Exponential Time Complexity of the Permanent and the Tutte Polynomial*

Holger Dell†
University of Wisconsin–Madison, USA
holger@cs.wisc.edu

Thore Husfeldt
IT University of Copenhagen, Denmark
Lund University, Sweden
thore@itu.dk

Dániel Marx‡
Computer and Automation Research Institute, Hungarian Academy of Sciences (MTA SZTAKI), Budapest, Hungary
dmarx@cs.bme.hu

Nina Taslaman
IT University of Copenhagen, Denmark
nsta@itu.dk

Martin Wahlén
Lund University, Sweden
Uppsala University, Sweden
mva@df.lth.se

June, 2012

Abstract

We show conditional lower bounds for well-studied #P-hard problems:

- The number of satisfying assignments of a 2-CNF formula with n variables cannot be computed in time $\exp(o(n))$, and the same is true for computing the number of all independent sets in an n-vertex graph.
- The permanent of an $n \times n$ matrix with entries 0 and 1 cannot be computed in time $\exp(o(n))$.
- The Tutte polynomial of an n-vertex multigraph cannot be computed in time $\exp(o(n))$ at most evaluation points (x, y) in the case of multigraphs, and it cannot be computed in time $\exp(o(n/\operatorname{poly}\log n))$ in the case of simple graphs.

Our lower bounds are relative to (variants of) the Exponential Time Hypothesis (ETH), which says that the satisfiability of n-variable 3-CNF formulas cannot be decided in time $\exp(o(n))$. We relax this hypothesis by introducing its counting version #ETH, namely that the satisfying assignments cannot be counted in time $\exp(o(n))$. In order to use #ETH for our lower bounds, we transfer the sparsification lemma for d-CNF formulas to the counting setting.

*The journal version of this paper appears in the ACM Transactions on Algorithms [DHM+12]. Preliminary versions appeared in the proceedings of ICALP 2010 [DHW10] and IPEC 2010 [HT10].
†Research partially supported by the Alexander von Humboldt Foundation and NSF grant 1017597.
‡Research supported by ERC Starting Grant PARAMTIGHT (280152).

1. Introduction

The permanent of a matrix and the Tutte polynomial of a graph are central topics in the study of counting algorithms. Originally defined in the combinatorics literature, they unify and abstract many enumeration problems, including immediate questions about graphs such as computing the number of perfect matchings, spanning trees, forests, colourings, certain flows and orientations, but also less obvious connections to other fields, such as link polynomials from knot theory, reliability polynomials from network theory, and (maybe most importantly) the Ising and Potts models from statistical physics.

From its definition (repeated in (1) below), the permanent of an $n \times n$-matrix can be computed in $O(n!n)$ time, and the Tutte polynomial (2) can be evaluated in time exponential in the number of edges. Both problems are famously #P-hard, which rules out the existence of polynomial-time algorithms under standard complexity-theoretic assumptions, but that does not mean that we have to resign ourselves to brute-force evaluation of the definition. In fact, Ryser's famous formula [Rys63] computes the permanent with only $\exp(O(n))$ arithmetic operations, and more recently, an algorithm with running time $\exp(O(n))$ for n-vertex graphs has also been found [BHK+08] for the Tutte polynomial. Curiously, both of these algorithms are based on the inclusion–exclusion principle. In the present paper, we show that these algorithms are not likely to be significantly improved, by providing conditional lower bounds of $\exp(\Omega(n))$ for both problems.

It is clear that #P-hardness is not the right conceptual framework for such claims, as it is unable to distinguish between different types of super-polynomial time complexities. For example, the Tutte polynomial for planar graphs remains #P-hard, but can be computed in time $\exp(O(\sqrt{n}))$ [SIT95]. Therefore, we work under Impagliazzo and Paturi's *Exponential Time Hypothesis* (ETH), viz. the complexity theoretic assumption that *some* hard problem, namely the satisfiability of 3-CNF formulas in n variables, cannot be solved in time $\exp(o(n))$ [IP01]. More specifically, we introduce #ETH, a counting analogue of ETH which models the hypothesis that *counting* the satisfying assignments cannot be done in time $\exp(o(n))$.

Computing the permanent

The permanent of an $n \times n$ matrix A is defined as

$$\operatorname{per} A = \sum_{\pi \in S_n} \prod_{1 \le i \le n} A_{i\pi(i)}\,, \tag{1}$$

where S_n is the set of permutations of $\{1, \ldots, n\}$. This is redolent of the determinant from linear algebra, $\det A = \sum_\pi \operatorname{sign}(\pi) \prod_i A_{i\pi(i)}$, the only difference is an easily computable sign for every summand. However small this difference in the definition may seem, the determinant and the permanent are believed to be of a vastly different computational calibre. Both definitions involve a summation with $n!$ terms and both problems have much faster algorithms that are textbook material: The determinant can be computed in polynomial time using Gaussian elimination and the permanent can be computed in

$O(2^n n)$ operations using Ryser's formula. Yet, the determinant seems to be exponentially easier to compute than the permanent.

Valiant's celebrated #P-hardness result for the permanent [Val79] shows that no polynomial-time algorithm à la "Gaussian elimination for the permanent" can exist unless P = NP, and indeed unless P = $\mathrm{P}^{\#\mathrm{P}}$. Several unconditional lower bounds for the permanent in restricted models of computation are also known. Jerrum and Snir [JS82] have shown that monotone arithmetic circuits need $n(2^{n-1}-1)$ multiplications to compute the permanent, a bound they can match with a variant of Laplace's determinant expansion. Raz [Raz09] has shown that multi-linear arithmetic formulas for the permanent require size $\exp(\Omega(\log^2 n))$. Ryser's formula belongs to this class of formulas, but is much larger than the lower bound; no smaller construction is known. Intriguingly, the same lower bound holds for the determinant, where it is matched by a formula of size $\exp(O(\log^2 n))$ due to Berkowitz [Ber84]. One of the consequences of the present paper is that Ryser's formula is in some sense optimal under #ETH. In particular, no uniformly constructible, subexponential size formula such as Berkowitz's can exist for the permanent unless #ETH fails.

A related topic is the expression of per A in terms of det $f(A)$, where $f(A)$ is a matrix of constants and entries from A and is typically much larger than A. This question has fascinated many mathematicians for a long time, see Agrawal's survey [Agr06]; the best known bound on the dimension of $f(A)$ is $\exp(O(n))$ and it is conjectured that all such constructions require exponential size. In particular, it is an important open problem if a permanent of size n can be expressed as a determinant of size $\exp(O(\log^2 n))$. We show that under #ETH, if such a matrix $f(A)$ exists, computing f must take time $\exp(\Omega(n))$.

Computing the Tutte polynomial

The Tutte polynomial, a bivariate polynomial associated with a given graph $G = (V, E)$ with n vertices and m edges, is defined as

$$T(G; x, y) = \sum_{A \subseteq E} (x-1)^{k(A)-k(E)} (y-1)^{k(A)+|A|-|V|}, \tag{2}$$

where $k(A)$ denotes the number of connected components of the subgraph (V, A).

Despite their unified definition (2), the various computational problems given by $T(G; x, y)$ for different points (x, y) differ widely in computational complexity, as well as in the methods used to find algorithms and lower bounds. For example, $T(G; 1, 1)$ equals the number of spanning trees in G, which happens to admit a polynomial-time algorithm, curiously again based on Gaussian elimination. On the other hand, the best known algorithm for computing $T(G; 2, 1)$, the number of forests, runs in $\exp(O(n))$ time.

Computation of the Tutte polynomial has fascinated researchers in computer science and other fields for many decades. For example, the algorithms of Onsager and Fischer from the 1940s and 1960s for computing the so-called partition function for the planar Ising model are viewed as major successes of statistical physics and theoretical chemistry; this corresponds to computing $T(G; x, y)$ along the hyperbola $(x-1)(y-1) = 2$ for planar G. Many serious attempts were made to extend these results to other hyperbolas

or graph classes, but "after a quarter of a century and absolutely no progress," Feynman in 1972 observed that "the exact solution for three dimensions has not yet been found."[1]

The failure of theoretical physics to "solve the Potts model" and sundry other questions implicit in the computational complexity of the Tutte polynomial were explained only with Valiant's #P-hardness programme. After a number of papers, culminating in the work of Jaeger, Vertigan, and Welsh [JVW90], the polynomial-time complexity of exactly computing the Tutte polynomial at points (x, y) is now completely understood: it is #P-hard everywhere except at those points (x, y) where a polynomial-time algorithm is known; these points consist of the hyperbola $(x-1)(y-1) = 1$ as well as the four points $(1, 1), (-1, -1), (0, -1), (-1, 0)$.

In the present paper, we show an $\exp(\Omega(n))$ lower bound to match the $\exp(O(n))$ algorithm from [BHK+08], which holds under #ETH everywhere except for $|y| = 1$. In particular, this establishes a gap to the planar case, which admits an $\exp(O(\sqrt{n}))$ algorithm [SIT95]. Our hardness results apply (though not everywhere, and sometimes with a weaker bound) even if the graphs are sparse and simple. These classes are of particular interest because most of the graphs arising from applications in statistical mechanics arise from bond structures, which are sparse and simple.

It has been known since the 1970s [Law76] that graph 3-colouring can be solved in time $\exp(O(n))$, and this is matched by an $\exp(\Omega(n))$ lower bound under ETH [IPZ01]. Since graph 3-colouring corresponds to evaluating T at $(-2, 0)$, the exponential time complexity for $T(G; -2, 0)$ was thereby already understood. In particular, computing $T(G; x, y)$ for input G *and* (x, y) requires vertex-exponential time, an observation that is already made in [GHN06] without explicit reference to ETH.

The literature for computing the Tutte polynomial is very rich, and we make no attempt to survey it here. A recent paper of Goldberg and Jerrum [GJ08], which shows that the Tutte polynomial is hard to even approximate for large parts of the Tutte plane, contains an overview. A list of graph classes for which subexponential time algorithms are known can be found in [BHK+08].

Complexity assumptions

The standard complexity assumption P $\neq$ NP is not sufficient for our purposes: it is consistent with current knowledge that P $\neq$ NP holds and yet NP-hard problems such as 3-Sat have subexponential time algorithms. What we need is a complexity assumption stating that certain problems can be solved only in exponential time.

The exponential time hypothesis (ETH) by Impagliazzo and Paturi [IP01] is that satisfiability of 3-CNF formulas cannot be computed substantially faster than by trying all possible assignments. Formally, this reads as follows:

(ETH) There is a constant $c > 0$ such that no deterministic algorithm can decide 3-Sat in time $\exp(c \cdot n)$.

A different way of formulating ETH is to say that there is no algorithm deciding 3-Sat

[1] The Feynman quote and many other quotes describing the frustration and puzzlement of physicists around that time can be found in the copious footnotes of [Ist00].

in time $\exp(o(n))$. The latter statement is clearly implied by the above statement, and it will be more convenient for discussion to use this form and state results this way.

In two of our lower bounds, Theorem 1.2 and Theorem 1.3(iii), we need a slightly stronger assumption that rules out the possibility of randomized algorithms as well:

(rETH) There is a constant $c > 0$ such that no *randomized* algorithm can decide 3-SAT in time $\exp(c \cdot n)$ with error probability at most 1/3.

The reason why we need rETH in these two proofs is that we are reducing from the promise problem UNIQUE 3-SAT, which is 3-SAT with the promise that the given 3-CNF formula has at most one satisfying assignment. Calabro, Impagliazzo, Kabanets, *et al.* [CIK+03] established a lower bound on UNIQUE 3-SAT assuming rETH, thus our results are also relative to this complexity assumption. By reducing from UNIQUE 3-SAT, we avoid the use of interpolation, which typically weakens the lower bound by polylogarithmic factors in the exponent.

Intuitively, counting the number of solutions is much harder than deciding the existence of a solution: in the latter case, we only need to find a single solution, while in the former case we have to somehow reason about the set of all possible solutions. A formal evidence is that many natural counting problems are #P-hard and therefore not only as hard as all problems in NP but as hard as all the problems in the polynomial-time hierarchy [Tod91]. If counting problems seem to be so much harder, then it is natural to ask if their hardness can be demonstrated by a weaker complexity assumption than what is needed for the decision problems. We show that our lower bounds, with the exception of Theorem 1.2 and Theorem 1.3(iii), can be obtained using the weaker complexity assumption stating that counting the number of solutions to a 3-CNF formula requires exponential time (i.e., a counting variant of ETH).

Name #3-SAT

Input 3-CNF formula φ with n variables and m clauses.

Output The number of satisfying assignments to φ.

The best known algorithm for this problem runs in time $O(1.6423^n)$ [Kut07].

