \subseteq V$ and $e = \{i, j\} \in E$,

$$\Omega_S := \prod_{v \in S \setminus \{r\}} \omega_v, \quad q_e := (\bar{\psi}_i - \bar{\psi}_j)(\psi_i - \psi_j) \in \Lambda^0.$$

The order of the factors of Ω_S is irrelevant since they are even and central, and $\Omega_S = \pm e_{\Gamma_S}$, where Γ_S is the set of generators of the vertices of S . The element q_e does not depend on the orientation of e , since both factors change sign.

Lemma 4.2 (Berezin form of Cauchy–Binet). *Let $N = n - 1$ and $b_1, \dots, b_N \in \mathbb{Z}^{V \setminus \{r\}}$. Put $\bar{\theta}(b) = \sum_u b(u) \bar{\psi}_u$ and $\theta(b) = \sum_u b(u) \psi_u$. Then*

$$[\Omega_V](\bar{\theta}(b_1)\theta(b_1)\bar{\theta}(b_2)\theta(b_2)\cdots\bar{\theta}(b_N)\theta(b_N)) = \det(\mathbf{B})^2,$$

where $\mathbf{B}$ is the $N \times N$ matrix with columns b_k , its rows indexed by $V \setminus \{r\}$ in any fixed order; $\det(\mathbf{B})^2$ does not depend on that order.

Proof. Enumerate $V \setminus \{r\}$ as $u_1, \dots, u_N$.

- (1) By [Theorem 3.2](#), moving each $\bar{\theta}(b_k)$ to the left past $\theta(b_1), \dots, \theta(b_{k-1})$ gives $\prod_k \bar{\theta}(b_k)\theta(b_k) = (-1)^{N(N-1)/2} \bar{\theta}(b_1) \cdots \bar{\theta}(b_N) \cdot \theta(b_1) \cdots \theta(b_N)$.
- (2) Expanding multilinearly, $\bar{\theta}(b_1) \cdots \bar{\theta}(b_N) = \sum_{\varphi: [N] \rightarrow [N]} \prod_k b_k(u_{\varphi(k)}) \bar{\psi}_{u_{\varphi(1)}} \cdots \bar{\psi}_{u_{\varphi(N)}}$. Only bijections φ survive, because $\bar{\psi}_u^2 = 0$, and for those $\bar{\psi}_{u_{\varphi(1)}} \cdots \bar{\psi}_{u_{\varphi(N)}} = \text{sgn}(\varphi) \bar{\psi}_{u_1} \cdots \bar{\psi}_{u_N}$ by [Theorem 3.2](#). Hence

$$\bar{\theta}(b_1) \cdots \bar{\theta}(b_N) = \det(\mathbf{B}) \bar{\psi}_{u_1} \cdots \bar{\psi}_{u_N}, \quad \theta(b_1) \cdots \theta(b_N) = \det(\mathbf{B}) \psi_{u_1} \cdots \psi_{u_N}.$$

- (3) Step (1) applied to the standard vectors gives

$$\bar{\psi}_{u_1} \cdots \bar{\psi}_{u_N} \psi_{u_1} \cdots \psi_{u_N} = (-1)^{N(N-1)/2} \omega_{u_1} \cdots \omega_{u_N} = (-1)^{N(N-1)/2} \Omega_V.$$

The two signs cancel and the product equals $\det(\mathbf{B})^2 \Omega_V$. □

Lemma 4.3 (Fermionic matrix-tree theorem [14]). *For every $X \subseteq E$,*

$$[\Omega_V](\prod_{e \in X} (1 + q_e)) = \kappa(V, X).$$

Proof. The factors are even and commute, so $\prod_{e \in X} (1 + q_e) = \sum_{F \subseteq X} \prod_{e \in F} q_e$. The term of F lies in the span of the e_S with $|S| = 2|F|$, while $\Omega_V = \pm e_{\Gamma}$ with $|\Gamma| = 2N$; so only $|F| = N$ contributes. For $e = \{i, j\}$ we have $q_e = \bar{\theta}(b_e)\theta(b_e)$ with $b_e = \mathbf{1}_i - \mathbf{1}_j$ restricted to $V \setminus \{r\}$ (that is, $b_e = \mathbf{1}_i$ if $j = r$). By [Theorem 4.2](#), F contributes $\det(\mathbf{B}_F)^2$, where $\mathbf{B}_F$ is the reduced incidence matrix of F .

- If F (with $|F| = n - 1$) is not a spanning tree, then (V, F) is disconnected, since a connected graph on n vertices with $n - 1$ edges is a tree. Let C be a component with $r \notin C$. The rows $u \in C$ of $\mathbf{B}_F$ sum to 0: the column of an edge inside C has entries $+1$ and -1 there, and the other columns vanish on C . So $\det \mathbf{B}_F = 0$.
- If F is a spanning tree, then $\det \mathbf{B}_F = \pm 1$, by induction on n . For $n = 1$ the matrix is empty. For $n \geq 2$ pick a leaf $v \neq r$ (a tree has at least two leaves). Row v has one nonzero entry, ± 1 , in the column of its edge f , and expanding along it gives $\det \mathbf{B}_F = \pm \det \mathbf{B}_{F-f}$, the reduced incidence matrix of the spanning tree $F - f$ of $V - v$.

Hence the coefficient is the number of spanning trees $F \subseteq X$, i.e. $\kappa(V, X)$. □

By Cauchy–Binet, [Theorem 4.3](#) is equivalent to Kirchhoff's theorem $\kappa(V, X) = \det L_X^{(r)}$; we do not need this form, but it is checked numerically ([Section 12](#)).

Corollary 4.4.

$$n \cdot \#\text{HC}(G) = \sum_{x \in \{0,1\}^E} \prod_{v \in V} [\deg_x(v) = 2] \cdot [\Omega_V] \left(\prod_{e: x_e=1} (1 + q_e) \right).$$

Proof. Combine [Theorems 4.1](#) and [4.3](#). □

5. BLOBS, PIECES AND THE SIGN RULE

5.1. Conventions. The following conventions are fixed once and for all.

- *Partition.* $\phi: V \rightarrow [g]$ is an arbitrary map, and the *blobs* are $V_a = \phi^{-1}(a)$. Blobs may be empty, and r may lie in any blob. The blob indices $1 < 2 < \dots < g$ form the “line” on which chords interleave.
- *Edges.* An edge is *internal* to a if both ends lie in V_a ; the set of these is $E_{\text{int}}(a)$. The other edges are *crossing*; let $m_\times$ be their number.
- *Crossing order.* The crossing edges are listed in an arbitrary but fixed total order $e_1, \dots, e_{m_\times}$; we write $k < l$ if e_k comes before e_l .
- *Ends.* A crossing edge e_k has a *low end* s_k and a *high end* t_k , with $a_k := \phi(s_k) < b_k := \phi(t_k)$. Its *chord* is $\{a_k, b_k\}$.
- *Incidences.* $\partial(a) := \{k : a \in \{a_k, b_k\}\}$ is the set of crossing edges at a . For $a < b$ let $E(a, b) := \{k : (a_k, b_k) = (a, b)\}$ (the *bundle* of $\{a, b\}$) and $m_{ab} := |E(a, b)|$, and set $m_{ba} := m_{ab}$.
- *Algebras.* Γ_a is the set of generators of $V_a \setminus \{r\}$, $\Lambda_a := \Lambda(\Gamma_a)$ and $\omega_a := \Omega_{V_a}$. Then $\Omega_V = \omega_1 \cdots \omega_g$, and [Theorem 3.3](#) applies to $\Gamma = \bigsqcup_a \Gamma_a$.

5.2. Labels and pieces. The label set is $\mathcal{L} = \{\mathbf{N}, 0, 1, 2, 3\}$, and the *parity* of a label is $\pi(\ell) := [\ell \in \{2, 3\}]$. For the crossing edge e_k with ends $s = s_k$ and $t = t_k$, the pieces $\alpha_\ell(k) \in \Lambda_{a_k}$ and $\beta_\ell(k) \in \Lambda_{b_k}$ are given in [Table 2](#).

TABLE 2. Labels of a crossing edge e_k with low end s and high end t , and their pieces.

label ℓ	$\alpha_\ell(k) \in \Lambda_{a_k}$	$\beta_\ell(k) \in \Lambda_{b_k}$	$\alpha_\ell(k)\beta_\ell(k)$	parity $\pi(\ell)$	meaning
N	1	1	1	0	$e_k \notin X$
0	1	$1 + \bar{\psi}_t \psi_t$	$1 + \bar{\psi}_t \psi_t$	0	$e_k \in X$, terms 1 and $\bar{\psi}_t \psi_t$
1	$\bar{\psi}_s \psi_s$	1	$\bar{\psi}_s \psi_s$	0	$e_k \in X$, term $\bar{\psi}_s \psi_s$
2	$\bar{\psi}_s$	$-\psi_t$	$-\bar{\psi}_s \psi_t$	1	$e_k \in X$, term $-\bar{\psi}_s \psi_t$
3	ψ_s	$\bar{\psi}_t$	$\psi_s \bar{\psi}_t$	1	$e_k \in X$, term $-\bar{\psi}_t \psi_s$

Remark 5.1 (Why five labels). For a used edge, $1 + q_e = 1 + \bar{\psi}_s \psi_s - \bar{\psi}_s \psi_t - \bar{\psi}_t \psi_s + \bar{\psi}_t \psi_t$ has five monomials; together with “unused” this gives six states. The two monomials 1 and $\bar{\psi}_t \psi_t$ both live entirely in the high blob, and label 0 merges them into the single high piece $1 + \bar{\psi}_t \psi_t$. This gives five labels, i.e. $\log_2 5$ bits per crossing edge. (A description as “unused, or one of four bilinear terms” would miss the term 1; the merge above is what makes five labels correct. Other merges work equally well and change only constants.)

Lemma 5.2.

- $\alpha_\ell(k)$ and $\beta_\ell(k)$ are homogeneous of parity $\pi(\ell)$. This includes the cases $s = r$ or $t = r$, in which some pieces are 0 or reduce to 1.
- $\alpha_{\mathbf{N}}(k)\beta_{\mathbf{N}}(k) = 1$ and $\sum_{\ell \in \{0,1,2,3\}} \alpha_\ell(k)\beta_\ell(k) = 1 + q_{e_k}$.

Proof. (a) Read off [Table 2](#). (b) By bilinearity $q_e = \bar{\psi}_s \psi_s - \bar{\psi}_s \psi_t - \bar{\psi}_t \psi_s + \bar{\psi}_t \psi_t$ for any elements, zero included, and $-\bar{\psi}_t \psi_s = \psi_s \bar{\psi}_t$. The four products in the table are $1 + \bar{\psi}_t \psi_t$, $\bar{\psi}_s \psi_s$, $-\bar{\psi}_s \psi_t$ and $\psi_s \bar{\psi}_t$, which sum to $1 + q_e$. $\square$

For a labelling $\eta \in \mathcal{L}^{m_\times}$ write $p_k := \pi(\eta_k)$. Define the *piece of k at a* as $\gamma_{k,a}(\eta) := \alpha_{\eta_k}(k)$ if $a = a_k$ and $\beta_{\eta_k}(k)$ if $a = b_k$, and the ordered product of the pieces at a ,

$$R_a(\eta) := \prod_{k \in \partial(a)}^{\rightarrow} \gamma_{k,a}(\eta) \in \Lambda_a \quad (\text{factors in increasing crossing order } k).$$

Finally, let

$$r_{kl} := \#\{(x, y) \in \{a_k, b_k\} \times \{a_l, b_l\} : y < x\}, \quad \varepsilon(\eta) := (-1)^{\sum_{k < l} p_k p_l r_{kl}}.$$

Proposition 5.3 (Sign rule). *For every partition, every crossing order and every $\eta \in \mathcal{L}^{m_\times}$,*

$$\prod_{k=1}^{m_\times} \alpha_{\eta_k}(k) \beta_{\eta_k}(k) = \varepsilon(\eta) \cdot R_1(\eta) R_2(\eta) \cdots R_g(\eta).$$

Proof. The left side is the product of the sequence $Y = (\alpha(1), \beta(1), \alpha(2), \beta(2), \dots, \alpha(m_\times), \beta(m_\times))$. Give the piece $\alpha(k)$ the key (a_k, k) and the piece $\beta(k)$ the key (b_k, k) . Sorting Y by key, lexicographically, lists blob 1's pieces in increasing k , then blob 2's, and so on; so the sorted product is $R_1 \cdots R_g$. By [Theorem 5.2\(a\)](#) all pieces are homogeneous, and [Theorem 3.2](#) counts the pairs of pieces whose relative order is reversed:

- *Two pieces of the same k .* In Y , $\alpha(k)$ comes first, and its key is also smaller since $a_k < b_k$. Not reversed.
- *A piece x of k and a piece y of l , with $k < l$.* In Y , x comes first. The keys are $(\text{blob}(x), k)$ and $(\text{blob}(y), l)$; since $k < l$, the pair is reversed iff $\text{blob}(y) < \text{blob}(x)$. The product of parities is $p_k p_l$, and the number of such pairs is r_{kl} .