(#ETH) There is a constant $c > 0$ such that no deterministic algorithm can compute #3-SAT in time $\exp(c \cdot n)$.

ETH trivially implies #ETH whereas the other direction is not known.

By introducing the sparsification lemma, Impagliazzo, Paturi, and Zane [IPZ01] show that ETH is a robust notion in the sense that the clause width 3 and the parameter n (number of variables) in its definition can be replaced by $d \geq 3$ and m (number of clauses), respectively, to get an equivalent hypothesis, albeit the constant c may change in doing so. As most of the reductions are sensitive to the number of clauses, this stronger form of ETH is essential for proving tight lower bounds for concrete problems. In order to be able to use #ETH in such reductions, we transfer the sparsification lemma to #d-SAT and get a similar kind of robustness for #ETH.

Theorem 1.1. *Let $d \geq 3$ be an integer. Then* #ETH *holds if and only if there is a constant $c > 0$ such that no deterministic algorithm can solve* #d-SAT *in time* $\exp(c \cdot m)$.

The proof of this theorem is spelled out in Appendix A. The relationship between #ETH and the parameterized complexity of counting problems is explained in Appendix B.

Results: Counting Independent Sets

In light of Theorem 1.1, it is natural to consider the exponential time complexity of #2-Sat. Restricted to antimonotone 2-CNF formulas, this corresponds to counting *all* independent sets in a given graph, which cannot be done in time $\exp(o(n/\log^3 n))$ under #ETH [Hof10]. The loss of the poly log-factor in the exponent is due to the interpolation inherent in the hardness reduction. We avoid interpolation using the isolation lemma for d-CNF formulas by Calabro, Impagliazzo, Kabanets, *et al.* [CIK+03], and we get an asymptotically tight lower bound. The drawback is that our lower bound only holds under the randomized version of ETH instead of #ETH.

Theorem 1.2. *Under* rETH*, there is no randomized algorithm that computes the number of all independent sets in time* $\exp(o(m))$*, where* m *is the number of edges. Under the same assumption, there is no randomized algorithm for* #2-Sat *that runs in time* $\exp(o(m))$*, where* m *is the number of clauses.*

We discuss the isolation technique and prove this theorem in §2.

Results: The Permanent

For a set S of rationals we define the following problems:

Name $\textsc{Perm}^S$

Input Square matrix A with entries from S.

Output The value of $\operatorname{per} A$.

We write Perm for $\textsc{Perm}^{\mathbb{N}}$. If B is a bipartite graph with A_{ij} edges from the ith vertex in the left half to the jth vertex in the right half ($1 \leq i, j \leq n$), then $\operatorname{per}(A)$ equals the number of perfect matchings of B. Thus Perm and $\textsc{Perm}^{0,1}$ can be viewed as counting the perfect matchings in bipartite multigraphs and bipartite simple graphs, respectively. We express our lower bounds in terms of m, the number of non-zero entries of A. Without loss of generality, $n \leq m$, so the same bounds hold for the parameter n as well.

Theorem 1.3.

(i) $\textsc{Perm}^{-1,0,1}$ *and* Perm *cannot be computed in time* $\exp(o(m))$ *under* #ETH.

(ii) $\textsc{Perm}^{0,1}$ *cannot be computed in time* $\exp(o(m/\log n))$ *under* #ETH.

(iii) $\textsc{Perm}^{0,1}$ *cannot be computed in time* $\exp(o(m))$ *under* rETH.

The proof of this theorem is in §3. For (i), we follow a standard reduction by Valiant [Val79; Pap94] but use a simple equality gadget derived from [BD07] instead of Valiant's XOR-gadget, and we use interpolation to get rid of the negative weights. To establish (ii)

we simulate edge weights $w > 1$ by gadgets of size logarithmic in w, which increases the number of vertices and edges by a logarithmic factor. For (iii) we use the isolation lemma and the reduction from part (i), and we simulate the edge weights -1 without interpolation by replacing them with 2 and doing computation modulo 3. Observe that (iii) is an asymptotically tight lower bound while (ii) is not, but it also uses the stronger complexity assumption rETH instead of #ETH.

Results: The Tutte Polynomial

The computational problem $\textsc{Tutte}(x, y)$ is defined for each pair (x, y) of rationals.

Name $\textsc{Tutte}(x, y)$.

Input Undirected multigraph G with n vertices.

Output The value of $T(G; x, y)$.

In general, parallel edges and loops are allowed; we write $\textsc{Tutte}^{0,1}(x, y)$ for the special case where the input graph is simple.

Our main result is that, under #ETH, $\textsc{Tutte}(x, y)$ cannot be computed in time $\exp(o(n))$ for specific points (x, y). However, the size of the bound, and the graph classes for which it holds, varies. We summarise our results in the theorem below, see also Figure 1. For quick reference, we state the propositions in which the individual results are proved and the techniques used in each case.

Theorem 1.4. *Let* $(x, y) \in \mathbb{Q}^2$. *Under* #ETH,

(i) $\textsc{Tutte}(x, y)$ *cannot be computed in time* $\exp(o(n))$ *if* $(x-1)(y-1) \neq 1$ *and* $y \notin \{0, \pm 1\}$,
(Stretching and thickening ; Proposition 5.1 in §5)

(ii) $\textsc{Tutte}^{0,1}(x, y)$ *cannot be computed in time* $\exp(o(n))$ *if* $y = 0$ *and* $x \notin \{0, \pm 1\}$,
(Linial's reduction ; Proposition C.6 in Appendix C)

(iii) $\textsc{Tutte}^{0,1}(x, y)$ *cannot be computed in time* $\exp(o(m/\log^2 m))$ *if* $x = 1$ *and* $y \neq 1$,
(Inflation with Wump graphs ; Proposition 6.11 in §6)

(iv) $\textsc{Tutte}^{0,1}(x, y)$ *cannot be computed in time* $\exp(o(m/\log^3 m))$ *if* $(x-1)(y-1) \notin \{0, 1\}$ *and* $(x, y) \notin \{(-1, -1), (-1, 0), (0, -1)\}$.
(Inflation with Theta graphs ; Proposition 6.4 in §6)

Above, the results (iii) and (iv) are stated in terms of the parameter m, the number of edges of the given graph, but the same results also hold for the parameter n, the number of vertices, because $n \leq m$ in connected graphs. The formulation with respect to m gives a stronger hardness result under #ETH since m can potentially be much larger than n. This is in the same spirit as the sparsification lemma of Impagliazzo, Paturi, and Zane

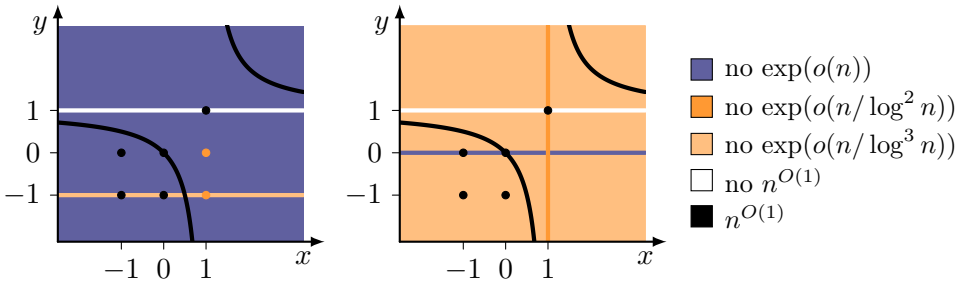


Figure 1: Exponential time complexity under #ETH of the Tutte plane for multigraphs (left) and simple graphs (right) in terms of n, the number of vertices. The white line $y = 1$ on the map is uncharted territory, and we only have the #P-hardness. The black hyperbola $(x-1)(y-1) = 1$ and the four points close to the origin are in P. Everywhere else, in the shaded regions, we prove a lower bound exponential in n, or within a polylogarithmic factor of it.

[IPZ01] and Theorem 1.1. Using this stronger formulation, Theorem 1.4 can be used as a starting point for further hardness reductions under #ETH.

In an attempt to prove Theorem 1.4, we may first turn to the literature, which contains a cornucopia of constructions for proving hardness of the Tutte polynomial in various models. In these arguments, a central role is played by graph transformations called thickenings and stretches. A k-thickening replaces every edge by a bundle of k edges ⊂⊃, and a k-stretch replaces every edge by a path of k edges ∘∼∘∼∘∼∘. This is used to "move" an evaluation from one point to another. For example, if H is the 2-stretch of G then $T(H; 2, 2) \sim T(G; 4, \frac{4}{3})$. Thus, every algorithm for $(2, 2)$ works also at $(4, \frac{4}{3})$, connecting the complexity of the two points. These reductions are very well-developed in the literature, and are used in models that are immune to polynomial-size changes in the input parameters, such as #P-hardness and approximation complexity. However, we cannot always afford such constructions in our setting, otherwise our bounds would be of the form $\exp(\Omega(n^{1/r}))$ for some constant r depending on the blow-up in the proof. In particular, the parameter n is destroyed already by a 2-stretch in a non-sparse graph.

The proofs are in §4–§6. Where we can, we sample from established methods, carefully avoiding or modifying those that are not parameter-preserving. At other times we require more subtle techniques, e.g., the constructions in §6, which use graph products with graphs of polylogarithmic size instead of thickenings and stretches. Like many recent papers, we use Sokal's multivariate version of the Tutte polynomial, which vastly simplifies many of the technical details.

Consequences

The permanent and Tutte polynomial are equivalent to, or generalisations of, various other graph problems, so our lower bounds under rETH and #ETH hold for these problems as well. In particular, the following graph polynomials (for example, as a list of their

coefficients) cannot be computed in time $\exp(o(m))$ for a given simple graph: the Ising partition function, the q-state Potts partition function ($q \neq 0, 1, 2$), the reliability polynomial, the chromatic polynomial, and the flow polynomial. Moreover, our results show that the following counting problems on multigraphs cannot be solved in time $\exp(o(n))$: # perfect matchings, # cycle covers in digraphs, # connected spanning subgraphs, all-terminal graph reliability with given edge failure probability $p > 0$, # nowhere-zero k-flows ($k \neq 0, \pm 1$), and # acyclic orientations.

The lower bound for counting the number of perfect matchings holds even in bipartite graphs, where an $O(1.414^n)$ algorithm is given by Ryser's formula. Such algorithms are also known for general graphs [BH08], the current best bound is $O(1.619^n)$ [Koi09].

For simple graphs, we have $\exp(\Omega(m))$ lower bounds for # perfect matchings and # cycle covers in digraphs.

2. Counting Independent Sets

In this section, we establish Theorem 1.2, the hardness of counting independent sets and of #2-Sat. For the proof, we make use of the randomized ETH-hardness of the following problem.

> Name Unique 3-Sat.
>
> Input 3-CNF formula φ with m clauses and at most one satisfying assignment.
>
> Decide Is φ satisfiable?

Calabro et al. [CIK+03] prove an isolation lemma for d-CNF formulas to show that solving this problem in subexponential time implies that the (randomized) exponential time hypothesis fails.

Theorem 2.1 (Corollary 2 of Calabro et al. [CIK+03]).
rETH *implies that* Unique 3-Sat *cannot be computed in time* $\exp(o(m))$.

We are now in the position to prove Theorem 1.2.

Theorem 1.2 (restated). *Under* rETH*, there is no randomized algorithm that computes the number of all independent sets in time* $\exp(o(m))$*, where* m *is the number of edges. Under the same assumption, there is no randomized algorithm for* #2-Sat *that runs in time* $\exp(o(m))$*, where* m *is the number of clauses.*

Proof. Let φ be an instance of Unique 3-Sat with m clauses. We construct a graph G with $O(m)$ edges that has an odd number of independent sets if and only if φ is satisfiable. For each variable x, we introduce vertices x and $\overline{x}$, and the edge $(x\overline{x})$. This makes sure that any independent set of G chooses at most one of $\{x, \overline{x}\}$, so we can interpret the independent set as a partial assignment to the variables of φ. For each clause $c = (\ell_1 \vee \ell_2 \vee \ell_3)$ of φ, we introduce a clique in G that consists of seven vertices $c_1, \ldots, c_7$. These vertices correspond to the seven partial assignments that assign truth values to the literals ℓ_1, ℓ_2, and ℓ_3 in such a way that c is satisfied. Any independent set of G

contains at most one c_i for each clause c. To ensure that the independent set chooses the variables and partial assignments of the clauses consistently, we add an edge for every c_i and every variable x occurring in the clause c: If the partial assignment that corresponds to c_i sets x to true, we add $(c_i\overline{x})$ to G; otherwise, we add $(c_i x)$ to G. To finalize the construction, we introduce guard vertices g_x and g_c for every variable x and every clause c, along with the edges $(g_x x)$, $(g_x\overline{x})$, and $(g_c c_i)$ for $i = 1, \ldots, 7$.