So $R_1 \cdots R_g = (-1)^{\sum_{k < l} p_k p_l r_{kl}} \prod_k \alpha(k) \beta(k)$; multiply both sides by $\varepsilon(\eta) = \pm 1$. $\square$

5.3. Factorisation of the sign.

Lemma 5.4. *If $\{a_k, b_k\} \cap \{a_l, b_l\} = \emptyset$, then $r_{kl} \equiv I(k, l) \pmod{2}$, where*

$$I(k, l) := [a_k < a_l < b_k < b_l \text{ or } a_l < a_k < b_l < b_k]$$

indicates that the chords interleave. In particular $r_{kl} \pmod{2}$ is symmetric and independent of the crossing order.

Proof. Put $Y = \{a_l, b_l\}$, whose elements differ from a_k and b_k . Then $r_{kl} = \#\{y \in Y : y < a_k\} + \#\{y \in Y : y < b_k\} = 2\#\{y \in Y : y < a_k\} + \#\{y \in Y : a_k < y < b_k\}$. The last number is odd iff exactly one endpoint of chord l lies strictly between a_k and b_k , that is, iff the chords interleave. $\square$

Two crossing edges $k < l$ share a blob if their chords meet. For such a pair put $s(k, l) := \min(\{a_k, b_k\} \cap \{a_l, b_l\})$; then $k, l \in \partial(s(k, l))$. Define the *local signs* and the *bundle parities*

$$\sigma_a(\eta) := (-1)^{\sum_{k < l : \text{chords meet, } s(k, l) = a} p_k p_l r_{kl}}, \quad \pi_{ab}(\eta) := \sum_{k \in E(a, b)} p_k \pmod{2} \quad (a < b).$$

Thus σ_a is a function of $\eta|_{\partial(a)}$, and π_{ab} of $\eta|_{E(a, b)}$.

Proposition 5.5 (Sign factorisation).

$$\varepsilon(\eta) = \prod_{a=1}^g \sigma_a(\eta) \cdot (-1)^{\sum_{a < c < b < d} \pi_{ab}(\eta) \pi_{cd}(\eta)}.$$

Proof. Split $\sum_{k < l} p_k p_l r_{kl}$ into the pairs whose chords meet and the pairs whose chords are disjoint.

- *Meeting chords.* Each such pair has exactly one $s(k, l)$, so these pairs give $\prod_a \sigma_a$.
- *Disjoint chords.* By [Theorem 5.4](#) these pairs give, modulo 2, $\sum_{k < l: I(k, l) = 1} p_k p_l$. Take an unordered pair $\{k, l\}$ with disjoint interleaving chords, let $\{a, b\}$ ($a < b$) be the chord with the smaller minimum and $\{c, d\}$ the other; interleaving means $a < c < b < d$. Conversely, every quadruple $a < c < b < d$ with $k \in E(a, b)$ and $l \in E(c, d)$ gives such a pair. This is a bijection, so the sum equals $\sum_{a < c < b < d} (\sum_{k \in E(a, b)} p_k) (\sum_{l \in E(c, d)} p_l) \equiv \sum_{a < c < b < d} \pi_{ab} \pi_{cd} \pmod{2}$. $\square$

Parity copies. Let $Q_a := \{(c, d) : a < c < d \leq g\}$, so $|Q_a| = \binom{g-a}{2}$. For $y \in \{0, 1\}^{Q_a}$ put

$$\chi_a(\eta, y) := (-1)^{\sum_{(c, d) \in Q_a} y_{cd} \sum_{b: c < b < d} \pi_{ab}(\eta)},$$

a function of $\eta|_{\partial(a)}$ and y .

Lemma 5.6. *If $y_{cd}^{(a)} = \pi_{cd}(\eta)$ for all a and all $(c, d) \in Q_a$, then $\prod_a \chi_a(\eta, y^{(a)}) = (-1)^{\sum_{a < c < b < d} \pi_{ab} \pi_{cd}}$.*

Proof. $\sum_a \sum_{a < c < d} \pi_{cd} \sum_{c < b < d} \pi_{ab} = \sum_{a < c < b < d} \pi_{ab} \pi_{cd}$. $\square$

So every interleaving pair of chords $\{a, b\}, \{c, d\}$ with $a < c < b < d$ is charged to node a , which knows π_{ab} , and node a receives a copy of π_{cd} from node c , over the pair $\{a, c\}$.

5.4. Blob factorisation. For $v \in V_a$ let $\deg_\eta(v) := \#\{k \in \partial(a) : v \in \{s_k, t_k\}, \eta_k \neq \mathbf{N}\}$, and define the *blob sums*

$$I_a(\eta) := \sum_{x \in \{0, 1\}^{E_{\text{int}}(a)}} \prod_{v \in V_a} [\deg_x(v) + \deg_\eta(v) = 2] \cdot [\omega_a](\mathcal{F}_a(x) R_a(\eta)),$$

where

$$\mathcal{F}_a(x) := \prod_{e \in E_{\text{int}}(a) : x_e = 1} (1 + q_e) \in \Lambda_a \cap \Lambda^0.$$

I_a depends only on $\eta|_{\partial(a)}$. An empty blob has $I_a = 1$, and so does a blob that consists of r alone and has no edges.

Theorem 5.7 (Blob factorisation). *For every partition and every crossing order,*

$$n \cdot \#\text{HC}(G) = \sum_{\eta \in \mathcal{L}^{m \times}} \varepsilon(\eta) \prod_{a=1}^g I_a(\eta) = \sum_{\eta \in \mathcal{L}^{m \times}} \prod_{a=1}^g \sigma_a(\eta) \chi_a(\eta, \pi(\eta)) I_a(\eta),$$

where $\chi_a(\eta, \pi(\eta))$ means $\chi_a(\eta, y^{(a)})$ with $y_{cd}^{(a)} = \pi_{cd}(\eta)$.

Proof. The second equality is [Theorem 5.5](#) together with [Theorem 5.6](#). For the first:

- (1) Expanding $\prod_a I_a$ gives a sum over $x_{\text{int}} \in \{0, 1\}^{\bigcup_a E_{\text{int}}(a)}$ of the degree indicators times $\prod_a [\omega_a](\mathcal{F}_a R_a)$.
- (2) By [Theorem 3.3](#), since $\mathcal{F}_a R_a \in \Lambda_a$, $\prod_a [\omega_a](\mathcal{F}_a R_a) = [\Omega_V](\prod_a \mathcal{F}_a R_a)$.
- (3) The $\mathcal{F}_a$ are even and central, so $\prod_a (\mathcal{F}_a R_a) = (\prod_a \mathcal{F}_a) \cdot R_1 \cdots R_g$.
- (4) By [Theorem 5.3](#), $\varepsilon(\eta) R_1 \cdots R_g = \prod_k \alpha_{\eta_k}(k) \beta_{\eta_k}(k)$.
- (5) Fix x_{int} and the support $T := \{k : \eta_k \neq \mathbf{N}\}$. The degree indicators say exactly that $x := x_{\text{int}} \cup \mathbf{1}_T$ is a 2-factor, and they do not depend on the labels on T .
- (6) By distributivity and [Theorem 5.2\(b\)](#), $\sum_{\eta: \text{supp } \eta = T} \prod_k \alpha_{\eta_k}(k) \beta_{\eta_k}(k) = \prod_{k \in T} (1 + q_{e_k})$.
- (7) Hence $\sum_{\eta} \varepsilon \prod_a I_a = \sum_{x \in \{0, 1\}^E} [x \text{ is a 2-factor}] [\Omega_V](\prod_{e=1} (1 + q_e)) = n \cdot \#\text{HC}(G)$ by [Theorem 4.4](#). $\square$

Remark 5.8. The interleaving sign matters exactly when odd labels sit on interleaving chords. In practice this happens on disconnected 2-factors whose non-root cycle winds through the blobs in interleaving order; there the odd terms are what cancel κ to 0. Dropping it gives wrong values on a positive fraction of test instances (Table 4, C12). Since every blob carries an even number of odd generators, (π_{ab}) is a cycle-space vector of K_g , which explains why the global sign is nontrivial only on a small fraction of random instances.

6. THE NETWORK

Fix G , r , ϕ and a crossing order as in Section 5. Let $\text{enc}: \mathcal{L} \rightarrow \{0, \dots, 4\}$ be the encoding $\text{enc}(\mathbf{N}) = 0$ and $\text{enc}(\mathbf{j}) = j + 1$. For an integer $t \geq 0$ let $w(t) := \min\{w \geq 0 : 2^w \geq 5^t\} = \lceil t \log 5 \rceil$, and put $w_{ab} := w(m_{ab})$.

Definition 6.1 (The network $\mathcal{N}(G, r, \phi, \text{order})$ on $[g]$).

Settings.: A setting of node a is a pair $z_a = (\eta^{(a)}, y^{(a)})$ with $\eta^{(a)} \in \mathcal{L}^{\partial(a)}$ and $y^{(a)} \in \{0, 1\}^{Q_a}$. It is encoded as the concatenation of

- for each $b \in [g] \setminus \{a\}$, in increasing order, the *block* $\text{block}_b(z_a)$: the w_{ab} -bit binary representation (lowest bit first) of $\sum_{j=1}^{m_{ab}} \text{enc}(\eta_{k_j}^{(a)}) 5^{j-1}$, where $k_1 < \dots < k_{m_{ab}}$ enumerate $E(\min(a, b), \max(a, b))$ in crossing order;
- the bits $y_{cd}^{(a)}$ for $(c, d) \in Q_a$, in lexicographic order.

Thus $P_a := \sum_{b \neq a} w_{ab} + |Q_a|$, and S_a is the set of strings of length P_a each of whose blocks is $< 5^{m_{ab}}$. These are exactly the encodings, and the encoding is a bijection, so $|S_a| = 5^{|\partial(a)|} 2^{|Q_a|}$.

Node functions.: $W_a(z_a) := \sigma_a(\eta^{(a)}) \cdot \chi_a(\eta^{(a)}, y^{(a)}) \cdot I_a(\eta^{(a)})$.

Edges.: For $a < b$,

$$\Psi_{a,b}(z_a, z_b) := \prod_{k \in E(a,b)} [\eta_k^{(a)} = \eta_k^{(b)}] \cdot \prod_{d=b+1}^g [y_{bd}^{(a)} = \pi_{bd}(\eta^{(b)})],$$

and $\Psi_{b,a}(z_b, z_a) := \Psi_{a,b}(z_a, z_b)$. The number of shared bits is $c_{a,b} = c_{b,a} := w_{ab} + (g - b)$, and the factorisation is $f_{a,b}(z_a, \theta) := [h_{ab}(z_a) = \theta]$, $f_{b,a}(z_b, \theta) := [h_{ba}(z_b) = \theta]$, where

$$\begin{aligned} h_{ab}(z_a) &:= (\text{block}_b(z_a), y_{b,b+1}^{(a)}, \dots, y_{b,g}^{(a)}) \in \{0, 1\}^{c_{a,b}}, \\ h_{ba}(z_b) &:= (\text{block}_a(z_b), \pi_{b,b+1}(\eta^{(b)}), \dots, \pi_{b,g}(\eta^{(b)})) \in \{0, 1\}^{c_{a,b}}. \end{aligned}$$

In words: the w_{ab} block bits on $\{a, b\}$ are the base-5 code of the labels of the crossing edges between V_a and V_b ; both ends hold a copy, and Ψ tests that the copies agree. For $a < b$ the remaining $g - b$ bits on $\{a, b\}$ are copies, held by a , of the parities π_{bd} for $d > b$, which node b computes from its own labels on $E(b, d)$.

Lemma 6.2. $(f_{a,b}, f_{b,a})$ factors $\Psi_{a,b}$ through $c_{a,b}$ bits.

Proof. Both ends list $E(a, b)$ in the same order, and the base-5 encoding is injective, so $\text{block}_b(z_a) = \text{block}_a(z_b)$ iff $\eta^{(a)}$ and $\eta^{(b)}$ agree on $E(a, b)$. The remaining coordinates of h_{ab} and h_{ba} agree iff every copy is correct. Hence $\Psi_{a,b} = [h_{ab}(z_a) = h_{ba}(z_b)]$, and Theorem 2.1 applies. $\square$

Theorem 6.3. The value of $\mathcal{N}(G, r, \phi, \text{order})$ is $n \cdot \#\text{HC}(G)$, for every $n \geq 3$, every $\phi: V \rightarrow [g]$, every root r and every crossing order.

Proof. The functions $\Psi_{a,b}$ are 0/1-valued, so a term $\prod_a W_a \prod_{a < b} \Psi_{a,b}$ is nonzero only if all $\Psi_{a,b}$ equal 1. This means: (i) there is $\eta \in \mathcal{L}^{m \times}$ with $\eta^{(a)} = \eta|_{\partial(a)}$ for every a , because each k lies in exactly $\partial(a_k)$ and $\partial(b_k)$, and Ψ_{a_k, b_k} tests agreement on k ; and (ii) $y_{cd}^{(a)} = \pi_{cd}(\eta)$ for all $a < c < d$,

because $\Psi_{a,c}$ tests this and $\pi_{cd}(\eta^{(c)}) = \pi_{cd}(\eta)$. Conversely, every $\eta \in \mathcal{L}^{m \times}$ determines exactly one z with (i) and (ii). So $Z = \sum_{\eta} \prod_a \sigma_a(\eta) \chi_a(\eta, \pi(\eta)) I_a(\eta) = n \cdot \#\text{HC}(G)$ by [Theorem 5.7](#). $\square$

Proposition 6.4 (Parameters). *Let $\mu_a := \sum_{v \in V_a} \deg_G(v)$ and $\mu := \max_a \mu_a$, and suppose that G has minimum degree at least 2. Then $\mathcal{N}(G, r, \phi, \text{order})$ is a network in the sense of [Theorem 2.2](#) with*

$$P := w(\mu) + g + g^2, \quad B := 2\mu + 1, \quad \tau := 2^{1.5\mu} n^{C_2} \quad (C_2 \text{ absolute}),$$

$$U = 2 \sum_{a < b} w_{ab} + 2 \binom{g}{3} \leq 2 \log 5 \cdot m_{\times} + g^2 + \frac{g^3}{3} \leq 2 \log 5 \cdot |E| + g^3,$$

where τ is counted after a polynomial-time preparation. In particular $P \leq \log 5 \cdot \mu + 2g^2$ and $gP \leq \log 5 \cdot g\mu + 2g^3$.

Proof. Counting. Each crossing edge at a has exactly one end in V_a and each internal edge two, so $2|E_{\text{int}}(a)| + |\partial(a)| = \mu_a$. Also $w(t) \leq t \log 5 + [t > 0]$ and $w(t) \geq t \log 5$, so $\sum_{b \neq a} w_{ab} \leq |\partial(a)| \log 5 + (g-1) \leq w(\mu) + g - 1$.

$$P_a \leq P. \quad P_a = \sum_b w_{ab} + \binom{g-a}{2} \leq w(\mu) + g - 1 + g^2/2 \leq P.$$

Condition (i). The copy bits on pairs at a number $\sum_{b>a} (g-b) + \sum_{b<a} (g-a) = \binom{g-a}{2} + (a-1)(g-a) \leq g^2$, so $\sum_{b \neq a} c_{a,b} \leq w(\mu) + g - 1 + g^2 \leq P$.

The total. $U = 2 \sum_{a < b} c_{a,b}$. Use $\sum_{a < b} (g-b) = \sum_b (b-1)(g-b) = \binom{g}{3}$ (choose the middle element of a triple), and $\#\{a < b : m_{ab} > 0\} \leq g^2/2$. Finally $g^2 + g^3/3 \leq g^3$ for $g \geq 2$.

Condition (ii). Ψ and f take values in $\{0, 1\}$, and $|\sigma_a \chi_a| = 1$. In I_a , the product $\mathcal{F}_a(x) R_a(\eta)$ has at most $|E_{\text{int}}(a)|$ factors with at most five signed monomials each and $|\partial(a)|$ factors with at most two each, and there are at most $2^{|E_{\text{int}}(a)|}$ choices of x . So

$$|W_a| \leq 10^{|E_{\text{int}}(a)|} 2^{|\partial(a)|} \leq 2^{\frac{\log 10}{2} \cdot 2|E_{\text{int}}(a)| + |\partial(a)|} \leq 2^{1.67\mu_a} < 2^B.$$

Condition (iii). The preparation stores the graph, ϕ , the crossing order, the lists $E(a, b)$, the numbers w_{ab} and the sets Q_a . Then membership in S_a , encoding and decoding, h_{ab} , Ψ and f take $\text{poly}(n + g)$ operations (base-5 arithmetic on at most P -bit numbers, with $P \leq \text{poly}(n)$); σ_a takes $O(|\partial(a)|^2)$ operations and χ_a takes $O(g^3)$. Finally, I_a loops over at most $2^{|E_{\text{int}}(a)|} \leq 2^{\mu/2}$ vectors x ; for each it multiplies at most μ factors (with at most five monomials each) in Λ_a , represented densely by $2^{|\Gamma_a|} \leq 4^{|V_a|} \leq 2^\mu$ integers of at most $B = O(n)$ bits (here $|V_a| \leq \mu_a/2$ because the minimum degree is at least 2), and reads one coefficient. Altogether $\tau \leq 2^{\mu/2} \cdot 2^\mu \cdot \text{poly}(n)$. $\square$

Remark 6.5. [Theorem 2.2](#) bounds the *largest* node, through P , and not only the total. With unbounded maximum degree one blob may have to carry $\Theta(n)$ crossing-edge bits, and then the factor $2^{2gP/K}$ of (2.1) is not small. [Section 7](#) removes high-degree vertices first; the rank-encoded network of [Section 8](#) avoids the problem instead. Note also that the interleaving sign costs $O(g^3)$ bits in total and at most g^2 bits per node.

7. THE ALGORITHM AND ITS RUNNING TIME

7.1. Preprocessing.

Lemma 7.1 (Degree reduction). *Let $\Theta \geq 2$ and $H := \{v : \deg(v) > \Theta\}$. For each choice $\nu = (\nu_v)_{v \in H}$ of a 2-element subset ν_v of the edges at v , let G_ν be G minus every edge e at a vertex $v \in H$ with $e \notin \nu_v$. Then*

- (a) $\#\text{HC}(G) = \sum_\nu \#\text{HC}(G_\nu)$;
- (b) *there are at most $\prod_{v \in H} \binom{\deg v}{2} \leq n^{2|H|}$ choices, and $|H| < 2m/\Theta$;*
- (c) *every G_ν has vertex set V , at most m edges, and maximum degree at most Θ .*

Proof. Map a Hamiltonian cycle C of G to $\nu(C) := (E(C) \cap \delta(v))_{v \in H}$, where $\delta(v)$ is the set of edges at v . Then C survives in $G_{\nu(C)}$, since every edge of C at $v \in H$ lies in $\nu_v(C)$. Conversely, let C be a Hamiltonian cycle of G_ν . For $v \in H$, the set $E(C) \cap \delta(v)$ has two elements and is contained in $\delta_{G_\nu}(v) \subseteq \nu_v$, so $\nu = \nu(C)$. Thus the pairs (ν, C) with C a Hamiltonian cycle of G_ν correspond bijectively to the Hamiltonian cycles of G . The remaining claims hold because $\sum_v \deg v = 2m$ and the vertices of H keep degree at most 2. $\square$

Lemma 7.2 (Degree-balanced partition). *List V as $v_1, \dots, v_n$ in any order, let $S_t := \sum_{s < t} \deg(v_s)$, and put $\phi(v_t) := \min(g, \lfloor gS_t/(2m) \rfloor + 1)$. Then every blob has $\mu_a \leq 2m/g + \Delta$, where Δ is the maximum degree.*

Proof. ϕ is nondecreasing in t , so each blob is a segment $v_t, \dots, v_{t'}$. If $a < g$, then $(a-1) \cdot 2m/g \leq S_t \leq S_{t'} < a \cdot 2m/g$, so $\mu_a = S_{t'} - S_t + \deg(v_{t'}) < 2m/g + \Delta$. If $a = g$, then $\mu_g = 2m - S_t \leq 2m/g$. $\square$

7.2. The algorithm. On input G the algorithm does the following.

Step 0. If $n \leq 2$, or some vertex has degree less than 2, output 0.

Step 1. Choose a level ℓ (see Sections 7.5 and 7.6). Put $K := K_\ell$ and $g := 3K^8$.

Step 2. Apply Theorem 7.1 with $\Theta := \max(2, \lceil n/g \rceil)$. For each branch G_ν do Steps 3–4; if G_ν has a vertex of degree less than 2, its count is 0 and it is skipped.

Step 3. Take ϕ from Theorem 7.2, any root r and any crossing order, and build $\mathcal{N}(G_\nu, r, \phi, \text{order})$ implicitly: perform the preparation of Theorem 6.4, computing P , B , P_a and $c_{a,b}$ exactly.

Step 4. Run the procedure of Theorem 2.4 on this network, with the algorithm of Theorem 2.5(b) at level ℓ as the clique algorithm; Theorem 2.4 calls it on a (K, L) -clique with B' in the role of B . Divide the result by n .

Step 5. Output the sum over the branches.

Correctness follows from Theorems 2.4, 2.5, 6.3 and 7.1.

7.3. Checking the imported hypotheses. Theorem 2.4 needs a network on $[g]$ with integer parameters $P, B, \tau \geq 1$, total U , and $g = 3K^8$; Theorem 6.4 supplies these. It then needs an algorithm for every (K, L) -clique with $|W| < 2^{B'}$: Theorem 2.5(b) at level ℓ , with $\lambda = L$ and $B = B'$, gives $T = 2^{e_\ell K^2 L} (K(L + B'))^{C_0 K^2}$, and its hypotheses $\lambda, B \geq 1$ hold because $P \geq 1$. Theorem 2.5 needs pieces, supplied by Theorem 2.6. Nothing in the clique algorithm depends on the graph, which reaches it only through the tables that Theorem 2.4 builds from the W_a and $f_{a,b}$. Balancing is done by Theorem 2.4 itself: it searches the K^g groupings for one with at most $2g/K$ nodes per group and all pairwise group totals at most L , which exists by the proof of [A, Lemma 1].

7.4. Bounds for one branch. Write $m := |E(G)|$. Then $|E(G_\nu)| \leq m$ and G_ν has minimum degree at least 2. By Theorem 7.2, $\Delta \leq \Theta \leq n/g + 2$, so $\mu \leq (2m + n)/g + 2$. Using $\log 5 \leq 2.33$ and $g \geq 3$, Theorem 6.4 gives

$$(7.1) \quad \begin{aligned} gP &\leq 2.33(2m + n) + 3g^3, & U &\leq 4.65m + g^3, & B &\leq 2(2m + n)/g + 5, \\ \log \tau &\leq 1.5(2m + n)/g + 3 + C_2 \log n, & L &\leq gP + 1, & B' &\leq 2gP + 1 + g^2 B. \end{aligned}$$

By (2.1) and (2.2), with $e := e_\ell \leq 1$ and $K^2 L \leq U + gP/K + K^2$, the time of Steps 3–4 for one branch is 2^X with

$$(7.2) \quad \begin{aligned} X &\leq 1 + \max \left\{ g \log K, e \left(U + \frac{gP}{K} + K^2 \right) + C_0 K^2 \log(K(L + B')) + \frac{2gP}{K} + \log \tau \right\} \\ &\quad + C_1 \log(g(P + B)) + O(\log n). \end{aligned}$$

There are at most $n^{2|H|} \leq n^{4mg/n}$ branches, because $|H| < 2m/\Theta \leq 2mg/n$.

7.5. Proof of Theorem 1.1. Let $D \geq 1$, $\delta \in (0, 1]$ and $m \leq Dn$, so that $2m + n \leq 3Dn$. By Theorem 2.5(a), $e_\ell \rightarrow 0$ and $K_\ell \rightarrow \infty$, so there is ℓ , depending only on (δ, D) , with $e := e_\ell \leq \delta/(40D)$ and $K := K_\ell \geq 60D/\delta$. Fix it, and put $g = 3K^8$, a constant. By (7.1), the terms of (7.2) satisfy

$$\begin{aligned} eU &\leq \frac{\delta}{40D}(4.65Dn + g^3) \leq 0.12\delta n + g^3, \\ (e+2)gP/K &\leq 3(2.33 \cdot 3Dn + 3g^3)/K \leq 21Dn/K + 9g^3 \leq 0.35\delta n + 9g^3, \\ \log \tau &\leq 4.5Dn/g + O(\log n) \leq 0.075\delta n + O(\log n), \end{aligned}$$

and the remaining terms are $O_{\delta,D}(\log n)$, since g and K are constants and L, B', P, B are $O_{\delta,D}(n)$. So each branch costs $2^{0.55\delta n + O_{\delta,D}(\log n)}$. There are at most $n^{4Dg} = n^{O_{\delta,D}(1)}$ branches, and the preparation is polynomial. The total is $2^{0.55\delta n + O_{\delta,D}(\log n)} = O(2^{\delta n})$. $\square$

The alternative route of Theorem 2.7 gives the same bound: with (K, e) achievable, $e \leq \delta/(40D)$ and $K \geq 60D/\delta$, the exponent of $C_1 2^{e(U+gP/K)+2gP/K} (P+B)^{C_1} \tau$ is bounded by the same three lines.