We now prove that G has the required properties. First, any independent set contains at most n literal vertices and at most m clause vertices. *Good* independent sets are those that contain exactly n literal and m clause vertices (and no guard vertex). Good independent sets correspond to the satisfying assignments of φ in a natural way. We now show that the number of bad independent sets is even. For this, let S be a bad independent set, that is, S is disjoint from $\{x, \overline{x}\}$ for some x or it is disjoint from $\{c_1, \ldots, c_7\}$ for some clause c. By construction, the neighbourhood of either g_x or g_c is disjoint from S. Let g be the lexicographically first guard vertex whose neighbourhood is disjoint from S. Both the sets $S \setminus \{g\}$ and $S \cup \{g\}$ are bad independent sets and S is one of these sets. Formally, we can therefore define a function that maps these sets onto each other. This function is a well-defined involution on the set of bad independent sets, and it does not have any fixed points. Therefore, the number of bad independent sets is even, and the parity of the number of independent sets of G is equal to the parity of the number of satisfying assignments of φ.

The above reduction shows that an $\exp(o(m))$-time algorithm for counting independent sets modulo 2 implies an $\exp(o(m))$-time algorithm for Unique 3-Sat. By Theorem 2.1, this implies that rETH fails.

To establish the hardness of #2-Sat, we reduce from counting independent sets. Let G be a graph. For each vertex v, we introduce a variable v, and each edge (uv) becomes a clause $(\overline{u} \vee \overline{v})$. The satisfying assignments of the so constructed 2-CNF formula are in one-to-one correspondence with the independent sets of G. ∎

3. The Permanent

This section contains the proof of Theorem 1.3. With $[0, n] = \{0, 1, \ldots, n\}$ we establish the reduction chain #3-Sat $\preccurlyeq$ Perm$^{-1,0,1}$ $\preccurlyeq$ Perm$^{[0,n]}$ $\preccurlyeq$ Perm^{0,1} while taking care of the instance sizes.

Theorem 1.3 (restated).

(i) Perm$^{-1,0,1}$ *and* Perm *cannot be computed in time* $\exp(o(m))$ *under* #ETH.

(ii) Perm^{0,1} *cannot be computed in time* $\exp(o(m/\log n))$ *under* #ETH.

(iii) Perm^{0,1} *cannot be computed in time* $\exp(o(m))$ *under* rETH.

Proof. To establish (i), we reduce #3-Sat in polynomial time to Perm$^{-1,0,1}$ such that 3-CNF formulas φ with m clauses are mapped to graphs G with $O(m)$ edges. For technical reasons, we preprocess φ such that every variable x occurs equally often as a

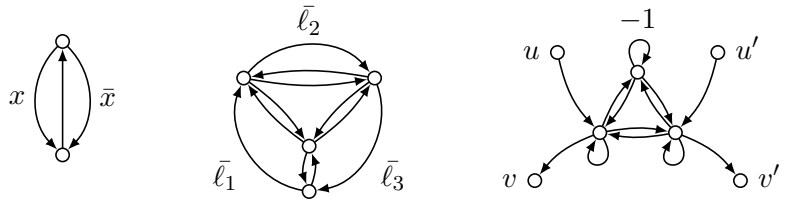


Figure 2: Left: A selector gadget for variable x. Depending on which of the two cycles is chosen, we assume x to be set to true or false. Middle: A clause gadget for the clause $\ell_1 \vee \ell_2 \vee \ell_3$. The gadget allows all possible configurations for the outer edges, except for the case that all three are chosen (which would correspond to $\ell_1 = \ell_2 = \ell_3 = 0$). Right: An equality gadget that replaces two edges uv and $u'v'$. The top loop carries a weight of -1. It can be checked that the gadget contributes a weight of -1 if all four outer edges are taken, $+2$ if none of them is taken, and 0 otherwise.

positive literal and as a negative literal $\bar{x}$ (e.g., by adding trivial clauses of the form $(x \vee \bar{x} \vee \bar{x})$ to φ). We construct G with $O(m)$ edges and weights $w : E \to \{\pm 1\}$ such that $\#\textsc{Sat}(\varphi)$ can be derived from $\operatorname{per} G$ in polynomial time. For weighted graphs, the permanent is

$$\operatorname{per} G = \sum_{C \subseteq E} w(C)\,, \qquad \text{where } w(C) = \prod_{e \in C} w(e)\,.$$

The sum above is over all cycle covers C of G, that is, subgraphs (V, C) with an in- and outdegree of 1 at every vertex.

In Figure 2, the gadgets of the construction are depicted. For every variable x that occurs in φ, we add a *selector gadget* to G. For every clause $c = \ell_1 \vee \ell_2 \vee \ell_3$ of φ, we add a *clause gadget* to G. Finally, we connect the edge labelled by a literal ℓ in the selector gadget with all occurrences of ℓ in the clause gadgets, using *equality gadgets*. That is, we use a fresh copy of the equality gadget for each occurrence of a literal. For the first occurrence of the literal, we replace the corresponding edge in the selector gadget with a path of length two and identify this path with the path from u to v in the corresponding copy of the equality gadget. Furthermore, we replace the corresponding edge in the clause gadget with a path of length two and identify this path with the path from u' to v'. For subsequent occurrences of the literal, we subdivide one of the edges on the corresponding path of the selector even further and use a new equality gadget as before. This concludes the construction of G.

The number of edges of the resulting graph G is linear in the number of clauses. The correctness of the reduction follows along the lines of [Pap94] and [BD07]. The satisfying assignments stand in bijection to cycle covers of weight $(-1)^i 2^j$ where i (resp. j) is the number of occurrences of literals set to false (resp. true) by the assignment, and all other cycle covers sum up to 0. Since we preprocessed φ such that $i = j$ holds and i is constant over all assignments, we obtain $\operatorname{per} G = (-2)^i \cdot \#\textsc{Sat}(\varphi)$.

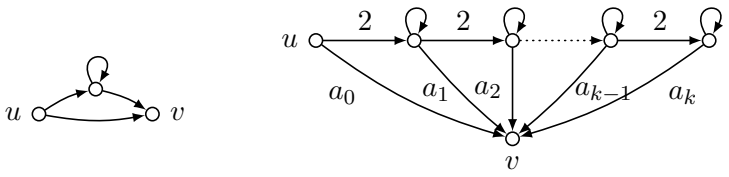


Figure 3: Left: This gadget simulates in unweighted graphs edges uv of weight 2. Right: This gadget simulates edges uv of weight $a = \sum_{i=0}^{k} a_i 2^i$ with $a_i \in \{0, 1\}$.

For the second part of (i), we reduce $\text{Perm}^{-1,0,1}$ in polynomial time to $\text{Perm}^{[0,n]}$ by interpolation: On input G, we conceptually replace all occurrences of the weight -1 by a variable x and call this new graph G_x. We can assume that only loops have weight x in G_x because the output graph G from the previous reduction has weight -1 only on loops. Then $p(x) = \operatorname{per} G_x$ is a polynomial of degree $d \leq n$.

If we replace x by a value $a \in [0, n]$, then G_a is a weighted graph with as many edges as G. As a consequence, we can use the oracle to compute $\operatorname{per} G_a$ for $a = 0, \ldots, d$ and then interpolate, to get the coefficients of the polynomial $p(x)$. At last, we return the value $p(-1) = \operatorname{per} G$. This completes the reduction, which queries the oracle $d+1$ graphs that have at most m edges each.

For (ii), we have to get rid of weights larger than 1. Let G_a be one query of the last reduction. Again we assume that $a \leq n$ and that weights $\neq 1$ are only allowed at loop edges. We replace every edge of weight a by the gadget that is drawn in Figure 3, and call this new unweighted graph G'. It can be checked easily that the gadget indeed simulates a weight of a (parallel paths correspond to addition, serial edges to multiplication), i.e., $\operatorname{per} G' = \operatorname{per} G_a$. Unfortunately, the reduction increases the number of edges by a superconstant factor: The number of edges of G' is $m(G') \leq (m + n \log a) \leq O(m + n \log n)$. But since $m(G')/\log m(G') \leq O(m)$, the reduction implies that (ii).

For (iii), we assume that rETH holds. Theorem 2.1 gives that Unique 3-Sat cannot be computed in time $\exp(o(m))$. Now we apply the first reduction of (i) to a formula φ which is promised to have at most one satisfying assignment. Then the number $\operatorname{per} G = (-2)^i \cdot \#\text{Sat}(\varphi)$ is either 0 or $(-2)^i$. In G, we replace each edge of weight -1 by a gadget of weight $2 \equiv -1 \mod 3$ and similarly get that $(\operatorname{per} G \bmod 3)$ is $(0 \bmod 3) = 0$ or $(4^i \bmod 3) = 1$. Hence we can distinguish the case in which φ is unsatisfiable from the case in which φ has exactly one satisfying assignment. ∎

4. Hyperbolas in the Tutte plane

Consider a hyperbola in the Tutte plane described by $(x-1)(y-1) = q$, where q is some fixed rational number. Our first goal is to show that it is hard to compute the coefficients of the (univariate) restriction of the Tutte polynomial to any such hyperbola. It is useful

to view the Tutte polynomial in the Fortuin–Kasteleyn formulation [FK72; Sok05]:

$$Z(G; q, w) = \sum_{A \subseteq E} q^{k(A)} w^{|A|} \,. \tag{3}$$

Here, $k(A)$ is the number of connected components in the subgraph (V, A). The connection to the Tutte polynomial is given by

$$\begin{aligned} T(G; x, y) &= (x-1)^{-k(E)}(y-1)^{-|V|} Z(G; q, w)\,, \\ &\text{where } q = (x-1)(y-1) \text{ and } w = y - 1\,, \end{aligned} \tag{4}$$

see [Sok05, eq. (2.26)].

The Ising Hyperbola

The Ising partition function is the Tutte polynomial from (3) when q is fixed to 2. We now show that computing the coefficients of this univariate polynomial is hard under #ETH.

Proposition 4.1. *If* #ETH *holds, the coefficients of the polynomial* $w \mapsto Z(G; 2, w)$ *for a given simple graph* G *cannot be computed in time* $\exp(o(m))$.

Proof. The reduction is from #MaxCut and well-known, see, *e.g.*, [JS93, Theorem 15].

Name #MaxCut

Input Simple undirected graph G.

Output The number of maximum cuts.

A maximum cut is a set $C \subseteq V(G)$ that maximizes the number $|E(C, \overline{C})|$ of edges of G that cross the cut. By the Fortuin–Kasteleyn identity [Sok05, Theorem 2.3], one can express $Z(G; 2, w)$ for $G = (V, E)$ as

$$\sum_{\sigma\colon V \mapsto \pm 1} \prod_{uv \in E} \big(1 + w \cdot [\sigma(u) = \sigma(v)]\big)\,.$$

Here the Iverson bracket $[P]$ is 1 if P is true and is 0 if P is false. The sets $\sigma^{-1}(1)$ and $\sigma^{-1}(-1)$ define a cut in G, so we can write the above expression as

$$\sum_{U \subseteq V} \prod_{\substack{uv \in E \\ [u \in U] = [v \in U]}} (1 + w) \;=\; \sum_{C \subseteq V(G)} (1 + w)^{m - |E(C, \overline{C})|}\,,$$

Now, the coefficient of $(1 + w)^{m-c}$ in $Z(G; 2, w)$ is the number of cuts in G of size c. In particular, after some interpolation, we can compute the number of maximum cuts in G from the coefficients of $w \mapsto Z(G; 2, w)$. But as we observe in Appendix C, #MaxCut cannot be computed in time $\exp(o(m))$ under #ETH. ∎

The Multivariate Tutte Polynomial

For other q, in particular non-integers, it is simpler to work with a *multivariate* formulation of the Tutte polynomial due to Fortuin and Kasteleyn [FK72]. We use the definition by Sokal [Sok05]: Let $G = (V, E)$ be an undirected graph whose edge weights are given by a function $\mathbf{w}\colon E \to \mathbb{Q}$. Then

$$Z(G; q, \mathbf{w}) = \sum_{A \subseteq E} q^{k(A)} \prod_{e \in A} \mathbf{w}(e)\,. \tag{5}$$

If $\mathbf{w}$ is single-valued, in the sense that $\mathbf{w}(e) = w$ for all $e \in E$, we recover $Z(G; q, w)$.

The conceptual strength of the multivariate perspective is that it turns the Tutte polynomial's second variable y, suitably transformed, into an edge weight of the graph. In particular, the multivariate formulation allows the graph to have different weights on different edges, which turns out to be a dramatic technical simplification even when, as in the present work, we are ultimately interested in the single-valued case.