7.6. Proof of Theorem 1.2. Use the pieces of Theorem 2.6 with $\rho = 1/\sqrt{2}$, so $\alpha' = \frac{1}{2}$, with c and x_0 absolute. For $n < n_0 := K_0^{30}$, an absolute constant, any exact method takes $O(1)$ time. Step 0 disposes of $n \leq 2$ and of minimum degree below 2; otherwise $m \geq n$.

For $n \geq n_0$, let ℓ be the largest index with $K_\ell^{30} \leq n$; K_ℓ is computable from the formula in Theorem 2.5(a). Then $K_{\ell+1} > n^{1/30}$, and $K_{\ell+1} \leq K_\ell^{10}$ gives $n^{1/300} < K := K_\ell \leq n^{1/30}$. So $g = 3K^8 \in [3n^{8/300}, 3n^{8/30}]$, and by Theorem 2.5(a)

$$e = e_\ell \leq 0.85 \left(\frac{3x_0 + 6}{\log K} \right)^{1/2} \leq \frac{0.85 (300(3x_0 + 6))^{1/2}}{\sqrt{\log n}} =: \frac{c_e}{\sqrt{\log n}}.$$

With $2m + n \leq 3m$, the terms of (7.2) and the branching satisfy

$$\begin{aligned} eU &\leq c_e (4.65m + 27n^{0.8})/\sqrt{\log n}, \\ (e+2)gP/K &\leq 3(7m + 81n^{0.8})/n^{1/300} = O(mn^{-1/300}), \\ \log \tau &\leq 4.5m/g + O(\log n) = O(mn^{-8/300}), \\ eK^2 + C_0K^2 \log(K(L+B')) &= O(n^{1/15} \log n) \quad (L, B' \leq \text{poly}(n), \text{ as } m \leq n^2), \\ g \log K &= O(n^{8/30} \log n), \quad C_1 \log(g(P+B)) = O(\log n), \\ \log(\#\text{branches}) &\leq 4(m/n)g \log n = O(mn^{8/30-1} \log n). \end{aligned}$$

Every term except the first is $O(m/\sqrt{\log n})$ uniformly, because $m \geq n$ and $n^{0.8}, n^{1/15} \log n$ and $n^{8/30} \log n$ are all $o(n/\sqrt{\log n})$. Hence the total time is $2^{C^*m/\sqrt{\log n}}$ for an absolute C^* . $\square$

Remark 7.3. We record four comments on these bounds.

- (a) The requirement $g = 3K^8 \leq n^{8/30}$ keeps the g^3 copy bits and the $n^{O(Dg)}$ branching at $o(n/\sqrt{\log n})$; it forces only $K \geq n^{1/300}$, which still gives $e_\ell = O(1/\sqrt{\log n})$.
- (b) With the pieces of [A, Lemma 2] only ($\rho = 11/12$), the same argument gives $2^{O(m/(\log n)^{0.125})}$.
- (c) The leading term is $e_\ell U \approx e_\ell \cdot 4.64 |E|$; every other term is of lower order. Hence any improvement of e_ℓ in Theorem 2.5, for example from better pieces, transfers verbatim.
- (d) Like the bounds of [A], these are purely asymptotic. Theorem 1.2 beats 2^n only when $e_\ell \cdot 4.64m < n$, which needs $\log n$ in the thousands even for $m = 2n$.

8. COUNTING WITH LOGARITHMIC COST IN THE DEGREE

This section proves Theorem 1.3. The proof is formalised in Lean (`theorem3`, `theorem3\_maxdeg`; files `Unrank`, `PairCode`, `VertexBalanced`, `RankNetwork`, `RankProcs`, `DegTableProc`, `RankWordProcs`, `Theorem3`, `DPackage`). It reuses Theorem 5.7 verbatim and changes only the

network of [Section 6](#): the pair $\{a, b\}$ no longer shares one base-5 digit for *every* crossing edge of $E(a, b)$, but only the set of crossing edges of $E(a, b)$ that carry a label other than $\mathbf{N}$, rank-encoded among the subsets of the right size, together with the labels of those edges. A 2-factor uses exactly two of the $\deg(v)$ edges at each vertex v , so this costs about $2 \log \deg(v)$ bits per vertex rather than $\log_2 5$ bits per edge. The rank-encoding device is that of [A, Appendix D, Step 3]; we prove what we need from scratch. Fix G with $n \geq 3$ and minimum degree at least 2 (so $m \geq n$ and $\bar{d} \geq 2$), a root r , a partition ϕ and a crossing order, with the conventions of [Section 5](#). Put $n_a := |V_a|$, $\bar{n} := \max_a n_a$, $\lambda_\Delta := \log(2e\Delta)$ and $\lambda_{\bar{d}} := \log(2e\bar{d})$; both are at least $\log(4e) > 3.4$.

8.1. Profiles and the rank-encoded network. A *profile* is an integer vector $\xi = (\xi_{ab})_{a < b}$ with $0 \leq \xi_{ab} \leq m_{ab}$; put $\xi_{ba} := \xi_{ab}$, $\xi_a := \sum_{b \neq a} \xi_{ab}$ and $|\xi| := \sum_{a < b} \xi_{ab}$. The profile of a labelling $\eta \in \mathcal{L}^{m \times}$ is $\xi(\eta)_{ab} := \#\{k \in E(a, b) : \eta_k \neq \mathbf{N}\}$. A profile is *admissible* if $\xi_a \leq 2n_a$ for every a . For $0 \leq s \leq t$, the colex rank $\{u_1 < \dots < u_s\} \mapsto \sum_{i=1}^s \binom{u_i}{i}$ is a bijection from the s -subsets of $\{0, \dots, t-1\}$ onto $\{0, \dots, \binom{t}{s} - 1\}$, inverted greedily with at most t binomial comparisons; write $\text{cw}(t, s) := \lceil \log \binom{t}{s} \rceil$.

Definition 8.1 (The network $\mathcal{N}_\xi$ on $[g]$). Fix a profile ξ . A setting of node a is a pair $z_a = (\eta^{(a)}, y^{(a)})$ with $\eta^{(a)} \in \mathcal{L}^{\partial(a)}$ such that $\#\{k \in E(a, b) : \eta_k^{(a)} \neq \mathbf{N}\} = \xi_{ab}$ for every $b \neq a$, and $y^{(a)} \in \{0, 1\}^{Q_a}$. It is encoded as the concatenation of

- for each $b \neq a$, in increasing order, the block $\text{block}_b(z_a)$ of width $c'_{ab} := \text{cw}(m_{ab}, \xi_{ab}) + 2\xi_{ab}$: if $k_1 < \dots < k_{m_{ab}}$ enumerate the bundle of $\{a, b\}$ in crossing order and $T_{ab} := \{i : \eta_{k_i}^{(a)} \neq \mathbf{N}\}$, the block is the colex rank of T_{ab} in $\text{cw}(m_{ab}, \xi_{ab})$ bits, followed by the 2-bit codes of the labels $\eta_{k_i}^{(a)} \in \{0, 1, 2, 3\}$, $i \in T_{ab}$ increasing;
- the bits $y_{cd}^{(a)}$, $(c, d) \in Q_a$, in lexicographic order.

S_a is the set of strings of length $P_a := \sum_{b \neq a} c'_{ab} + |Q_a|$ whose rank field in every block is $< \binom{m_{ab}}{\xi_{ab}}$. The functions $W_a = \sigma_a \chi_a I_a$ and $\Psi_{a,b}$ are those of [Theorem 6.1](#), and $c_{a,b} := c'_{ab} + (g - \max(a, b))$, with h_{ab}, h_{ba} and $f_{a,b}, f_{b,a}$ as there, built from the new blocks.

Lemma 8.2. *The encoding is a bijection from the settings of a onto S_a , and $\text{block}_b(z_a) = \text{block}_a(z_b)$ if and only if $\eta^{(a)}$ and $\eta^{(b)}$ agree on the bundle of $\{a, b\}$. Hence $\Psi_{a,b} = [h_{ab}(z_a) = h_{ba}(z_b)]$, and $(f_{a,b}, f_{b,a})$ factors $\Psi_{a,b}$ through $c_{a,b}$ bits.*

Proof. Both ends list the bundle in the same order and use the same ξ_{ab} . The block determines T_{ab} and the labels on it, hence the restriction of $\eta^{(a)}$ to the bundle (the other labels are $\mathbf{N}$); conversely a restriction with exactly ξ_{ab} labels other than $\mathbf{N}$ gives exactly one block, and every rank below $\binom{m_{ab}}{\xi_{ab}}$ is the rank of exactly one subset. The rest is the proof of [Theorem 6.2](#). $\square$

Theorem 8.3. $\sum_{\xi \text{ admissible}} \text{value}(\mathcal{N}_\xi) = n \cdot \#\text{HC}(G)$, and $\text{value}(\mathcal{N}_\xi) = 0$ for every profile that is not admissible.

Proof. As in the proof of [Theorem 6.3](#), a nonzero term of $\text{value}(\mathcal{N}_\xi)$ comes from a global $\eta \in \mathcal{L}^{m \times}$ with correct copies, and by [Theorem 8.1](#) that η has $\xi(\eta) = \xi$; conversely each such η gives exactly one z . So $\text{value}(\mathcal{N}_\xi) = \sum_{\eta: \xi(\eta) = \xi} \prod_a \sigma_a \chi_a(\eta, \pi(\eta)) I_a$. The profiles partition $\mathcal{L}^{m \times}$, so the sum over all profiles is $n \cdot \#\text{HC}(G)$ by [Theorem 5.7](#). If $\xi_a > 2n_a$, then every η with $\xi(\eta) = \xi$ has $\sum_{v \in V_a} \deg_\eta(v) = \xi_a > 2n_a$, since each crossing edge at a has exactly one end in V_a ; so some $v \in V_a$ has $\deg_\eta(v) \geq 3$, every indicator in $I_a(\eta)$ vanishes, and the value is 0. $\square$

Proposition 8.4 (Parameters). *Let ξ be admissible. Then $\mathcal{N}_\xi$ is a network in the sense of [Theorem 2.2](#) with*

$$P := \lceil 2\bar{n}\lambda_\Delta \rceil + g + g^2, \quad B := \lceil 2\bar{n}\lambda_\Delta \rceil + 1, \quad \tau := 2^{2\bar{n}\lambda_\Delta} n^{C_2} \quad (C_2 \text{ absolute}),$$

τ counted after a polynomial-time preparation, and

$$U = 2 \sum_{a < b} c'_{ab} + 2 \binom{g}{3} \leq 2n \log(2e\bar{d}) + g^3.$$

P and B do not depend on ξ .

Proof. We use $\binom{N}{M} \leq (eN/M)^M$, Vandermonde's inequality $\prod_i \binom{N_i}{M_i} \leq \binom{\sum N_i}{\sum M_i}$, $\sum_i \lceil x_i \rceil \leq \sum_i x_i + \#\{i : x_i > 0\}$, and the fact that $F_N(M) := M \log(eN/M) + 2M$ is nondecreasing on $[0, N]$ (its derivative is $\log(N/M) + 2$).

Per node. Let $N_a := |\partial(a)| \leq \Delta n_a$. Admissibility gives $\xi_a \leq \min(2n_a, N_a)$, so $\sum_{b \neq a} c'_{ab} \leq \log \binom{N_a}{\xi_a} + 2\xi_a + g - 1 \leq F_{N_a}(\xi_a) + g - 1$. If $2n_a \leq N_a$, then $F_{N_a}(\xi_a) \leq F_{N_a}(2n_a) \leq 2n_a \log(e\Delta/2) + 4n_a = 2n_a \lambda_\Delta$; otherwise $F_{N_a}(\xi_a) \leq (\log e + 2)N_a < 2n_a \log(4e) \leq 2n_a \lambda_\Delta$. With the copy bits counted as in [Theorem 6.4](#), $P_a \leq P$ and $\sum_{b \neq a} c_{a,b} \leq 2n_a \lambda_\Delta + g - 1 + g^2 \leq P$.