Sokal's polynomial vanishes at $q = 0$, so we sometimes use the polynomial

$$Z_0(G; q, \mathbf{w}) = \sum_{A \subseteq E} q^{k(A)-k(E)} \prod_{e \in A} \mathbf{w}(e)\,,$$

which gives something non-trivial for $q = 0$ and is otherwise a proxy for Z:

$$Z(G; q, \mathbf{w}) = q^{k(E)} Z_0(G; q, \mathbf{w})\,. \tag{6}$$

Three-terminal minimum cut

For $q \notin \{1, 2\}$, we first establish that, with two different edge weights, one of them negative, the multivariate Tutte polynomial computes the number of 3-terminal minimum cuts:

Name #3-Terminal MinCut

Input Simple undirected graph $G = (V, E)$ with three distinguished vertices ("terminals") $t_1, t_2, t_3 \in V$.

Output The number of edge subsets $A \subseteq E$ of minimal size that separate t_1 from t_2, t_2 from t_3, and t_3 from t_1.

We establish the hardness of this problem under #ETH in Appendix C. The connection of this problem with the Tutte polynomial has been used already by Goldberg and Jerrum [GJ07; GJ08], with different reductions, to prove hardness of approximation.

The graphs we consider here are connected and have rather simple weight functions. The edges are partitioned into two sets $E \dot\cup T$ and, for some fixed rational w, the weight function is given by

$$\mathbf{w}(e) = \begin{cases} -1, & \text{if } e \in T, \\ w, & \text{if } e \in E. \end{cases} \tag{7}$$

For such a graph, we have

$$Z_0(G;q,\mathbf{w}) = \sum_{A \subseteq E \cup T} q^{k(A)-1} w^{|A \cap E|} (-1)^{|A \cap T|}. \tag{8}$$

For fixed G and q, this is a polynomial in w of degree at most m.

Lemma 4.2. *Let q be a rational number with $q \notin \{1,2\}$. The coefficients of the polynomial $w \mapsto Z_0(G;q,\mathbf{w})$, with $\mathbf{w}$ as in (7), for a given simple graph G cannot be computed in time $\exp(o(m))$ under #ETH. Moreover, this is true even if $|T| = 3$.*

Proof. In Appendix C, we argue that a standard reduction from #MaxCut already implies that the problem #3-Terminal MinCut cannot be computed in time $\exp(o(m))$ under #ETH. We reduce this problem to the problem of evaluating the coefficients of Z_0 at $q \notin \{1,2\}$. Suppose $G' = (V, E, t_1, t_2, t_3)$ is an instance of #3-Terminal MinCut with $n = |V|$ and $m = |E|$. We can assume that G' is simple and connected. We modify G' by adding a triangle between the terminals, obtaining the graph $G = (V, E \cup T)$ where $T = \{t_1t_2, t_2t_3, t_1t_3\}$; note that $n(G) = n$, $m(G) = m + 3$, and $|T| = 3$.

We focus our attention on the family $\mathscr{A}$ of edge subsets $A \subseteq E$ for which t_1, t_2, and t_3 each belong to a distinct component in the graph (V, A). In other words, A belongs to $\mathscr{A}$ if and only if $E - A$ is a 3-terminal cut in G'. Then we can split the sum in (8) into

$$Z_0(G;q,\mathbf{w}) = \sum_{B \subseteq T} \left(\sum_{A \in \mathscr{A}} q^{k(A \cup B)-1} w^{|A|} (-1)^{|B|} + \sum_{A \notin \mathscr{A}} q^{k(A \cup B)-1} w^{|A|} (-1)^{|B|} \right). \tag{9}$$

We first show that the second term of (9) vanishes. Consider an edge subset $A \notin \mathscr{A}$ and assume without loss of generality that it connects the terminals t_1 and t_2. Consider $B \subseteq T$, and let $B' = B \oplus \{t_1t_2\}$, so that B' is the same as B except for t_1t_2. Then the contributions of $A \cup B$ and $A \cup B'$ cancel: First, $k(A \cup B)$ equals $k(A \cup B')$ because t_1 and t_2 are connected through A already, so the presence or absence of the edge t_1t_2 makes no difference. Second, $(-1)^{|B|}$ equals $-(-1)^{|B'|}$.

We proceed to simplify the first term of (9). The edges in B only ever connect vertices in T, and for $A \in \mathscr{A}$, each of these lies in a separate component of (V, A), so

$$k(A \cup B) = \begin{cases} k(A) - |B|\,, & \text{if } |B| = 0, 1, 2, \\ k(A) - 2\,, & \text{if } |B| = 3. \end{cases}$$

Calculating the contribution of B for each size $|B|$, we arrive at

$$\sum_{B \subseteq T} \sum_{A \in \mathscr{A}} q^{k(A \cup B)-1} w^{|A|} (-1)^{|B|} = \sum_{A \in \mathscr{A}} q^{k(A)-1} (q^0 - 3q^{-1} + 3q^{-2} - q^{-2}) w^{|A|}\,,$$

and after some simplifications we can write (9) as

$$Z_0(G;q,\mathbf{w}) = Q \cdot \sum_{A \in \mathscr{A}} q^{k(A)-3} w^{|A|}\,, \quad \text{where } Q = (q-1)(q-2)\,. \tag{10}$$

Note that, by assumption on q, we have $Q \neq 0$.

Let us write $\sum_{i=0}^{m} d_i w^i = Q^{-1} Z_0(G; q, \mathbf{w})$, i.e., d_i is the coefficient of the monomial w^i in the sum above. More specifically,

$$Q \cdot d_i = \sum_{A \in \mathscr{A} \colon |A| = i} q^{k(A)-3}.$$

The edge subsets $A \in \mathscr{A}$ are exactly the complements of the 3-terminal cuts in G'. Now consider the family $\mathscr{C}$ of *minimal* 3-terminal cuts, all of size c. The sets $E - A$ in $\mathscr{C}$ are exactly the sets A of size $m - c$ in $\mathscr{A}$, and by minimality, $k(A) = 3$. Thus,

$$Q \cdot d_{m-c} = \sum_{A \in \mathscr{A} \colon |A| = m-c} q^{3-3} = |\mathscr{C}|.$$

Thus, if we could compute the coefficients $d_0, \ldots, d_m$ of $w \mapsto Q^{-1} Z_0(G; q, \mathbf{w})$, then we could determine the smallest c so that $d_{m-c} \neq 0$ and return $d_{m-c} = |\mathscr{C}|/Q$, the number of 3-terminal mincuts. ∎

General Hyperbolas

We use Lemma 4.2 to show that the coefficients of the univariate Tutte polynomial from (3) are hard to compute for any fixed $q \notin \{1, 2\}$. For this, we need to get rid of negative weights and reduce to a single-valued weight function. Goldberg and Jerrum [GJ08] achieve this using stretching and thickening, which we want to avoid. Since the number of edges with a negative weight is small (in fact, 3), we can use another tool: deletion–contraction.

A *deletion–contraction* identity expresses a function of the graph G in terms of two graphs $G - e$ and G/e, where $G - e$ arises from G by *deleting* the edge e () and G/e arises from G by *contracting* the edge e () that is, deleting it and identifying its endpoints (so any remaining edges between these two endpoints become loops).

It is known [Sok05, eq. (4.6)] that

$$Z(G; q, \mathbf{w}) = Z(G - e; q, \mathbf{w}) + \mathbf{w}(e) Z(G/e; q, \mathbf{w}).$$

An edge e is a *bridge* of G if deleting e from G increases the number of connected components. The above gives a deletion–contraction identity for Z_0 as well:

$$Z_0(G; q, \mathbf{w}) = \begin{cases} q Z_0(G - e; q, \mathbf{w}) + \mathbf{w}(e) Z_0(G/e; q, \mathbf{w}) & \text{if } e \text{ is a bridge,} \\ Z_0(G - e; q, \mathbf{w}) + \mathbf{w}(e) Z_0(G/e; q, \mathbf{w}) & \text{otherwise.} \end{cases} \tag{11}$$

Proposition 4.3. *Let q be a rational number with $q \notin \{1, 2\}$. The coefficients of the polynomial $v \mapsto Z_0(G; q, v)$ for a given simple graph G cannot be computed in time* $\exp(o(m))$ *under* #ETH.

By (6), this proposition also holds for Z instead of Z_0 when $q \notin \{0, 1, 2\}$.

Proof. Let $G = (V, E)$ be a graph as in the previous lemma, with three edges $T = \{e_1, e_2, e_3\}$ of weight -1. The given reduction actually uses the restriction that $G' = (V, E \setminus T)$ is connected, so we can assume that this is the case. Thus, none of the T-edges is a bridge, so three applications of (11) to delete and contract these edges, gives

$$Z_0(G; q, \mathbf{w}) = \sum_{C \subseteq \{1,2,3\}} (-1)^{|C|} Z_0(G_C; q, \mathbf{w}), \tag{12}$$

where for each $C \subseteq \{1, 2, 3\}$, the graph G_C is constructed from G by removing e_1, e_2, e_3 as follows: If $i \in C$ then e_i is contracted, otherwise it is deleted. In any case, the edges of T have disappeared and remaining edges of G_C are in one-to-one correspondence with the edges in E; especially, they all have the same weight w, so $Z_0(G_C; q, \mathbf{w}) = Z_0(G_C; q, w)$.

The resulting G_C are not necessarily simple, because the contracted edges from T may have been part of a triangle and may have produced a loop. (In fact, investigating the details of the previous lemma, we can see that this is indeed the case.) Thus we construct the simple graph G'_C from G_C by subdividing every edge into a 3-path. This operation, known as a 3-stretch, is known to largely preserve the value of Z and Z_0 (see [Sok04] for the former and [GJ08] for the latter). In particular,

$$Z_0(G_C; q, w) = f(q, w')^m \cdot Z_0(G'_C; q, w')\,,$$

where for $q \neq 0$

$$1 + \frac{q}{w} = \left(1 + \frac{q}{w'}\right)^3 \quad \text{and} \quad f(q, w') = q^{-1} \cdot ((q + w')^3 - w'^3)\,,$$

and for $q = 0$

$$w = w'/3 \quad \text{and} \quad f(q, w') = 1/(3w'^2)\,.$$

In summary, to compute the coefficients of the polynomial $w \mapsto Z_0(G; q, \mathbf{w})$, we need to compute the 8 polynomials $v \mapsto Z_0(G_C; q, v)$, one for each G_C. We use the above equation and the assumed oracle for simple graphs to do this. We note that every G'_C is simple and has at most $n + 2m$ vertices and at most $3m$ edges. ∎

5. Individual Points for Multigraphs

If we allow graphs to have multiple edges, we can use thickening and interpolation, one of the original strategies of Jaeger, Vertigan, and Welsh [JVW90], for relocating the hardness result for hyperbolas from Proposition 4.1 and Proposition 4.3 to individual points in the Tutte plane. For most points, this gives us tight bounds in terms of n, the number of vertices, but not for points with $y \in \{0, \pm 1\}$, where thickening fails completely.

We recall the thickening identities for the Tutte polynomial. The *k-thickening of G* is the graph G_k in which all edges have been replaced by k parallel edges. One can show [Sok05, (4.21)] that, with $w_k = (1 + w)^k - 1$,

$$Z(G; q, w_k) = Z(G_k; q, w)\,. \tag{13}$$

It is easy to transfer this result to the Tutte polynomial T using (4), yielding special cases of Brylawski's well-known graph transformation rules.

We use interpolation and obtain Theorem 1.4(i) for $y \neq 0$ from the following.

Proposition 5.1. *Let $(q, w) \in \mathbb{Q}^2$ with $w \notin \{0, -1, -2\}$ and $q \neq 1$.*

$Z(G; q, w)$ for a given graph G (not necessarily simple) cannot be computed in time $\exp(o(n))$ under #ETH.

Proof. We observe that the values $w_k = (1 + w)^k - 1$ are all distinct for $k = 0, 1, \ldots, m$. Thus, the k-thickenings G_k of G give rise to $m+1$ different weight shifts, the evaluations of which, $Z(G; q, w_k)$, can be obtained from $Z(G_k; q, w)$ using (13). Thus, with oracle access to $G' \mapsto Z(G'; q, w)$, we can compute the coefficients of the polynomial $v \mapsto Z(G; q, v)$ in polynomial time for any given G. By Proposition 4.1 and Proposition 4.3, this cannot be done in time $\exp(o(n))$ under #ETH. Since the number of vertices is n in each G_k, computing $G' \mapsto Z(G'; q, w)$ cannot be done in time $\exp(o(n))$ under #ETH. ∎

The proof of Theorem 1.4(ii) uses Linial's well-known reduction for the chromatic polynomial [Lin86] , and is deferred to Proposition C.6 in Appendix C.