Total. $U = 2 \sum_{a < b} c'_{ab} + 2 \binom{g}{3}$ as in [Theorem 6.4](#), and $\sum_{a < b} c'_{ab} \leq \log \binom{m_\times}{|\xi|} + 2|\xi| + \binom{g}{2}$. Admissibility gives $2|\xi| = \sum_a \xi_a \leq 2n$, and $|\xi| \leq m_\times \leq m$. If $n \leq m_\times$, monotonicity gives $\log \binom{m_\times}{|\xi|} + 2|\xi| \leq F_{m_\times}(n) \leq n \log(em/n) + 2n$; if $m_\times < n$, it is at most $(\log e + 2)m_\times < n \log(4e) \leq n \log(4em/n)$, as $m \geq n$. Since $n \log(4em/n) = n \log(2e\bar{d})$, we get $U \leq 2n \log(2e\bar{d}) + g^2 + 2 \binom{g}{3} \leq 2n \log(2e\bar{d}) + g^3$.

Condition (ii). Ψ and f are 0/1-valued and $|\sigma_a \chi_a| = 1$. In $I_a(\eta)$ only those x contribute with $\deg_x(v) = 2 - \deg_\eta(v)$ for all $v \in V_a$. The map $x \mapsto (x \cap \delta(v))_{v \in V_a}$ is injective, so there are at most $\prod_{v \in V_a} (1 + \deg v + \binom{\deg v}{2}) \leq \Delta^{2n_a}$ of them (as $1 + d + \binom{d}{2} \leq d^2$ for $d \geq 2$). For each, $|x| \leq n_a$, and the top coefficient is at most the ℓ^1 -norm of $\mathcal{F}_a(x) R_a(\eta)$, which is at most $5^{|x|} 2^{\xi_a} \leq 20^{n_a}$: the ℓ^1 -norm is submultiplicative (structure constants 0, ± 1), $1 + q_e$ has norm at most 5, and a piece has norm at most 2. Hence $\log |W_a| \leq 2n_a(\log \Delta + 2.17) < B$.

Condition (iii). The preparation stores the graph, ϕ , the order, the bundles, the binomials $\binom{m_{ab}}{\xi_{ab}}$ and the layout. Membership, encoding, decoding (one unranking per block), h_{ab} , Ψ and f take $\text{poly}(n)$ operations, σ_a and χ_a as in [Theorem 6.4](#). For I_a the procedure enumerates the tuples $(x_v)_{v \in V_a}$, x_v a subset of size $2 - \deg_\eta(v)$ of the internal edges at v , keeps the consistent ones (each chosen edge chosen at both ends), and for each computes the top coefficient by choosing one signed monomial from each factor, multiplying (sign by inversion count) and accumulating when the product is $\pm \omega_a$. This takes at most $\Delta^{2n_a} \cdot 20^{n_a} \cdot \text{poly}(n) \leq 2^{2n_a \lambda_\Delta} n^{C_2}$ operations, with a state of $\text{poly}(n)$ bits. $\square$

There are at most $\prod_{a < b} (\min(m_{ab}, 2\bar{n}) + 1) \leq (2\bar{n} + 1)^{\binom{g}{2}}$ admissible profiles, enumerated and tested in polynomial time each.

8.2. The algorithm and the proof of [Theorem 1.3](#).

Step 0. If $n \leq 2$, or some vertex has degree less than 2, output 0.

Step 1. Choose the level ℓ , $K := K_\ell$ and $g := 3K^8$ as in [Section 7.6](#).

Step 2. List V as $v_1, \dots, v_n$ and put $\phi(v_t) := \lceil tg/n \rceil$, so that $n_a \leq \lceil n/g \rceil$ (each blob is the set of integers in a half-open interval of length n/g). Take any root and any crossing order, and do the preparation of [Theorem 8.4](#).

Step 3. For each admissible profile ξ , run the procedure of [Theorem 2.4](#) on $\mathcal{N}_\xi$ with the algorithm of [Theorem 2.5\(b\)](#) at level ℓ as the clique algorithm.

Step 4. Output the sum of the results divided by n .

There is no degree reduction and no degree-balanced partition. Correctness follows from [Theorems 2.4, 2.5](#) and [8.3](#); the imported hypotheses are checked as in [Section 7](#), with [Theorem 8.4](#) in place of [Theorem 6.4](#), and [Theorem 2.4](#) is applied to each $\mathcal{N}_\xi$ separately (only U depends on ξ).

Proof of Theorem 1.3. For $n < n_0$ use any exact method. For $n \geq n_0$, as in Section 7.6, $n^{1/300} < K \leq n^{1/30}$, $g \in [3n^{8/300}, 3n^{8/30}]$ and $e := e_\ell \leq c_e/\sqrt{\log n}$. By Step 2, $\bar{n} \leq n/g + 1$, and $\lambda_\Delta \leq \log(2en) \leq 2\log n$ because $\Delta < n$. By Theorem 8.4,

$$U \leq 2n\lambda_{\bar{d}} + g^3, \quad gP \leq 2(n+g)\lambda_\Delta + 3g^3, \quad \log \tau \leq 2\bar{n}\lambda_\Delta + C_2 \log n,$$

and $L \leq U + gP + 1$, $B' \leq 2gP + 1 + g^2B$ are $\text{poly}(n)$. The time for one profile is 2^X with X bounded by (7.2), whose terms satisfy

$$eU \leq c_e(2n\lambda_{\bar{d}} + 27n^{0.8})/\sqrt{\log n},$$

$$(e+2)gP/K \leq 3(4(n+g)\log n + 3g^3)/n^{1/300} = O(n^{1-1/300} \log n),$$

$$\log \tau \leq 4(n/g + 1)\log n + C_2 \log n = O(n^{1-8/300} \log n),$$

$$eK^2 + C_0K^2 \log(K(L+B')) = O(n^{1/15} \log n), \quad g \log K = O(n^{8/30} \log n),$$

and $C_1 \log(g(P+B)) = O(\log n)$. The number of admissible profiles is at most $(2n/g + 3)^{g^2/2} = 2^{O(n^{16/30} \log n)}$. Every term except the first is $o(n/\sqrt{\log n})$ for every $\Delta \leq n$, and $\lambda_{\bar{d}} > 3.4$, so the total time is $2^{C^\sharp n \lambda_{\bar{d}}/\sqrt{\log n}}$ for an absolute $C^\sharp$. Finally $\bar{d} \leq \Delta$. $\square$

Remark 8.5.

- (a) The degree enters in two ways. The total U depends only on $\bar{d}$, for every partition. The per-node parameters depend on Δ , but only logarithmically, so a blob of $\lceil n/g \rceil$ vertices of degree $n-1$ still has only $O((n/g) \log n)$ bits, and the term $2gP/K$ of (2.1), which forced the degree reduction of Section 7 (Theorem 6.5), is $o(n/\sqrt{\log n})$ for every Δ .
- (b) The leading term is $e_\ell U \approx 2e_\ell n \log(2e\bar{d})$, against $e_\ell \cdot 2 \log_2 5 \cdot m \approx 2.32 e_\ell \bar{d} n$ in Theorem 1.2; the new term is the smaller one for $\bar{d} \geq 4$. Better pieces in [A] transfer verbatim.
- (c) The self-reductions of Theorem 1.4 delete edges, which does not increase $\bar{d}$; for paths, the added vertex raises $\bar{d}$ by at most 2. So decision, search and paths follow within the bound of Theorem 1.3 times $n^2 + 1$; this extension is not machine-checked and is not claimed.
- (d) With a *fixed* level, as in Section 7.5, the same computation gives $O(2^{\delta n})$ on graphs of maximum degree at most Δ_0 as soon as $e \leq \delta/(8 \log(2e\Delta_0))$ and $K \geq 24 \log(2e\Delta_0)/\delta$, which is weaker than the requirement $e \leq \delta/(40D)$ of Theorem 1.1 when D is large. The average-degree form does not follow this way, because $gP/K \approx 2n\lambda_\Delta/K$ is not $O(\delta n)$ when Δ is unbounded. We do not use this.

9. DECISION, SEARCH AND PATHS

Proof of Theorem 1.4. Decision. G is Hamiltonian iff $\#\text{HC}(G) > 0$; one call of Theorem 1.1 or Theorem 1.2 suffices.

Search, by self-reduction. Let $H := G$. For each edge e of G in turn, if $\#\text{HC}(H - e) > 0$, set $H := H - e$. The invariant is $\#\text{HC}(H) > 0$. Let C be a Hamiltonian cycle of the final H . Every intermediate H contains the final one, hence C . Suppose some edge f of the final H were not in C . When f was examined, $H - f$ contained C , so f would have been deleted. Hence the final H equals C . The machine runs over all n^2 vertex pairs (deleting a non-edge changes nothing), so there are $n^2 + 1$ calls, each on a graph with n vertices and at most m edges; the polynomial factor is absorbed into the constant.

Hamiltonian paths. Let G^+ be G plus a new vertex w adjacent to all of V . For $n \geq 2$, the Hamiltonian paths of G correspond bijectively to the Hamiltonian cycles of G^+ , so $\#\text{HP}(G) = \#\text{HC}(G^+)$. The graph G^+ has $n+1$ vertices and $m+n \leq (D+1)(n+1)$ edges; the apex w has degree n , which Theorem 1.2 handles directly; for the fixed-density form the Lean proof applies Theorem 1.1 to G^+ with $D+1$ in place of D . Search works as above. (For s - t paths one would join a new vertex to s and t only; that variant is not machine-checked and not claimed.) $\square$

10. RESULTS THAT REST ONLY ON THE TOP-LEVEL THEOREMS

This section concerns [Theorems 1.6](#) and [1.7](#). Unlike [Theorem 1.1](#), they use the labs' #SAT theorems only as black boxes, through CNF encodings.

10.1. The Cut&Count identity. Let $r \in V$. The CNF $\text{cut}(G, r)$ has variables x_e ($e \in E$) and c_v ($v \in V$) and imposes: at each vertex v , exactly two of the x_e with $e \ni v$ are true; for each edge uw , $x_{uw} \rightarrow (c_u \leftrightarrow c_w)$; and $c_r = \text{false}$. Each local constraint becomes clauses by forbidding its violating local patterns.

Lemma 10.1 (Cut&Count [20]). *The number of models of $\text{cut}(G, r)$ is $\sum_X 2^{\text{cc}(X)-1}$, the sum over the 2-factors X of G , where $\text{cc}(X)$ is the number of components of (V, X) . In particular it is congruent to $\#\text{HC}(G)$ modulo 2.*

Proof. A model is a pair (X, c) where X is a 2-factor and $c: V \rightarrow \{0, 1\}$ is constant on the components of (V, X) with $c(r) = 0$. There are $2^{\text{cc}(X)-1}$ such c , which is odd iff X is connected, i.e. a Hamiltonian cycle. $\square$

For unbounded degree, the vertex constraint is replaced by a sequential counter (two bits per prefix of the incident edges), giving a 5-CNF on $5m + 3n$ variables with the same model count. The Lean development proves these identities (`count_cutCNF`, `count_cutCNF_mod_two`, `count_bigCNF`).

Proof of Theorem 1.6. Feed the 5-CNF to the respective #SAT algorithm. With [A], the formula has $N = 5m + 3n$ variables and at most $85m + 11n + 2$ clauses (after padding so that every variable occurs), and the Main Theorem gives a count within $2^{C \cdot 5m' / \sqrt{\log m'}}$ steps, where $N \leq m' \leq 85m + 11n + 2$ is the number of clauses. This is the Lean theorem `hamParity_allGraphs_anthropic`, whose hypothesis `MainResults.MainTheorem` is discharged by [A]'s Lean proof ([Section 12](#)). With [35], the Lean theorem `hamParity_allGraphs` gives one machine with time $C(L + 1)^{d_{Ap}(5m+3n)}$ and space polynomial in the encoding length L ; its hypothesis `exact_count_and_sat` is discharged by OpenAI's Lean proof. The directed version uses one selected out-arc and in-arc per vertex (`dirHamParity_subexponential`). Building the CNF from G takes polynomial time; that step is not formalised as a machine. $\square$

10.2. Why colourings do not give exact counts. Cut&Count extracts $[X \text{ connected}]$ from a sum over consistent colourings with local weights. The following barrier shows that such filters cannot be exact with few colours; this is what motivates the Grassmann construction.

Proposition 10.2 (machine-checked: `colouring_barrier_graph`). *Work over a field of characteristic 0. Suppose that weights $z_0, \dots, z_{q-1}$ on q colours, with the root coloured 0 and every component $C \not\ni r$ of a 2-factor contributing $h(C) = \sum_j z_j^{|C|}$, give a sum equal to $\#\text{HC}(G)$ up to a nonzero factor, for every graph G on n vertices. Then $q \geq n - 4$.*

Proof. We have $z_0 \neq 0$, since otherwise Hamiltonian cycles contribute 0; normalise $z_0 = 1$. The root's cycle has at least 3 vertices, so a non-root cycle has length $k \in \{3, \dots, n - 3\}$, and every such length occurs: in the disjoint union of $C_{n-k} \ni r$ and C_k , the only 2-factor is disconnected. So the filter needs $\sum_{j=0}^{q-1} z_j^k = 0$ for $k = 3, \dots, n - 3$. Drop the z_j equal to 0, and group the rest into distinct nonzero values λ_i with positive integer multiplicities. This gives an exponential polynomial with at most q distinct nonzero bases and positive coefficients that vanishes at $n - 5$ consecutive integers. If $n - 5 \geq q$, the Vandermonde matrix on those integers forces every coefficient to 0, a contradiction because $\lambda = 1$ has multiplicity at least 1. $\square$

Conversely $q = n - 2$ colours with $z_j = \zeta^j$, ζ a primitive $(n - 2)$ -th root of unity, suffice. So exact counting by colourings needs about $\log n$ bits per vertex, and the exponent becomes worse than brute force; the positive multiplicities vanish only in small characteristic, which is how the modulo-2 filter escapes. The proposition does not cover vertex-dependent weights.

10.3. Randomized decision.

Proof of Theorem 1.7(a). Add to $\text{cut}(G, r)$ an arbitrary CNF $\mathcal{H}$ on the edge variables. Then the model count is congruent modulo 2 to $\#\{X \in \text{HC}(G) : \mathcal{H}(\mathbf{1}_X)\}$ (`count_cutCNFWith_mod_two`); it is even if G has no Hamiltonian cycle and odd if exactly one Hamiltonian cycle satisfies $\mathcal{H}$. Take $\mathcal{H}$ to be a random sparse hash in the sense of Calabro, Impagliazzo, Kabanets and Paturi [13]. Their isolation lemma [13, Lemma 3] is stated for solution sets of k -CNFs, but its proof uses only that the set is a set; the Lean development re-proves it for arbitrary finite $S \subseteq \{0, 1\}^N$ as a counting statement (`isolation_count`), together with the entropy estimate needed to make the isolating fraction at least $\exp(-\delta N)$ for every $\delta > 0$ (`isolation_fraction`). With $S = \text{HC}(G)$ and $N = |E| \leq \Delta n/2$ this gives soundness (`decision_sound`), the success fraction (`hamDecision_fraction`) and one machine for all hashes (`hamDecision_subexponential`). Repeating $t \exp(\delta|E|)$ independent trials for each of polynomially many parameter choices gives error at most e^{-t} ; this outer loop is the only part not modelled in Lean. $\square$

Remark 10.3 (The superseded part (b); not claimed). Let $P_G(w) = \sum_{H \in \text{HC}(G)} \prod_{e \in H} w_e$. It is multilinear with 0/1 coefficients, so as a function on $\mathbb{F}_p^E$ it is nonzero iff G is Hamiltonian. By the Alon–Füredi theorem [1], a polynomial of degree d that does not vanish on $A_1 \times \cdots \times A_N$ is nonzero on at least $\min\{\prod y_i : 1 \leq y_i \leq |A_i|, \sum y_i \geq \sum |A_i| - d\}$ points; with $A_i = \mathbb{F}_p$ and $d = n$, a uniform w gives $P_G(w) \neq 0$ with probability at least $p^{-n/(p-1)-1}$. The value $P_G(w) \bmod p$ is the model count, modulo p , of a CNF with p colours per vertex ($\lceil \log p \rceil$ bits with a comparator), the root coloured 0, bitwise constraints $x_e \rightarrow (c_u = c_v)$, the vertex constraint “exactly two”, and weight bits z_e with $x_e = 0 \Rightarrow z_e = 0$ and $x_e = 1 \Rightarrow z_e < w_e$; its count is $\sum_X p^{\text{cc}(X)-1} \prod_{e \in X} w_e \equiv P_G(w) \pmod{p}$. It has $O(|E| \log p)$ clauses of width $O(\log p)$. [A]’s Main Theorem, which is uniform in k , gives time $2^{O(|E| \log^2 p / \sqrt{\log |E|})}$ per test. Taking a prime $p \approx \sqrt{\log n}$ and $p^{n/(p-1)+1} \cdot t$ independent tests gives one-sided error e^{-t} in total time $2^{O_D(n(\log \log n)^2 / \sqrt{\log n})}$. This argument is not machine-checked (the Alon–Füredi theorem is, as far as we know, not in Mathlib) and is recorded only for the history of the project.

11. EXTENSIONS (SKETCHED)

This section is not at the level of rigour of the rest of the paper. The identities below were checked exactly by brute force on small instances, and the blob networks were evaluated on random partitions; the algorithmic consequences are sketched, and all of them are conditional on (H_A) applied to general networks, exactly as for Theorem 1.1.

11.1. The template. All problems below fit one template:

$$(11.1) \quad Z = \sum_{s, \xi} \prod_v C_v(s_v, \xi|_{\delta(v)}) \cdot [\Omega] \left(\prod_v M_v(s_v) \prod_{e=uv} F_e(s_u, s_v, \xi_e) \right),$$

where s_v is a vertex state from a constant-size alphabet, ξ_e an edge state ($\delta(v)$ is the set of edges at v), C_v an arbitrary local integer table, M_v an even element in the generators of v (for example the mass term $1 + w\psi_v\psi_v$, or $\psi_v\psi_v$, which deletes v), and F_e an even element in the generators of the ends of e , possibly involving pairwise scalars such as z^{w_e} . A root has no generators. On a blob partition, each crossing edge is split by an exact rank factorisation of its crossing matrix for each parity, a label being a term of this factorisation; the edge costs $\log_2(\#\text{labels})$ bits. Because each F_e is even, both pieces of a term have the same parity, and the sign rule of Section 5 applies unchanged, at a cost of $O(g^3)$ parity bits. Global counting constraints are handled by generating variables, folded into one integer by Kronecker substitution; this changes only B , which stays polynomial. Several connectivity predicates can be imposed at once by giving each its own independent Grassmann field; the top coefficient then factors over the fields (sharing one field would compute $\det(L_{X_1} + L_{X_2})$ instead).

TABLE 3. Problems fitting the template (11.1). All identities were checked exactly on small instances (brute force = determinant identity = blob network on random partitions with 2–4 blobs). “Labels” is the number of labels per crossing edge.

Problem	Identity (local constraints $\times$ one determinant)	Labels
#HC (this paper)	$n\#\text{HC} = \sum_{2\text{-factors } X} \det L_X^{(r)}$	5
directed #HC	$\#\text{DHC} = \sum_{\text{cycle covers } X} \det(D_{\text{out}} - A)_X^{(r)}$	3
TSP/ATSP, weights $\leq \text{poly}(n)$	$n \sum_H z^{w(H)} = \sum_X z^{w(X)} \kappa(X)$	5 / 3
Hamiltonian s – t paths	$\sum_{X: \deg s = \deg t = 1, \text{ else } 2} \det L_X^{(s)}$	5
spanning trees with local degree constraints	$[z^{n-1}] \sum_X z^{ X } \prod_v c_v(\deg_X v) \kappa(X)$	5
Steiner tree (count, min. weight)	$\sum_{S \supseteq T} \det L_w(G[S])^{(t_0)}$	5
induced forests, min. FVS (count)	$[\text{forest}] = 2^{ E(S) - S } [z^{ S - E(S) }] \det(L_S + z \text{diag}(2 - \deg_S))$	5
all forests, $T_G(2, 1)$ and $T_G(x, 1)$	the same identity with $S = V$ and free edges	4
k -paths, k -cycles (all k)	coefficient extraction from $\det(L_X + wI)$ with markers	5
connected VC / connected DS	decision and minimum only; counts are κ -weighted	4 / 5

Machine-checked rows. For directed Hamiltonian cycles and for ATSP the statement below is proved in Lean, without the degree hypothesis and also in rate form for #DHC, by splitting every vertex into an in-, a middle and an out-copy and applying [Theorems 1.1, 1.2 and 1.5](#) to the resulting undirected graph (`theoremDHC`, `theoremDHC\_rate`, `theoremATSP`, `theoremATSP\_decode`; standard axioms only). The directed identity of the table is formalised as well (`card\_dirHamCycles\_eq\_sum`, `dblob\_factorisation\_w`). For all other rows the proposition is a sketch and is not claimed.

Proposition 11.1 (Sketch; not claimed except for the machine-checked rows). *Assume (H_A) , and let $\mathcal{P}$ be one of the problems of [Table 3](#), on graphs with $|E| \leq Dn$ and either bounded maximum degree or the hub-handling of [Theorem 11.2](#). Then for every $\delta > 0$ the quantity stated in the table (exact count, optimum, or κ -weighted count) can be computed deterministically in time $2^{\delta n}$.*

Proof sketch. The identity in the table is an instance of (11.1); the blob network is built as in [Section 6](#), with $U \approx 2 \log_2(\#\text{labels}) \cdot |E| + O(g^3)$ and $P \approx U/g + g^2$, and the bookkeeping of [Section 7](#) applies verbatim. The identities have been checked numerically; the uniform statements of the bit bounds, and the hub-handling, have not been written out in full. $\square$

Remark 11.2 (Hubs). A vertex of degree $\Theta(n)$ breaks the per-node bound P . For degree-2 problems (cycles, paths, cycle covers) one can branch as in [Theorem 7.1](#), or compress the hub’s realisable labellings into $O(\log d)$ bits per pair. For tree and forest problems one can split the hub into a path of copies joined by *forced* edges (factor q_e instead of $1 + q_e$), which preserves the counts of subtrees and induced forests. A root vertex costs nothing, because it has no generators. These devices were checked on small instances only.

11.2. The Tutte polynomial on the line $y = 1$. On the line $y = 1$, the Tutte polynomial is $T_G(x, 1) = \sum_F (x-1)^{r(E)-|F|}$, the sum over the forests $F \subseteq E$, where $r(E) = n - \text{cc}(G)$; so counting forests by number of edges gives the whole polynomial $T_G(x, 1)$. The key identity is exact and short.

Lemma 11.3 (Forest indicator; written proof, not machine-checked, not claimed). *Let H be a graph on a vertex set S with edge set X , and let $\deg_H$ denote degrees in H . Then*

$$[z^{|S|-|X|}] \det(L_H + z \operatorname{diag}(2 - \deg_H)) = \begin{cases} 2^{|S|-|X|} & \text{if } H \text{ is a forest,} \\ 0 & \text{otherwise.} \end{cases}$$

Proof. By the matrix-forest theorem, $\det(L_H + \operatorname{diag}(d)) = \sum_F \prod_T (\sum_{v \in T} d_v)$, the sum over the spanning forests F of H and the product over the trees T of F . With $d_v = z(2 - \deg_H v)$, a forest with j trees contributes a multiple of z^j . Every spanning forest has at least $\operatorname{cc}(H) \geq |S| - |X|$ trees, with equality throughout iff H is itself a forest and $F = H$. If H is not a forest, the coefficient of $z^{|S|-|X|}$ is therefore 0. If H is a forest, the coefficient comes from $F = H$ alone and equals $\prod_T \sum_{v \in T} (2 - \deg_T v) = \prod_T (2|T| - 2(|T| - 1)) = 2^{\operatorname{cc}(H)} = 2^{|S|-|X|}$. $\square$

In Grassmann form the mass $-z \deg_H(v) \bar{\psi}_v \psi_v$ is spread over the edges, so all factors are local or pairwise and (11.1) applies with four labels per crossing edge. Brand, Dell and Roth [12] showed that, under #ETH, for every real $x \neq 1$ there is $\varepsilon > 0$ such that $T_G(x, 1)$ has no $\exp(\varepsilon n)$ -time algorithm on simple graphs with $O(n)$ edges; this was the last open line of the #ETH dichotomy for the Tutte plane. Theorem 11.3 together with Theorem 11.1 would give, conditional on (H_A) and with the caveats of this section, a deterministic $2^{\delta n}$ algorithm for the whole polynomial $T_G(x, 1)$ on such graphs. (The published implication of course remains true; it is vacuous once #ETH fails.) The best previous algorithm computes the Tutte polynomial in vertex-exponential time [7].