6. Individual Points for Simple Graphs

In this section we show that most points (x, y) of the Tutte plane are as hard as the entire hyperbola on which they lie, even for sparse, simple graphs. The drawback of our method is that we lose a polylogarithmic factor in the exponent of the lower bound. The results are particularly interesting for the points on the line $y = -1$, for which we know no other good exponential lower bounds under #ETH, even in more general graph classes. We remark that the points $(-1, -1)$, $(0, -1)$, and $(\frac{1}{2}, -1)$ on this line are known to admit a polynomial-time algorithm, and indeed our hardness result does not apply here.

Graph inflations

We use the graph theoretic version of Brylawski's tensor product for matroids [Bry11]. We found the following terminology more intuitive in our setting.

Definition 6.1 (Graph inflation). *Let H be an undirected graph with two distinguished vertices called* terminals*. For any undirected graph $G = (V, E)$, an H-*inflation *of G, denoted $G \otimes H$, is obtained by replacing every edge $xy \in E$ by (a fresh copy of) H, identifying x with one of the terminals of H and y with the other.*

If H is not symmetric with respect to its two terminals, then the graph $G \otimes H$ need not be unique since there are in general two non-isomorphic ways two replace an edge xy by H. For us this difference does not matter since the resulting Tutte polynomials turn out to be the same; in fact, in any graph one can remove a maximal biconnected component and reinsert it in the other direction without changing the Tutte polynomial,

an operation that is called the *Whitney twist* [Whi33]. Thus we choose $G \otimes H$ arbitrarily among the graphs that satisfy the condition in the definition above. Graph inflation is not commutative and Sokal uses the notation $\vec{G}^H$.

If H is a simple path of k edges, $G \otimes H$ gives the usual k-stretch of G, and a bundle of k parallel edges results in a k-thickening. What makes graph inflations so useful in the study of Tutte polynomials is that the Tutte polynomial of $G \otimes H$ can be expressed in terms of the Tutte polynomials of G and H, so that $Z(G \otimes H; q, w) \sim Z(G; q, w')$ for some "shifted" weight w'.

For fixed rational points (q, w), we want to use interpolation to prove the hardness of computing $Z(G; q, w)$ for a given graph G. The basic idea is to find a suitable class of graphs $\{H_i\}$, such that we can compute the coefficients of the univariate polynomial $v \mapsto Z(G; q, v)$ for given G and q by interpolation from sufficiently many evaluations of $Z(G; q, w_i) \sim Z(G \otimes H_i; q, w)$. For this, we need that the number of different weight shifts $\{w_i\}$ provided by the graph class $\{H_i\}$ is at least $|E(G)| + 1$, one more than the degree of the polynomial.

Generalised Theta Graphs

For a set $S = \{s_1, \ldots, s_k\}$ of positive integers, the *generalised Theta graph* Θ_S consists of two vertices x and y joined by k internally disjoint paths of $s_1, \ldots, s_k$ edges, respectively. For example,

$\Theta_{\{2,3,5\}}$ is x  y .

For such graphs Θ_S, we study the behaviour of the *Theta inflation* $G \otimes \Theta_S$.

The Tutte polynomial of Theta graphs has already been studied by Sokal in the context of complex roots of the chromatic polynomial. The necessary formulas for $Z(G \otimes \Theta_S)$ can be derived from [Sok04, prop 2.2, prop 2.3]. We present them here for the special case where all edge weights are the same.

Lemma 6.2 (Sokal). *Let q and w be rational numbers with $w \neq 0$ and $q \notin \{0, -2w\}$. Then, for all graphs G and finite sets S of positive integers,*

$$Z(G \otimes \Theta_S; q, w) = q^{-|E| \cdot |S|} \cdot \prod_{s \in S} \left((q + w)^s - w^s \right)^{|E|} \cdot Z(G; q, w_S) \,, \tag{14}$$

where

$$w_S = -1 + \prod_{s \in S} \left(1 + \frac{q}{(1 + q/w)^s - 1} \right) . \tag{15}$$

This lemma can be derived from Sokal's series and parallel reduction rules for Z using a straightforward calculation. Since all edge weights are the same, the result can also be established from the classical Tutte polynomial via the series and parallel reduction rules in [JVW90], but the calculation would be slightly more laborious.

We now show that the class of Theta graphs provides a rich enough spectrum of weight shifts to allow for interpolation. In the following lemma, we use the definition of w_S from (15).

Lemma 6.3. *Let q and w be rational numbers with $w \neq 0$ and $q \notin \{0, 1, -w, -2w\}$. For all integers $m \geq 1$, there exist sets $S_0, \ldots, S_m$ of positive integers such that*

(i) $\sum_{s \in S_i} s \leq O(\log^3 m)$ *for all i, and*

(ii) $w_{S_i} \neq w_{S_j}$ *for all $i \neq j$.*

Furthermore, the sets S_i can be computed in time polynomial in m.

Proof. Let $b = |1 + q/w|$ and $f(s) = 1 + q/(b^s - 1)$ for $s > 0$. Our choice of parameters ensures that $b > 0$ and $b \neq 1$, so f is a well-defined, continuous, and strictly monotone function from $\mathbb{R}^+ \to \mathbb{R}$. Furthermore, $w_S = -1 + \prod_{s \in S} f(s)$ for all finite sets S of positive even integers. Now let $s_0 \geq 2$ be an even integer such that $f(s)$ is nonzero and has the same sign as $f(s_0)$ for all $s \geq s_0$. For $i = 0, \ldots, m$, let $b_\ell \cdots b_0$ denote the binary expansion of i where $\ell = \lfloor \log m \rfloor$. Let $\Delta > 6$ be a gap parameter that is a large and even integer chosen later, but only depends on q and w. We define

$$S_i = \left\{ s_0 + \Delta \lceil \log m \rceil \cdot (2j + b_j) \,:\, 0 \leq j \leq \ell \right\}.$$

The salient feature of this construction is that all sets S_i are different, of equal small cardinality, contain only positive even integers, and are from a range where f does not change sign. Most important for our analysis is that the elements of the S_i are spaced apart significantly, i.e.,

$$\text{for } i, j \text{ and any } s \in S_i \text{ and } t \in S_j, \text{ either } s = t \text{ or } |s - t| \geq \Delta \log m. \tag{P}$$

From $|S_i| = \lfloor \log m \rfloor + 1$ and the fact that all numbers in the sets are bounded by $O(\log^2 m)$, we immediately get (i).

To establish (ii), let $0 \leq i < j \leq m$. We want to show that $w_{S_i} \neq w_{S_j}$. Let us define $S = S_i \setminus S_j$ and $T = S_j \setminus S_i$. From (15), we see by multiplying with $(w_{S_i \cap S_j} + 1)$ on both sides that $w_S + 1 = w_T + 1$ is equivalent to $w_{S_i} = w_{S_j}$ since $w_{S_i \cap S_j} \neq -1$.

It remains to show that $\prod_{s \in S} f(s) \neq \prod_{t \in T} f(t)$. Equivalently,

$$\prod_{s \in S} \left(b^s + q - 1\right) \prod_{t \in T} \left(b^t - 1\right) - \prod_{t \in T} \left(b^t + q - 1\right) \prod_{s \in S} \left(b^s - 1\right) \neq 0 \tag{16}$$

We will multiply out the products in (16). Using the notation $\|X\| = \sum_{x \in X} x$, we rewrite

$$\prod_{s \in S} \left(b^s + q - 1\right) \prod_{t \in T} \left(b^t - 1\right) = \sum_{X \subseteq S \cup T} (-1)^{|T \setminus X|} (q-1)^{|S \setminus X|} b^{\|X\|}.$$

Here we use the convention that for $X \subseteq S \cup T$, the term b^s is taken in the first factor if $s \in X \cap S$, and b^t is taken in the second factor if $t \in X \cap T$. Doing this for both terms of (16) and collecting terms we arrive at the equivalent claim

$$\sum_{X \subseteq S \cup T} g(X) \neq 0, \tag{17}$$

where

$$g(X) = \Big((-1)^{|T\setminus X|}(q-1)^{|S\setminus X|} - (-1)^{|S\setminus X|}(q-1)^{|T\setminus X|}\Big) \cdot b^{\|X\|}\,. \tag{18}$$

Let s_1 be the smallest element of $S \cup T$ and without loss of generality assume that $s_1 \in S$ (otherwise exchange S and T). Now from (18) and $|S| = |T|$, it follows that

$$\begin{aligned} g\big(S \cup T\big) &= g(\emptyset) = 0 \\ g\big((S \cup T) \setminus \{s_1\}\big) &= q \cdot b^{\|S\cup T\| - s_1} \\ g\big(\{s_1\}\big) &= (-q) \cdot (1-q)^{|S|-1} \cdot b^{s_1}\,. \end{aligned}$$

Since $q \neq 0$, the largest exponent of b with nonzero coefficient in (18) is $\|S \cup T\| - s_1$ and all other exponents are at least $\Delta \log m$ smaller than that. Similarly since $q \notin \{0,1\}$, the smallest exponent of b with nonzero coefficient is s_1 and all other exponents are at least $\Delta \log m$ larger.

We let X_0 be the index in (17) that maximizes the value $|g(X_0)|$. By the above considerations, we have $X_0 = S \cup T \setminus \{s_1\}$ for $b > 1$ and $X_0 = \{s_1\}$ for $b < 1$. The total contribution of the remaining terms is $h = \sum_{X \neq X_0} g(X)$. We prove (17) by showing $|h| < |g(X_0)|$. From the triangle inequality and the fact that $S \cup T$ has at most $4m^2$ subsets X, we get

$$|h| \leq 4m^2 \cdot \max_{X \neq X_0} |g(X)| \leq 4m^2 \cdot 2|q-1|^{1+\log m} \cdot b^{\|X_0\| \pm \Delta \log m}$$

where the sign in $\pm\Delta \log m$ depends on whether b is larger or smaller than 1. If $b > 1$, the sign is negative. In this case, notice that $\Delta = \Delta(q,w)$ can be chosen such that $4m^2 \cdot 2|q-1|^{1+\log m} < |q| \cdot b^{\Delta \log m}$ for all $m \geq 2$. If $b < 1$, we can similarly choose Δ as to satisfy $4m^2 \cdot 2|q-1|^{1+\log m} < |q| \cdot |1-q|^{|S|-1} \cdot b^{-\Delta \log m}$. Thus, in both cases we have $|h| < |g(X_0)|$, which establishes (ii). ∎

Points on the Hyperbolas

The following proposition establishes Theorem 1.4(iv), which states that Z is hard to evaluate at most points (q,w) with $q \notin \{0,1\}$.

Proposition 6.4. *Let $(q,w) \in \mathbb{Q}^2 \setminus \{(4,-2),(2,-1),(2,-2)\}$ with $q \notin \{0,1\}$ and $w \neq 0$. If #ETH holds, then $Z(G;q,w)$ for a given simple graph G cannot be computed in time $\exp(o(m/\log^3 m))$.*

By (4), the points $(4,-2)$, $(2,-1)$, and $(2,-2)$ in the (q,w)-plane correspond to the polynomial-time computable points $(-1,-1)$, $(-1,0)$, and $(0,-1)$ in the (x,y)-plane.

Proof. We reduce from the problem of computing the coefficients of the polynomial $v \mapsto Z(G;q,v)$, which cannot be done in time $\exp(o(m))$ for $q \notin \{0,1\}$ by Proposition 4.1 and Proposition 4.3 (assuming #ETH). We interpolate as in the proof of Proposition 5.1, but instead of thickenings we use Theta inflations to keep the number of edges relatively small.

First we consider the degenerate case in which $q = -w$ or $q = -2w$. For a positive integer constant k, let G' be the k-thickening of G. This transformation shifts the weight to w' with

$$w' = (1 + w)^k - 1\,,$$

which allows us to compute $Z(G; q, w')$ from $Z(G'; q, w)$ using (13). In the case $q = -w$, we have $1 + w = 1 - q$, which cannot be 1 or 0, but which can also not be -1 since then $(q, w) = (2, -2)$. Similarly, in the case $q = -2w$, we have $1 + w = 1 - q/2$, which cannot be 1. It can also not be 0 since then $(q, w) = (2, -1)$, neither can it be -1 since then $(q, w) = (4, -2)$. Thus, in any case, $(1 + w) \notin \{0, \pm 1\}$. This means that we can choose k large enough so that $q \notin \{-w', -2w'\}$. This remains true if we let G'' be the 2-stretch to G', which shifts the weight to w'' with

$$1 + \frac{q}{w''} = \left(1 + \frac{q}{w'}\right)^2\,,$$

so that $Z(G; q, w'')$ can be computed from $Z(G''; q, w)$ (see [Sok04]). We choose k so that $q \notin \{-w'', -2w''\}$. The graph G'' after this transformation is simple and the number of edges is only increased by a constant factor of $2k$.