Remark 11.4 (Further problems). A more general version of the template allows a constant number of global predicates (connected, tree, path, single cycle, forest, minimum number of components, and directed analogues such as branchings and strong connectivity), each certified by a determinant that is positive exactly when the predicate holds. It covers, among others, connected feedback vertex set, connected odd cycle transversal, minimum cycle cover, k -leaf out-branchings, maximum full-degree spanning trees and graph-metric TSP. Exact counting is obtained when each certificate is constant on solutions; for connected vertex covers, connected dominating sets and connected subgraphs, only κ -weighted counts are obtained. All of this is exploratory and numerically checked on instances with at most seven vertices.

12. VERIFICATION AND FORMALISATION STATUS

We describe what has been checked, by what means, and what remains open. The aim is to make clear how much trust each statement deserves. All Lean sources, the axiom-audit files (`lean/EthFollowup/Axioms.lean`, `lean/EthFollowup/AxiomsExtended.lean` and `lean-bridge-anthropic/bridge/EthFollowup/ExactHC/AuditAll.lean`), build instructions and the numerical check scripts are in the public repository [38], <https://github.com/xennygrimmato/eth-hamiltonicity>, Lean sources as of commit `ca17a37`.

12.1. Machine-checked in Lean.

- *Parity and randomized decision* (Theorems 1.6 and 1.7(a)). The Cut&Count identities, the counter encoding for unbounded degree, the restriction by edge CNFs, the Calabro–Impagliazzo–Kabanets–Paturi isolation lemma for arbitrary finite sets, its entropy estimate, and the machine theorems are proved in Lean 4 [21] with Mathlib [37], taking the labs’ top-level theorems as hypotheses.
- *Discharging the hypotheses.* OpenAI’s development was built from source at a pinned commit (Lean v4.34.1; 11,492 build jobs, no sorry); #printaxioms reports only `propext`, `Classical.choice` and `Quot.sound` for `exact_count_and_sat`. The instantiated theorems (`hamParity_allGraphs_unconditional`, `hamDecision_unconditional`, `dirHamParity_unconditional`, `dirHamDecision_unconditional`, and others) are closed statements with

the same three axioms, and a kernel replay with `leanchecker` of the full import closure succeeds. Anthropic’s development was likewise built from source (Lean v4.33.1; 9,768 jobs); `MainResults.mainTheorem` depends only on the three standard axioms, and so does `hamParity_allGraphs_anthropic` instantiated with it.

- *Exact-count formalisation.* The mathematical content of [Theorem 5.7](#) is machine-checked. The Lean theorem `blob_factorisation` states that, for every graph G , root r , partition ϕ into blobs and crossing order, $\sum_{\ell} \prod_a \sigma_a(\ell) \chi_a(\ell, P(\ell)) I_a(\ell) = n \cdot \#HC(G)$. It rests only on `propext`, `Classical.choice` and `Quot.sound`. Its proof formalises:

- [Theorem 4.1](#);
- the Grassmann algebra of [Section 3](#) (Koszul rule, block factorisation, linear forms);
- [Theorem 4.2](#) and [Theorem 4.3](#), including $\det(\mathbf{B}_F)^2 = [F \text{ is a spanning tree}]$;
- [Theorem 5.3](#);
- the sign combinatorics ([Theorem 5.4](#), [Theorem 5.5](#), [Theorem 5.6](#));
- the locality of the blob tables.

The network of [Section 6](#) is also formalised in [A]’s own `Net` format, inside [A]’s Lean environment. Its value is proved to equal $n \cdot \#HC(G)$ ([Theorem 6.3](#)), and its parameters satisfy [A]’s conditions ([Theorem 6.4](#), with 3 bits per label instead of $\log_2 5$). [A]’s procedure interface for networks is instantiated generically. The preprocessing lemmas ([Theorems 7.1](#) and [7.2](#)) are formalised. The streaming procedure for the blob tables is also formalised in [A]’s cost model. It is proved to compute $\sigma_a \chi_a I_a$ within $2 \cdot 8^{\mu_a}$ steps using states of linear size. On top of it, the composition with [A]’s Lean theorems gives [Theorem 1.1](#) at the level of Turing machines (`theorem1_TM`): for every $\delta > 0$ and D , one machine outputs $\#HC(G)$ within $C \cdot 2^{\delta n}$ steps whenever $|E| \leq Dn$. The hypothesis bundling the network, its procedures and their bounds is then constructed from the formalised network and procedures (a polynomial-time input reader, the layout, and the size bounds), so [Theorem 1.1](#) is formally verified in Lean end to end (`theorem1`). On input the adjacency matrix of G , a Turing machine with finite alphabets outputs $\#HC(G)$ in binary within $\lfloor C \cdot 2^{\delta n} \rfloor$ steps whenever $|E| \leq Dn$. The theorem rests only on the three standard axioms and on [A]’s Lean development. The sharper single-algorithm bound of [Theorem 1.2](#) is formalised as well (`theorem2`): one machine, with the same axioms, outputs $\#HC(G)$ within $\lfloor C \cdot 2^{C(m+n)/\sqrt{\log_2(n+2)}} \rfloor$ steps for every graph G with m edges. A fresh kernel replay with `leanchecker` of the full import closure has passed for both `theorem1` and `theorem2`.

- *Degree bound, corollaries, directed graphs.* [Theorem 1.3](#) is `theorem3` and `theorem3\_maxdeg`; [Theorem 1.4](#) is `decision\_TM`, `search\_TM`, `hamPaths\_count\_TM` and the related machines; directed Hamiltonian cycles and ATSP are `theoremDHC`, `theoremDHC\_rate`, `theoremATSP`; [Theorem 10.2](#) is `colouring\_barrier\_graph`. All use the standard axioms only.
- *Weighted layer (TSP).* [Theorem 1.5](#) is formally verified end to end (`theoremTSP`, `theoremTSP\_decode`): for every $\delta > 0$ and D there are a Turing machine M and a constant C such that, on the input word `tspCodeGWw` (the graph, W in unary and the n^2 weights in binary), M outputs the binary expansion of $n \sum_H 2^{bw(H)}$ within $\lfloor C \cdot 2^{\delta n} (W+2)^C \rfloor$ steps whenever $|E| \leq Dn$ and all weights are at most W ; the decoding statements (digit j equals n times the number of tours of weight j , the lowest non-zero digit is the optimum, its value over n is the number of optimal tours) are part of the formal theorem. The ingredients are the weighted blob factorisation `blob_factorisation_w`, the decoding lemma `kronecker_decode`, the weighted network `hcNetW_kron` in [A]’s `Net` format, and a weighted table procedure. Standard axioms only; a fresh `leanchecker` replay of its closure has passed.

Lean guarantees that the labs’ proofs establish their formal statements; it does not guarantee that those statements formalise the printed theorems faithfully, which is a matter for expert review. [A] itself notes that its Lean proof was compared with the printed one by AI models, not mechanically.

12.2. Numerical verification. The check scripts are attached to this record as ancillary files in `anc/` (run `python3 anc/run_all.py`; reference outputs, where recorded, in the matching `.log` files) and are also in the project repository [38], directory `research/scripts`: `proof_checks.py` and `proof_params.py` (the identities and parameter bounds of Sections 3 to 6), `review_exact_count*.py` and `review_hc_blobs.py` (the independent referee’s checks, including an ablation of the sign term), `hc_degree_network.py` (Theorem 1.3, with regression and mutation tests), `tsp_check.py` and `tsp_atsp.py` (TSP and ATSP), and `ext_*.py` (the extensions of Section 11). All identities of Sections 3 to 6 were implemented literally and independently of the code used during discovery, in exact integer arithmetic: an exterior algebra over $\mathbb{Z}$ with a *randomised* generator order; the pieces, ε , σ_a , π_{ab} and χ_a ; the network of Theorem 6.1 with fixed-length bit-string settings; and two evaluators that use W , Ψ and f only as black boxes, one by brute force over $\prod_a S_a$ and one by [A]’s grouping into $K = 2$ or 3 groups. Instances include random graphs with arbitrary maps ϕ (empty blobs allowed), planted blob graphs, planted Hamiltonian cycles, and instances with $g \in \{4, 5\}$ whose planted Hamiltonian cycle, or whose planted *disconnected* 2-factor, visits the blobs in interleaving order. Every instance uses a random root, crossing order and generator order. Table 4 summarises one run (60 instances, $n \leq 12$); runs with several other seeds also pass.

TABLE 4. Numerical checks (exact integers). Counts are for one run with 60 instances.

Identity checked	Checks
C1 $n\#\text{HC} = \sum_X \kappa(V, X)$ and $\kappa \in \{0, n\}$ (Theorem 4.1)	60 / 135
C2 $\kappa = \det L_X^{(r)} = [\Omega_V](\prod(1 + q_e))$ (Theorem 4.3)	135
C3 Theorem 4.2 on random integer matrices	200
C4 Koszul rule on random homogeneous sequences (Theorem 3.2)	300
C5 block factorisation (Theorem 3.3)	180
C6 Theorem 5.2, both orientations and root ends	5,216
C7 Theorem 5.3 as an equality of algebra elements	162,875
C8 Theorem 5.5 and Theorem 5.6	162,875 each
C9 Theorem 5.3 with fresh generators (no cancellation); Theorem 5.4	400 / 1,033
C10 Theorem 2.2 for the network: (i)–(iii), U , $ S_a $, $ W_a < 2^B$, $\Psi = \sum_\theta ff'$	$\approx 594,000$
C11 Theorem 5.7; network value by brute force; network value via K -grouping	60 / 60 / 103
C12 ablations: no interleaving sign / no local signs / no sign: wrong on 20 / 8 / 18 of 60	–

A second script checks Theorem 7.1 (on 109 random graphs and several thresholds), Theorem 7.2 in the exact form $g\mu_a \leq 2m + g\Delta$ (on 400 graphs, $g \leq 12$), the counting identities used in Theorem 6.4, and the self-reduction of Theorem 1.4. These checks test the identities and the network algebra, sign included; they cannot test the asymptotic bookkeeping, which is checked by hand.

A third script checks the rank-encoded network of Section 8 on top of the same library: the colex rank, Theorem 8.2 (exact count, injectivity, round trip, surjectivity on random strings), the factorisation of Ψ , the bounds of Theorem 8.4 per node and in total, the profile count, Theorem 8.3 (sum over admissible profiles equals $n \cdot \#\text{HC}(G)$ by brute force, evaluated through the shared bit strings only; inadmissible profiles give 0), and the degree-pruned computation of I_a with its iteration counts. Besides seeded random instances ($n \leq 9$, $g \leq 4$, including hubs of degree $n - 1$), it runs four fixed instances with $g = 4$ and $n \in \{7, 8, 9\}$ on which the network without the interleaving

sign is wrong, and a built-in mutation test: with χ_a removed, or with the labels not shared, the rank-encoded network must give a wrong value on at least one of them (it does on all four). Runs with three seeds pass. An independent AI referee re-derived the construction and its parameter bounds and found it sound; it also found that the first version of the script did not exercise the interleaving sign, which led to the fixed instances and the mutation test.

12.3. Independent referee. An independent referee (an AI agent working from the one-paragraph claim) derived the sign rule on its own and wrote its own code before reading the write-up. It checked about 500 instances of “Berezin integral = determinant”, about 790 blob evaluations and 297 evaluations of a literal [A]-style network (with $g \leq 5$, random, very unequal and empty blobs, random roots, and in half of the tests random integer edge weights), with no failures; removing the global sign gave wrong answers on 8 instances. Its verdict is that the construction and bookkeeping are sound, conditional on [A] (Theorem A(b), Lemma 8(i) and the existence of pieces), and that the unbounded-degree trap is handled correctly by Theorem 7.1. It raised one wording point, about the label alphabet, which Theorem 5.1 addresses. The referee did not build or check the Lean files.

12.4. What is pending.

- Human expert review of this paper and, above all, of the labs’ manuscripts.
- A human check that the formal statements of **theorem1** and **theorem2** say what Theorems 1.1 and 1.2 say (the definitions to read are listed in Section 1.7(b)); the proofs were written by AI agents and reviewed by an AI coordinator.
- A full write-up of the extensions of Section 11 other than TSP.

12.5. Dependence on [A]. If [A, Theorem A(b)] or [A, Lemma 8] failed, Theorems 1.1 to 1.4 would lose their proofs, while Sections 3 to 6 would be unaffected. The uniformity in Theorem 1.2 uses the constant c of Theorem 2.6 through x_0 ; if c is not explicit, the statement asserts the existence of one algorithm, as in [A]. The cost model is that of [A]; our arithmetic on $O(n)$ -bit integers costs $\text{poly}(n)$ operations and is absorbed in τ and in the polynomial factors.

13. OPEN PROBLEMS

13.1. Dense graphs. The network of Section 6 has $U \approx 2 \log_2 5 \cdot |E|$ bits, and the rank-encoded network of Section 8 has $U \approx 2n \log(2e\bar{d})$ bits, so Theorem 1.3 beats 2^n only for $\bar{d} = 2^{o(\sqrt{\log n})}$, roughly. Nothing is claimed for dense graphs beyond the classical $O^*(2^n)$ [30, 2] and its $2^{n-\Omega(\sqrt{n/\log n})}$ refinement [5]. For $\bar{d} = \Theta(n)$, about $2n \log n$ bits is the information content of a Hamiltonian cycle, so for any network of this kind that shares the cycle across the blobs the leading term $e_\ell U$ is $\Omega(e_\ell n \log n)$, which is above n unless $e_\ell = o(1/\log n)$. Is there a subexponential algorithm for #HC on dense graphs, under the same hypotheses?

13.2. Connected subgraphs and $T(1, 2)$. The determinant certificate is a weight, not a 0/1 test, for objects that are connected but not trees. Exact counts of connected spanning subgraphs ($T_G(1, 2)$), of connected vertex covers and of connected dominating sets, and the whole line $x = 1$ of the Tutte plane, are not obtained. A κ -free connectivity indicator that is local in the sense of (11.1) would settle these.

13.3. Derandomization from the top-level theorems. Theorem 1.1 gives deterministic decision, but it uses [A]’s internal lemmas. Can deterministic decision be obtained from the top-level #SAT theorems alone, which are machine-checked? Any such algorithm must use the structure of the instance: if an algorithm accesses $S = \text{HC}(G) \subseteq \{0, 1\}^N$ only through adaptive queries $\sum_{x \in S} w_i(x) \bmod q_i$ with $q_i \leq Q$ and arbitrary weights w_i , then a Davenport-constant argument shows that distinguishing $S = \emptyset$ from all nonempty S requires $2^N \leq \text{lcm}(1, \dots, Q)(1 + F \ln Q)$ for F queries, so $F \geq 2^{N-o(N)}$ for $Q = 2^{(\log N)^{1/2}}$. (This generalises the observation that parity queries

alone need 2^N queries; it is related to the isolation bounds of [23].) On the other hand, a fixed list of $2^{o(n)}$ weight vectors works for all bounded-degree graphs on n vertices, and hardness assumptions for E against #P-oracle circuits would give deterministic $2^{o(n)}$ decision via [29], in the relativised form of [31]. So the question is one of explicit hitting sets. These statements have written proofs but are not formalised.

13.4. The exponent. The leading term of the exponent is $e_\ell U$, and $e_\ell = \Theta(\rho^\ell)$ is governed by the pieces of [A]. Pieces with $\rho < 1/\sqrt{2}$ would improve the exponent beyond $1/\sqrt{\log n}$, and $\rho \rightarrow \frac{1}{2}$ would give $2^{O(m/\log m)}$, and $2^{O(n \log \bar{d}/\log n)}$ in Theorem 1.3. Can the $\sqrt{\log n}$ be improved, for #HC specifically or in general?

13.5. Planar and minor-free graphs. A bound of $2^{\delta n}$ does not beat the known $2^{O(\sqrt{n})}$ on planar graphs. Does the blob network compose with balanced separators to give $2^{o(\sqrt{n})}$?

13.6. Practicality. As in [A], the bounds are astronomically far from practical. A less asymptotic analysis, or a different evaluation engine, would be needed for any concrete gain.

ACKNOWLEDGEMENTS

This work was done with AI assistance (Claude Code, Anthropic): the construction, the proofs, the numerical verification scripts, the independent referee pass and the Lean developments were produced in sessions with Claude Code. The Lean components described in Section 12 are machine-checked; everything else has been checked numerically and by an independent AI referee, but awaits human expert review. Responsibility for any errors rests with the author.

REFERENCES

- [1] Noga Alon and Zoltán Füredi, *Covering the cube by affine hyperplanes*, European J. Combin. **14** (1993), no. 2, 79–83.
- [2] Eric T. Bax, *Inclusion and exclusion algorithm for the Hamiltonian path problem*, Inform. Process. Lett. **47** (1993), no. 4, 203–207.
- [3] Richard Bellman, *Dynamic programming treatment of the travelling salesman problem*, J. ACM **9** (1962), no. 1, 61–63.
- [4] Andreas Björklund, *Determinant sums for undirected Hamiltonicity*, SIAM J. Comput. **43** (2014), no. 1, 280–299, Preliminary version in FOCS 2010.
- [5] ———, *Below all subsets for some permutational counting problems*, Proc. 15th Scandinavian Symposium and Workshops on Algorithm Theory (SWAT), LIPIcs, vol. 53, 2016, pp. 17:1–17:11.
- [6] Andreas Björklund and Thore Husfeldt, *The parity of directed Hamiltonian cycles*, Proc. 54th IEEE Symposium on Foundations of Computer Science (FOCS), 2013, pp. 727–735.
- [7] Andreas Björklund, Thore Husfeldt, Petteri Kaski, and Mikko Koivisto, *Computing the Tutte polynomial in vertex-exponential time*, Proc. 49th IEEE Symposium on Foundations of Computer Science (FOCS), 2008, pp. 677–686.
- [8] ———, *The traveling salesman problem in bounded degree graphs*, ACM Trans. Algorithms **8** (2012), no. 2, Art. 18, Preliminary version in ICALP 2008.
- [9] Andreas Björklund and Ryan Williams, *Computing permanents and counting Hamiltonian cycles by listing dissimilar vectors*, Proc. 46th International Colloquium on Automata, Languages, and Programming (ICALP), LIPIcs, vol. 132, 2019, pp. 25:1–25:14.
- [10] Hans L. Bodlaender, Marek Cygan, Stefan Kratsch, and Jesper Nederlof, *Deterministic single exponential time algorithms for connectivity problems parameterized by treewidth*, Inform. and Comput. **243** (2015), 86–111, Preliminary version in ICALP 2013.
- [11] Nicolas Bourbaki, *Algebra I. Chapters 1–3*, Elements of Mathematics, Springer, 1989.
- [12] Cornelius Brand, Holger Dell, and Marc Roth, *Fine-grained dichotomies for the Tutte plane and Boolean #CSP*, Algorithmica **81** (2019), no. 2, 541–556, Preliminary version in IPEC 2016; arXiv:1606.06581.
- [13] Chris Calabro, Russell Impagliazzo, Valentine Kabanets, and Ramamohan Paturi, *The complexity of Unique k-SAT: an isolation lemma for k-CNFs*, J. Comput. System Sci. **74** (2008), no. 3, 386–393, Preliminary version in CCC 2003.

- [14] Sergio Caracciolo, Jesper Lykke Jacobsen, Hubert Saleur, Alan D. Sokal, and Andrea Sportiello, *Fermionic field theory for trees and forests*, Phys. Rev. Lett. **93** (2004), no. 8, 080601.
- [15] Claude (Anthropic), *A $2^{o(n)}$ time algorithm for 3-SAT*, Preprint dated October 26, 2026. Paper and Lean formalisation in the repository <https://github.com/anthropics/formal-math>, folder `eth`, 2026.
- [16] Radu Curticapean, *Block interpolation: a framework for tight exponential-time counting complexity*, Inform. and Comput. **261** (2018), 265–280, Preliminary version in ICALP 2015.
- [17] Radu Curticapean, Nathan Lindzey, and Jesper Nederlof, *A tight lower bound for counting Hamiltonian cycles via matrix rank*, Proc. 29th ACM-SIAM Symposium on Discrete Algorithms (SODA), 2018, arXiv:1709.02311, pp. 1080–1099.
- [18] Marek Cygan, Fedor V. Fomin, Łukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michał Pilipczuk, and Saket Saurabh, *Parameterized algorithms*, Springer, 2015.
- [19] Marek Cygan, Stefan Kratsch, and Jesper Nederlof, *Fast Hamiltonicity checking via bases of perfect matchings*, J. ACM **65** (2018), no. 3, Art. 12, Preliminary version in STOC 2013.
- [20] Marek Cygan, Jesper Nederlof, Marcin Pilipczuk, Michał Pilipczuk, Johan M. M. van Rooij, and Jakub Onufry Wojtaszczyk, *Solving connectivity problems parameterized by treewidth in single exponential time*, Proc. 52nd IEEE Symposium on Foundations of Computer Science (FOCS), 2011, Journal version in ACM Trans. Algorithms **18** (2022), no. 2, pp. 150–159.
- [21] Leonardo de Moura and Sebastian Ullrich, *The Lean 4 theorem prover and programming language*, Automated Deduction – CADE 28, Lecture Notes in Comput. Sci., vol. 12699, Springer, 2021, pp. 625–635.
- [22] Holger Dell, Thore Husfeldt, Dániel Marx, Nina Taslamani, and Martin Wahlén, *Exponential time complexity of the permanent and the Tutte polynomial*, ACM Trans. Algorithms **10** (2014), no. 4, Art. 21.
- [23] Holger Dell, Valentine Kabanets, Dieter van Melkebeek, and Osamu Watanabe, *Is Valiant–Vazirani’s isolation probability improvable?*, Comput. Complexity **22** (2013), no. 2, 345–383.
- [24] David Eppstein, *The traveling salesman problem for cubic graphs*, J. Graph Algorithms Appl. **11** (2007), no. 1, 61–81.
- [25] Fedor V. Fomin and Kjartan Høie, *Pathwidth of cubic graphs and exact algorithms*, Inform. Process. Lett. **97** (2006), no. 5, 191–196.
- [26] Michael Held and Richard M. Karp, *A dynamic programming approach to sequencing problems*, J. Soc. Indust. Appl. Math. **10** (1962), no. 1, 196–210.
- [27] Russell Impagliazzo and Ramamohan Paturi, *On the complexity of k -SAT*, J. Comput. System Sci. **62** (2001), no. 2, 367–375.
- [28] Russell Impagliazzo, Ramamohan Paturi, and Francis Zane, *Which problems have strongly exponential complexity?*, J. Comput. System Sci. **63** (2001), no. 4, 512–530.
- [29] Russell Impagliazzo and Avi Wigderson, *$P = BPP$ if E requires exponential circuits: derandomizing the XOR lemma*, Proc. 29th ACM Symposium on Theory of Computing (STOC), 1997, pp. 220–229.
- [30] Richard M. Karp, *Dynamic programming meets the principle of inclusion and exclusion*, Oper. Res. Lett. **1** (1982), no. 2, 49–51.
- [31] Adam R. Klivans and Dieter van Melkebeek, *Graph nonisomorphism has subexponential size proofs unless the polynomial-time hierarchy collapses*, SIAM J. Comput. **31** (2002), no. 5, 1501–1526.
- [32] Samuel Kohn, Allan Gottlieb, and Meryle Kohn, *A generating function approach to the traveling salesman problem*, Proc. 1977 ACM Annual Conference, 1977, pp. 294–300.
- [33] Daniel Lokshtanov, Dániel Marx, and Saket Saurabh, *Lower bounds based on the Exponential Time Hypothesis*, Bull. Eur. Assoc. Theor. Comput. Sci. EATCS **105** (2011), 41–72.
- [34] Jesper Nederlof, *Bipartite TSP in $O(1.9999^n)$ time, assuming quadratic time matrix multiplication*, Proc. 52nd ACM Symposium on Theory of Computing (STOC), 2020, pp. 40–53.
- [35] OpenAI, *The Exponential Time Hypothesis is false*, Preprint dated 17 October 2026, revised 25 October 2026. Paper and Lean formalisation in the repository <https://github.com/openai/math>, 2026.
- [36] Ran Raz and Boris Spieker, *On the “log rank”-conjecture in communication complexity*, Combinatorica **15** (1995), no. 4, 567–588.
- [37] The mathlib Community, *The Lean mathematical library*, Proc. 9th ACM SIGPLAN International Conference on Certified Programs and Proofs (CPP), ACM, 2020, pp. 367–381.
- [38] Vaibhav Tulsyan, *eth-hamiltonicity: Lean 4 formalisation and research notes*, Public repository <https://github.com/xennygrimmato/eth-hamiltonicity>, Lean sources as of commit <https://github.com/xennygrimmato/eth-hamiltonicity/tree/ca17a3762bb21ddbe5942eeac9f29fa58123bb42>, 2026.
- [39] Mingyu Xiao and Hiroshi Nagamochi, *An exact algorithm for TSP in degree-3 graphs via circuit procedure and amortization on connectivity structure*, Algorithmica **74** (2016), no. 2, 713–741.