By the above, we can assume w.l.o.g. that $q \notin \{-w, -2w\}$. We observe that the conditions $w \neq 0$ and $q \notin \{0, 1, -w, -2w\}$ of Lemma 6.3 now hold, and thus we can compute $m+1$ sets $S_0, S_1, \ldots, S_m$ with all distinct weight shifts $w_0, \ldots, w_m$ under Theta inflations.

For a given graph G, let $G_i = G \otimes \Theta_{S_i}$. Using Lemma 6.2, we can compute the values $Z(G; q; w_i)$ from $Z(G_i; q, w)$. Moreover, as is clear from (5), the function $v \mapsto Z(G; q, v)$ is a polynomial of degree at most m, so we can use interpolation to recover its coefficients. We remark that the G_i are simple graphs with at most $O(m \log^3 m)$ edges, so the claim follows. ∎

Wump Graphs

The line $x = 1$ in the Tutte plane, the *reliability line*, is not covered by the above since here $q = 0$ holds. On this line, the Tutte polynomial specializes (up to a closed-form multiplicative factor) to the *reliability polynomial* $R(G; p)$ (with $p = 1/y$), an object studied in algebraic graph theory [GR01, Section 15.8]. Given a connected graph G and a probability p, $R(G; p)$ is the probability that G stays connected if every edge independently fails with probability p. For example $R(\text{  }; \frac{1}{3}) = Pr(\text{}) + 5Pr(\text{}) = (\frac{2}{3})^5 + 5 \cdot \frac{1}{3} \cdot (\frac{2}{3})^4 = \frac{112}{243}$. Note that $R(G; 1) = 0$ for all connected graphs, so $p = 1$ is easy to evaluate, which we know is also the case (though for less trivial reasons) for the corresponding limit point $(1, 1)$ in the Tutte plane.

Along the reliability line, weight shift identities take a different form. We use deletion–contraction identities to derive the following rules. They are simple multi-weighted generalizations of [GJ08, Section 4.3].

Lemma 6.5. *Let G be a graph with edge weights given by $\mathbf{w} : E(G) \to \mathbb{Q}$.*

If $\varphi(G)$ is obtained from G by replacing a single edge $e \in E$ with a simple path of k edges $P = \{e_1, ..., e_k\}$ with $\mathbf{w}(e_i) = w_i$, then

$$Z_0(\varphi(G); 0, \mathbf{w}) = C_P \cdot Z_0(G; 0, \mathbf{w}[e \mapsto w']) \,,$$

where

$$\frac{1}{w'} = \frac{1}{w_1} + \cdots + \frac{1}{w_k} \qquad \textit{and} \qquad C_P = \frac{1}{w'} \prod_{i=1}^{k} w_i \,.$$

Here $\mathbf{w}[e \mapsto w']$ denotes the function $\mathbf{w}' : E(G) \to \mathbb{Q}$ that is identical to $\mathbf{w}$ except at the point e where it is $\mathbf{w}'(e) = w'$.

Lemma 6.6. *If $\varphi(G)$ is obtained from G by replacing a single edge $e \in E$ with a bundle of parallel edges $B = \{e_1, \ldots, e_k\}$ with $\mathbf{w}(e_i) = w_i$, then*

$$Z_0(\varphi(G); 0, \mathbf{w}) = Z_0(G; 0, \mathbf{w}[e \mapsto w']) \,,$$

where

$$w' = -1 + \prod_{i=1}^{k} (1 + w_i) \,.$$

Corollary 6.7. *If $\varphi(G)$ is obtained from G by replacing a single edge $e \in E$ with a simple path of k edges of constant weight w, then*

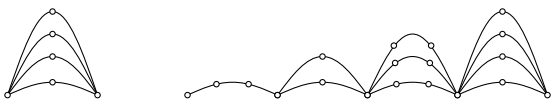
$$Z_0(\varphi(G); 0, \mathbf{w}) = k w^{k-1} \cdot Z_0(G; 0, \mathbf{w}[e \mapsto w/k]) \,, \tag{19}$$

and if it is obtained from G by replacing $e \in E$ with a bundle of k parallel edges of constant weight w, then

$$Z_0(\varphi(G); 0, \mathbf{w}) = Z_0(G; 0, \mathbf{w}[e \mapsto (1 + w)^k - 1]) \,. \tag{20}$$

These rules are transitive [GJ08, Lemma 1], and so can be freely combined for more intricate weight shifts. We define a class of graph inflations, *Wump inflations*, and use the above to show that they give rise to distinct weight shifts along the reliability line of the Tutte polynomial. Wump inflations are mildly inspired by l-byte numbers, in the sense that each has associated to it a sequence of length l, such that the lexicographic order of these sequences determines the size of the corresponding (shifted) weights.

Definition 6.8 (Wump graph). *For positive integers i (height) and s (width), an (i, s)-*hump *is the graph obtained by identifying all the left and all the right endpoints of i simple paths of length s each. Given a sequence $S = \langle s_1, s_2, \ldots, s_l \rangle$ of l positive integers, the* Wump graph *W_S is the graph obtained by concatenating l humps at their endpoints, where the i-th hump is an (i, s_i)-hump, i.e., its height is i and its width is s_i.*



$(4, 2)$-*hump* $\qquad\qquad$ $S = \langle 3, 2, 3, 2 \rangle$

The number l is the length *of the Wump graph W_S.*

Inflating a graph by a Wump graph shifts the weights on the reliability line as follows.

Lemma 6.9. *For any graph G with m edges, any sequence $S = \langle s_1, s_2, \ldots, s_l \rangle$ of positive integers, and any non-zero rational number w, we have*

$$Z_0(G \otimes W_S; 0, w) = C_S^m \cdot Z_0(G; 0, w_S)\,,$$

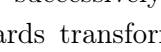
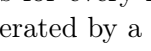
where

$$\frac{1}{w_S} = \sum_{i=1}^{l} \frac{1}{(1 + w/s_i)^i - 1} \quad \text{and} \quad C_S = \frac{1}{w_S} \cdot \prod_{i=1}^{l} w^{(s_i-1)i} \left((w + s_i)^i - s_i^i\right)\,. \tag{21}$$

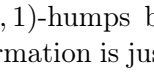
Proof. We start with $G \otimes W_S$ and consider the effect that replacing one of the m canonical copies of W_S with a single edge e has. We show that, with φ denoting this operation,

$$Z_0(G \otimes W_S; 0, w) = C_S \cdot Z_0(\varphi(G \otimes W_S); 0, \mathbf{w}[e \mapsto w_S])\,, \tag{22}$$

where w_S has the above form, and $\mathbf{w}$ has the old value w on all unaffected edges. The lemma then follows by successively applying φ to each canonical copy of W_S in $G \otimes W_S$.

The first step towards transforming a Wump graph (say, ) into a single edge, consists of contracting the paths of the humps to a single edge each. For the i-th hump, this is just the inverse of an s_i-stretching applied to each of the i paths. By (19) of Corollary 6.7, this "unstretching" gives a factor $(s_i w^{s_i-1})^i$ to the polynomial, and each edge in the resulting $(i, 1)$-hump receives a weight of w/s_i in the modified graph. Repeating this process for every hump simplifies the Wump graph into a Wump graph of length l that is generated by a sequence of 1s (). Let $\phi(G \otimes W_S)$ denote the graph in which one Wump graph has been simplified. By transitivity, we have the weight shift

$$Z_0(G \otimes W_S; 0, w) = \left(\prod_{i=1}^{l} (s_i w^{s_i-1})^i\right) \cdot Z_0(\phi(G \otimes W_S); 0, \mathbf{w}')\,,$$

where $\mathbf{w}'$ takes the value w/s_i on every edge of the ith hump of the simplified Wump graph, and the old value w outside the simplified Wump graph. Next, we successively replace each of its $(i, 1)$-humps by a single edge to get a simple path () of length l. This transformation is just an "unthickening" of each $(i, 1)$-hump, and from (20) of Corollary 6.7 we know that it does not produce any new factors for the polynomial, but the weight of the ith edge in this path becomes

$$w_i = (1 + w/s_i)^i - 1\,.$$

Finally, we compress the path into a single edge e. Then the claim in (22) follows by a single application of Lemma 6.5. ∎

We now show that Wump inflations provide a rich enough class of weight shifts. The ranges of w for which we prove this is general enough to allow for interpolation on the whole reliability line, and we make no attempt at extending the ranges. In the following lemma, we use the definition of w_S from (21).

Lemma 6.10. *Let w be a rational number with $w \in (-1, 0)$ or $w \in (9, \infty)$. For all integers $m \geq 1$, there exist sequences $S_0, \ldots, S_m$ of positive integers such that*

(i) $|E(W_{S_i})| \leq O(\log^2 m)$ for all i, and

(ii) $w_{S_i} \neq w_{S_j}$ for all $i \neq j$.

Furthermore, the sequences S_i can be computed in time polynomial in m.

Proof. We consider the set of sequences $S = \langle s_1, \ldots, s_l \rangle$ of length $l = r \log(m+1)$, with $s_i \in \{2, 3\}$ for all i which are positive integer multiples of r, and $s_i = 2$ for all other i. Here r is a positive integer and will be chosen later, only depending on w. Since r is a constant, this construction satisfies (i).

Now consider any two distinct sequences $S = \langle s_i \rangle$ and $T = \langle t_i \rangle$. To show (ii), we consider the difference

$$\Delta = \frac{1}{w_S} - \frac{1}{w_T},$$

and show that $\Delta \neq 0$.

Using Lemma 6.9 we get a sum expression for Δ.

$$\begin{aligned} \Delta &= \sum_{i=1}^{l} \frac{1}{(1 + w/s_i)^i - 1} - \sum_{i=1}^{l} \frac{1}{(1 + w/t_i)^i - 1} \\ &= \sum_{i=1}^{l} g\left((1 + w/s_i)^i\right) - \sum_{i=1}^{l} g\left((1 + w/t_i)^i\right), \end{aligned} \tag{23}$$

where g is the function $g(x) = \frac{1}{x-1}$. This function is negative and strictly decreasing on $(0, 1)$ and positive and strictly decreasing on $(1, \infty)$. It is convenient to choose $a, b \in \{(1 + w/3), (1 + w/2)\}$ so that $a < b$. By the monotonicity of g, we have $g(a^i) > g(b^i)$ for all positive i.

Case 1: $w > 9$. Here we have $a = (1 + w/3)$ and $b = (1 + w/2)$. We set $r = 1$ and let k be the smallest index for which the sequences differ, i.e., $s_k \neq t_k$. We assume w.l.o.g. that $s_k = 3$ and $t_k = 2$, otherwise we exchange the roles of S and T. In (23), terms of the sum for $i < k$ cancel. The terms corresponding to $i = k$ are $g(a^k) - g(b^k) > 0$. We apply the monotonicity of g to the terms for $i > k$, which allows us to lower bound Δ as follows.

$$\Delta \geq g(a^k) + \sum_{i=k+1}^{l} g(b^i) - g(b^k) - \sum_{i=k+1}^{l} g(a^i) = f(a) - f(b),$$

where

$$f(x) = g(x^k) - \sum_{i=k+1}^{l} g(x^i) = \frac{1}{x^k - 1} - \sum_{i=k+1}^{l} \frac{1}{x^i - 1}. \tag{24}$$

We now claim that f is strictly decreasing in $(4, \infty)$. This implies $\Delta > 0$ since $w > 9$ guarantees $a, b > 4$, and we get $\Delta \geq f(a) - f(b) > 0$. To prove the claim, we show that

the derivative of f is negative on $(4,\infty)$. This is a routine calculation, but we include it here for completeness. We have

$$f'(x) = -\frac{kx^{k-1}}{(x^k-1)^2} + \sum_{i=k+1}^{l} \frac{ix^{i-1}}{(x^i-1)^2}\,. \tag{25}$$

The terms of the sum here, let us call them $T_i(x)$, satisfy

$$T_i(x) > 2\cdot T_{i+1}(x)$$

for all i and all $x>4$. To see this, note that the inequality is equivalent to

$$2\left(1+\frac{1}{i}\right)x < \left(x+\frac{x-1}{x^i-1}\right)^2\,.$$

This statement is true for all reals $x>4$ and all positive integers i since then we have that LHS $\le 4x < x^2 \le$ RHS. Thus, for $x>4$, we have

$$f'(x) < \frac{kx^{k-1}}{(x^k-1)^2}\left(-1+\sum_{i=k+1}^{l}\frac{1}{2^{i-k}}\right) < 0\,.$$

Case 2: $w\in(-1,0)$. Here we have $a=(1+w/2)$ and $b=(1+w/3)$. We choose r to be a positive integer that satisfies $b^r<\frac{1}{4}$. Let rk be the smallest index for which the sequences differ, i.e., $s_{rk}\neq t_{rk}$. We assume w.l.o.g. that $s_{rk}=3$ and $t_{rk}=2$, otherwise we exchange the roles of S and T. In (23), terms of the sum for $i<rk$ cancel, and so do terms for those i's which are not integer multiples of r. The terms corresponding to $i=rk$ are $g(b^{rk})-g(a^{rk})<0$. We apply the monotonicity of g to the remaining terms for $i>rk$, which allows us to upper bound Δ as follows.

$$\Delta \le g(b^{rk}) + \sum_{i=k+1}^{l/r} g(a^{ri}) - g(a^{rk}) - \sum_{i=k+1}^{l/r} g(b^{ri})$$

For $x\in(0,1)$, we can expand $g(x)$ into the geometric series

$$g(x) = \frac{1}{x-1} = -\sum_{j=0}^{\infty} x^j\,.$$

Applying this representation to our estimate for Δ and rearranging terms, we arrive at

$$\Delta \le \sum_{j=0}^{\infty}\left((a^{rj})^k-(b^{rj})^k+\sum_{i=k+1}^{l/r}\left((b^{rj})^i-(a^{rj})^i\right)\right) = \sum_{j=0}^{\infty}\left(F(a^{rj})-F(b^{rj})\right)\,,$$

where F is the function

$$F(y) = y^k - \sum_{i=k+1}^{l/r} y^i\,.$$

We claim that F is strictly increasing on $(0, \frac{1}{4})$. This, together with the fact that r is chosen such that $a^{rj}, b^{rj} \in (0, \frac{1}{4})$ for all positive integers j, implies $\Delta < 0$, because then $F(a^{rj}) - F(b^{rj}) < 0$ for $j \geq 1$, and for $j = 0$ the term is 0. To prove the claim we show that the derivative of F is positive on $(0, \frac{1}{4})$. Again, we give the details here for completeness. We have

$$F'(y) = ky^{k-1} - \sum_{i=k+1}^{l/r} iy^{i-1}\,,$$

and obtain $F'(y) > 0$ from the following calculation, using the fact that $y \in (0, \frac{1}{4})$.

$$\begin{aligned}(ky^{k-1})^{-1} \cdot \sum_{i=k+1}^{l/r} iy^{i-1} &= \sum_{i=k+1}^{l/r} \frac{i}{k} y^{i-k} = \sum_{i=1}^{l/r-k} \left(1 + \frac{i}{k}\right) y^i \\ &\leq \sum_{i=1}^{l/r-k} (1+i)\, y^i \leq \sum_{i=1}^{\infty} y^i + \sum_{i=1}^{\infty} iy^i \\ &= \frac{1}{1-y} - 1 + \frac{y}{(1-y)^2} \leq \frac{4}{3} - 1 + \frac{4}{9} < 1\,. \end{aligned}$$

■

Points on the Reliability Line

We prove Theorem 1.4(iii).

Proposition 6.11. *Let $w \neq 0$ be a rational number. If #ETH holds, then $Z_0(G; 0, w)$ for a given simple graph G cannot be computed in time $\exp(o(m/\log^2 m))$.*

Proof. If $w < 0$, we can pick a positive integer k big enough such that

$$w' := w/k > -1\,.$$

This weight shift corresponds to the k-stretch of G (Corollary 6.7). On the other hand, if $w > 0$, we can pick a positive integer k such that

$$w' := (w/2 + 1)^k - 1 > 9\,.$$

This is the weight shift that corresponds to the 2-stretch of the k-thickening of G (Corollary 6.7). In any case we can compute $Z(G; w', q)$ from $Z(G'; w, q)$. The graph remains simple after any of these transformations, and the number of edges is only increased by a constant factor of at most $2k$.

By the above, we can assume w.l.o.g. that $w \in (-1, 0)$ or $w > 9$. We use Lemma 6.10 to construct $m + 1$ Wump graphs W_S whose corresponding weight shifts w_S are all distinct by property (ii) of Lemma 6.10. By Lemma 6.9, we can compute the values $Z_0(G; 0, w_S)$ from $Z_0(G \otimes W_S; 0, w)$, i.e., we get evaluations of $v \mapsto Z_0(G; 0, v)$ at $m+1$ distinct points. Since the degree of this polynomial is m, we obtain its coefficients by interpolation. By Proposition 4.3, these coefficients cannot be computed in time $\exp(o(m))$ under #ETH. By Lemma 6.10(i), each $G \otimes W_S$ has at most $O(m \log^2 m)$ edges, which implies that $Z_0(G; 0, w)$ for given G cannot be computed in time $\exp(o(m/\log^2 m))$ as claimed. ■

7. Conclusion and Further Work

Our results for the Tutte polynomial leave open the line $y = 1$ except for the point $(1, 1)$, even in the case of multigraphs. That line corresponds to counting the number of forest weighted by the number of edges, i.e., $T(G; 1 + 1/w, 1) \sim F(G; w) = \sum_{\text{forests } F} w^{|F|}$. Thickening and Theta inflation, with the analysis in the proof of Lemma 6.9, suffice to show that every point is as hard as computing the coefficients of $F(G; w)$, without increasing the number of vertices for multigraphs and with an increase in the number of edges by a factor of $O(\log^2 m)$ in the case of simple graphs. However, we do not know whether computing those coefficients requires exponential time under #ETH. And of course, it would be nice to improve our conditional lower bounds $\exp(\Omega(n/\operatorname{poly}\log n))$ to match the corresponding upper bounds $\exp(O(n))$.

Acknowledgements

The authors are grateful to Andreas Björklund, Leslie Ann Goldberg, and Dieter van Melkebeek for valuable comments.

Wump graphs are named for a fictional creature notable for its number of humps, which appears in the American children's book "One Fish Two Fish Red Fish Blue Fish" by Dr. Seuss; the name was suggested by Prasad Tetali.

References

[Agr06] Manindra Agrawal, "Determinant versus permanent," in *Proceedings of the 25th International Congress of Mathematicians, ICM 2006*, vol. 3, 2006, pp. 985–997.

[BD07] Markus Bläser and Holger Dell, "Complexity of the cover polynomial," in *Proceedings of the 34th International Colloquium on Automata, Languages and Programming, ICALP 2007*, ser. Lecture Notes in Computer Science, vol. 4596, Springer, 2007, pp. 801–812. DOI: 10.1007/978-3-540-73420-8_69.

[Ber84] Stuart J. Berkowitz, "On computing the determinant in small parallel time using a small number of processors," *Information Processing Letters*, vol. 18, no. 3, pp. 147–150, 1984. DOI: 10.1016/0020-0190(84)90018-8.

[BH08] Andreas Björklund and Thore Husfeldt, "Exact algorithms for exact satisfiability and number of perfect matchings," *Algorithmica*, vol. 52, no. 2, pp. 226–249, 2008. DOI: 10.1007/s00453-007-9149-8.

[BHK+08] Andreas Björklund, Thore Husfeldt, Petteri Kaski, and Mikko Koivisto, "Computing the Tutte polynomial in vertex-exponential time," in *Proceedings of the 47th annual IEEE Symposium on Foundations of Computer Science, FOCS 2008*, 2008, pp. 677–686. DOI: 10.1109/FOCS.2008.40.

[Bry11] Thomas Brylawski, "The Tutte polynomial part I: General theory," in *Matroid Theory and its Applications*, ser. Centro Internazionale Matematico Estivo Summer Schools, vol. 83, Springer, 2011, pp. 125–275. DOI: 10.1007/978-3-642-11110-5_3.

[CIK+03] Chris Calabro, Russell Impagliazzo, Valentine Kabanets, and Ramamohan Paturi, "The complexity of unique k-SAT: An isolation lemma for k-CNFs," in *Proceedings of the 18th IEEE Conference on Computational Complexity, CCC 2003*, 2003, p. 135. DOI: 10.1109/CCC.2003.1214416.

[CJ01] Liming Cai and David W. Juedes, "Subexponential parameterized algorithms collapse the W-hierarchy," in *Proceedings of the 28th International Colloquium on Automata, Languages and Programming, ICALP 2001*, 2001, pp. 273–284.

[DECF+03] Rodney G. Downey, Vladimir Estivill-Castro, Michael R. Fellows, Elena Prieto, and Frances A. Rosamund, "Cutting up is hard to do: the parameterised complexity of k-cut and related problems," *Electronic Notes in Theoretical Computer Science*, vol. 78, pp. 209–222, 2003. DOI: 10.1016/S1571-0661(04)81014-4.

[DHM+12] Holger Dell, Thore Husfeldt, Dániel Marx, Nina Taslaman, and Martin Wahlén, "Exponential time complexity of the permanent and the Tutte polynomial," *Transactions on Algorithms*, 2012+, to appear.

[DHW10] Holger Dell, Thore Husfeldt, and Martin Wahlén, "Exponential time complexity of the permanent and the Tutte polynomial," in *Proceedings of the 37th International Colloquium on Automata, Languages and Programming, ICALP 2010*, ser. Lecture Notes in Computer Science, vol. 6198, Springer, 2010, pp. 426–437. DOI: 10.1007/978-3-642-14165-2_37.

[DJP+94] Elias Dahlhaus, David S. Johnson, Christos H. Papadimitriou, Paul D. Seymour, and Mihalis Yannakakis, "The complexity of multiterminal cuts," *SIAM Journal on Computing*, vol. 23, no. 4, pp. 864–894, 1994. DOI: 10.1137/S0097539792225297.

[FG04] Jörg Flum and Martin Grohe, "The parameterized complexity of counting problems," *SIAM Journal on Computing*, no. 4, pp. 892–922, 2004. DOI: 10.1137/S0097539703427203.

[FG06] ——, *Parameterized Complexity Theory*. Springer, 2006, ISBN: 978-3-540-29952-3.

[FK72] Cees M. Fortuin and Pieter W. Kasteleyn, "On the random-cluster model: I. Introduction and relation to other models," *Physica*, vol. 57, no. 4, pp. 536–564, 1972, ISSN: 0031-8914. DOI: 10.1016/0031-8914(72)90045-6.

[GHN06] Omer Giménez, Petr Hliněný, and Marc Noy, "Computing the Tutte polynomial on graphs of bounded clique-width," *SIAM Journal on Discrete Mathematics*, vol. 20, pp. 932–946, 2006. DOI: 10.1007/11604686_6.

[GJ07] Leslie Ann Goldberg and Mark Jerrum, "The complexity of ferromagnetic Ising with local fields," *Combinatorics, Probability and Computing*, vol. 16, no. 1, pp. 43–61, 2007. DOI: 10.1017/S096354830600767X.

[GJ08] ——, "Inapproximability of the Tutte polynomial," *Information and Computation*, vol. 206, no. 7, pp. 908–929, 2008. DOI: 10.1016/j.ic.2008.04.003.

[GJS76] Michael R. Garey, David S. Johnson, and Larry Stockmeyer, "Some simplified NP-complete graph problems," *Theoretical Computer Science*, vol. 1, no. 3, pp. 237–267, 1976. DOI: 10.1016/0304-3975(76)90059-1.

[GR01] Chris Godsil and Gordon Royle, *Algebraic Graph Theory*, ser. Graduate Texts in Mathematics. Springer, Apr. 2001, ISBN: 0387952209.

[Hof10] Christian Hoffmann, "Exponential time complexity of weighted counting of independent sets," in *Proceedings of the 5th International Symposium on Parameterized and Exact Complexity, IPEC 2010*, ser. Lecture Notes in Computer Science, vol. 6478, Springer, 2010, pp. 180–191. DOI: 10.1007/978-3-642-17493-3_18.

[HT10] Thore Husfeldt and Nina Taslaman, "The exponential time complexity of computing the probability that a graph is connected," in *Proceedings of the 5th International Symposium on Parameterized and Exact Complexity, IPEC 2010*, ser. Lecture Notes in Computer Science, vol. 6478, Springer, 2010, pp. 192–203. DOI: 10.1007/978-3-642-17493-3_19.

[IP01] Russel Impagliazzo and Ramamohan Paturi, "On the complexity of k-SAT," *Journal of Computer and System Sciences*, vol. 62, no. 2, pp. 367–375, 2001. DOI: 10.1006/jcss.2000.1727.

[IPZ01] Russell Impagliazzo, Ramamohan Paturi, and Francis Zane, "Which problems have strongly exponential complexity?," *Journal of Computer and System Sciences*, vol. 63, no. 4, pp. 512–530, 2001. DOI: 10.1006/jcss.2001.1774.

[Ist00] Sorin Istrail, "Statistical mechanics, three-dimensionality and NP-completeness. I. Universality of intractability for the partition function of the Ising model across non-planar lattices," in *Proceedings of the 32nd annual ACM Symposium on Theory of Computing, STOC 2000*, 2000, pp. 87–96. DOI: 10.1145/335305.335316.

[JS82] Mark Jerrum and Marc Snir, "Some exact complexity results for straight-line computations over semirings," *Journal of the ACM*, vol. 29, no. 3, pp. 874–897, 1982. DOI: 10.1145/322326.322341.

[JS93] Mark Jerrum and Alistair Sinclair, "Polynomial-time approximation algorithms for the Ising model," *SIAM Journal on Computing*, vol. 22, no. 5, pp. 1087–1116, 1993. DOI: 10.1137/0222066.

[JVW90] François Jaeger, Dirk L. Vertigan, and Dominic J.A. Welsh, "On the computational complexity of the Jones and Tutte polynomials," *Mathematical proceedings of the Cambridge Philosophical Society*, vol. 108, no. 1, pp. 35–53, 1990. DOI: 10.1017/S0305004100068936.

[Koi09] Mikko Koivisto, "Partitioning into sets of bounded cardinality," in *Proceedings of the 4th International Workshop on Parameterized and Exact Complexity, IWPEC 2009*, ser. Lecture Notes in Computer Science, vol. 5917, Springer, 2009, pp. 258–263. DOI: 10.1007/978-3-642-11269-0_21.

[Kut07] Konstantin Kutzkov, "New upper bound for the #3-sat problem," *Information Processing Letters*, vol. 105, no. 1, pp. 1–5, 2007. DOI: 10.1016/j.ipl.2007.06.017.

[Law76] Eugene L. Lawler, "A note on the complexity of the chromatic number problem," *Information Processing Letters*, vol. 5, no. 3, pp. 66–67, 1976. DOI: 10.1016/0020-0190(76)90065-X.

[Lin86] Nathan Linial, "Hard enumeration problems in geometry and combinatorics," *SIAM Journal on Algebraic and Discrete Methods*, vol. 7, no. 2, pp. 331–335, 1986. DOI: 10.1137/0607036.

[Pap94] Christos H. Papadimitriou, *Computational Complexity*. Addison-Wesley, 1994, ISBN: 978-0-201-53082-7.

[Raz09] Ran Raz, "Multi-linear formulas for permanent and determinant are of super-polynomial size," *Journal of the ACM*, vol. 56, no. 2, pp. 1–17, 2009. DOI: 10.1145/1502793.1502797.

[Rys63] Herbert J. Ryser, "Combinatorial mathematics," *Number 14 in Carus Math. Monographs. Mathematical Association of America*, 1963.

[SIT95] Kyoko Sekine, Hiroshi Imai, and Seiichiro Tani, "Computing the Tutte polynomial of a graph of moderate size," in *Proceedings of the 6th International Symposium on Algorithms and Computation, ISAAC 1995*, ser. Lecture Notes in Computer Science, Springer, 1995, pp. 224–233. DOI: 10.1007/BFb0015427.

[Sok04] Alan D. Sokal, "Chromatic roots are dense in the whole complex plane," *Combinatorics, Probability and Computing*, vol. 13, no. 2, pp. 221–261, 2004. DOI: 10.1017/S0963548303006023.

[Sok05] ——, "The multivariate Tutte polynomial (alias Potts model) for graphs and matroids," in *Surveys in Combinatorics*, ser. London Mathematical Society Lecture Note Series, vol. 327, 2005, pp. 173–226.

[Tod91] Seinosuke Toda, "PP is as hard as the polynomial-time hierarchy," 5, vol. 20, 1991, pp. 865–877. DOI: 10.1137/0220053.

[Val79] Leslie G. Valiant, "The complexity of computing the permanent," *Theoretical Computer Science*, vol. 8, no. 2, pp. 189–201, 1979. DOI: 10.1016/0304-3975(79)90044-6.

[Whi33] Hassler Whitney, "2-isomorphic graphs," *American Journal of Mathematics*, vol. 55, no. 1, pp. 245–254, 1933. [Online]. Available: http://www.jstor.org/stable/2371127.

A. The Sparsification Lemma

Sparsification is the process of reducing the density of graphs, formulas, or other combinatorial objects, while some properties of the objects like the answer to a computational problem are preserved.

The objective of sparsification is twofold. From an algorithmic perspective, efficient sparsification procedures can be used as a preprocessing step to make input instances sparse and thus possibly simpler and smaller, such that only the core information about the input remains. In the literature, such applications of sparsification procedures are called kernelizations. From a complexity-theoretic point of view, sparsification is a tool to identify those instances of a problem that are computationally the hardest. If an NP-hard problem admits efficient sparsification, the hardest instances are sparse.

In the context of the exponential time hypothesis, the sparsification lemma provides a way to show that the hardest instances of d-SAT are sparse and thus the parameter n can be replaced with m in the statement of the exponential time hypothesis. The following is the sparsification lemma as formulated in [FG06, Lemma 16.17].

Lemma A.1 (Sparsification Lemma). *Let $d \geq 2$. There exists a computable function $f : \mathbb{N}^2 \to \mathbb{N}$ such that for every $k \in \mathbb{N}$ and every d-CNF formula γ with n variables, we can find a formula*

$$\beta = \bigvee_{i \in [t]} \gamma_i$$

such that:

(1) β is equivalent to γ (ie., they have the same satisfying assignments),

(2) $t \leq 2^{n/k}$, and

(3) the γ_i are d-CNF formulas in which each variable occurs at most $f(d,k)$ times.

Furthermore, β can be computed from γ and k in time $t \cdot \mathrm{poly}(n)$.

We sketch below a small modification in the proof of the sparsification lemma that allows us to replace (1) with the condition

(1′) $\mathrm{sat}(\gamma) = \dot{\bigcup}_i \mathrm{sat}(\gamma_i)$,

where $\mathrm{sat}(\varphi)$ is the set of assignments that satisfy the formula φ. That is, not only is β equivalent to γ, it even holds that every satisfying assignment of β satisfies exactly one γ_i. In particular, *(1′)* implies $\#\textsc{Sat}(\gamma) = \sum_i \#\textsc{Sat}(\gamma_i)$, which means that the sparsification lemma can be used for the counting version of 3-SAT.

Proof (sketch). We adapt the terminology of [FG06, Proof of Lemma 16.17] and we follow their construction precisely, except for a small change in the sparsification algorithm. When the algorithm decides to branch for a CNF-formula γ and a flower $\alpha = \{\delta_1, \ldots, \delta_p\}$, the original algorithm would branch on the two formulas

$$\gamma^{\alpha}_{\text{heart}} = \gamma \setminus \{\delta_1, \ldots, \delta_p\} \cup \{\delta\},$$
$$\gamma^{\alpha}_{\text{petals}} = \gamma \setminus \{\delta_1, \ldots, \delta_p\} \cup \{\delta_1 \setminus \delta, \ldots, \delta_p \setminus \delta\}.$$

We modify the branching on the petals to read

$$\gamma^{\alpha}_{\text{petals}} = \gamma \setminus \{\delta_1, \ldots, \delta_p\} \cup \{\delta_1 \setminus \delta, \ldots, \delta_p \setminus \delta\} \cup \big\{\ \{\neg l\} : l \in \delta\ \big\}.$$

This way, the satisfying assignments become disjoint: In each branching step, we guess whether the heart contains a literal set to true, or whether all literals in the heart are set to false and each petal contains a literals set to true.

Now we have that, for all CNF-formulas γ, all assignments σ to the variables of γ, and all flowers α of γ,

(i) σ satisfies γ if and only if σ satisfies $\gamma^{\alpha}_{\text{heart}} \vee \gamma^{\alpha}_{\text{petals}}$, and

(ii) σ does not satisfy $\gamma^{\alpha}_{\text{heart}}$ or σ does not satisfy $\gamma^{\alpha}_{\text{petals}}$.

By induction, we see that at the end of the algorithm,

(i) σ satisfies γ if and only if σ satisfies some γ_i, and

(ii) σ satisfies at most one γ_i.

This implies that $\text{sat}(\gamma) = \dot{\bigcup}_{i\in[t]} \text{sat}(\gamma_i)$.

Notice that our new construction adds at most n clauses of size 1 to the formulas γ_i compared to the old one. Furthermore, our construction does not make t any larger because the REDUCE-step removes all clauses that properly contain $\{\neg l\}$ and thus these unit clauses never appear in a flower. ∎

Proof (of Theorem 1.1). For all integers $d \geq 3$ and $k \geq 1$, the sparsification lemma gives an oracle reduction from #d-SAT to #d-SAT that, on input a formula γ with n variables, only queries formulas with $m' = O(n)$ clauses, such that the reduction runs in time $\exp(O(n/k))$. Now, if for every $c > 0$ there is an algorithm for #d-SAT that runs in time $\exp(cm)$, we can combine this algorithm and the above oracle reduction to obtain an algorithm for #d-SAT that runs in time $\exp(O(n/k)+c\cdot m') = \exp(O(n/k)+c\cdot O(n))$. Since this holds for all small $c > 0$ and large k, we have for every $c' > 0$ an algorithm for #d-SAT running in time $\exp(c'\cdot n)$. This proves that for all $d \geq 3$, #d-SAT can be solved in variable-subexponential time if and only if it can be solved in clause-subexponential time.

It remains to show that #d-SAT reduces to #3-SAT. We transform an instance φ of #d-SAT into an instance φ' of #3-SAT that has the same number of satisfying assignments. The formula φ' is constructed as in the standard width-reduction for d-CNF formulas, i.e., by introducing a constant number of new variables for every clause of φ. Thus, since the number of clauses of φ' is $O(m)$, any clause-subexponential algorithm for #3-SAT implies a clause-subexponential algorithm for #d-SAT. ∎

B. Parameterized Complexity

Our hypothesis #ETH relates to parameterized complexity, which is a branch of computational complexity that considers problems in terms of two parameters n and k. Of

special interest in that field are problems that have algorithm whose running times are of the form $f(k)\,\mathrm{poly}(n)$ for some computable function f. Such problems are called fixed parameter tractable, or FPT.

Flum and Grohe [FG04] introduce the class #W[1] of parameterized counting problems. This class is characterized by complete problems such as computing the number of cliques of size k or computing the number of simple paths of length k in an n-vertex graph. Implicitly, Flum and Grohe [FG04] show that these problems are not fixed-parameter tractable under #ETH.

Theorem B.1 (Flum and Grohe). *If* #ETH *holds, then* #W[1] $\neq$ FPT.

The latter is only an implication and, as in the case of decision problems, we do not know whether the two claims are equivalent. For a claim that is equivalent to a uniform variant of #ETH, we can follow a construction due to Downey, Estivill-Castro, Fellows, *et al.* [DECF+03]. They consider the following problem:

> Name #Mini-3-Sat
>
> Input Integers k and n; a 3-CNF formula φ with at most $k \log n$ clauses.
>
> Output The number of satisfying assignments of φ.

Without explicit reference to ETH, Downey et al. [DECF+03] (based on ideas of Cai and Juedes [CJ01]) prove that the decision version of this problem is equivalent to a uniform variant of ETH. By a straightforward modification of their reduction, one can establish the following equivalence (see also [FG06, chapter 16]).

Theorem B.2 (Downey et al.). *The following two statements are equivalent.*

(i) There is no computable function $T(n) \leq 2^{o(n)}$ such that #3-Sat *has a deterministic algorithm that runs in time $T(n)$ for n-variable formulas.*

(ii) #Mini-3-Sat $\notin$ FPT.

C. Hardness of 3-Colouring and 3-Terminal MinCut

The purpose of this section is to show that the standard reductions from 3-Sat to 3-Colouring, NAE-3-Sat, MaxCut, and 3-Terminal MinCut computationally preserve the number of solutions and increase the number of clauses or edges of the instances by at most a constant factor. This implies that the corresponding counting problems cannot be computed in clause-subexponential or edge-subexponential time unless #ETH fails.

Theorem C.1. *The problems* #NAE-3-Sat, #MaxCut, #3-Terminal MinCut, *and* #3-Colouring *cannot be deterministically computed in time* $\exp(o(m))$ *unless* #ETH *fails.*

In the following, we formally define the problems, sketch the standard NP-hardness reductions, and provide their analyses as needed to prove Theorem C.1. For the purposes of this section, *polynomial-time reductions* between counting problems are oracle reductions that make at most one query. The reductions we sketch need not be parsimonious, that is, they map instances of one problems to instances of another problem (which they query), but the number of solutions need not be exactly equal. In fact, there is no parsimonious reduction from #3-Sat or #NAE-3-Sat to #MaxCut since every graph has at least one maximum cut while not every formula is satisfiable. Similarly, reductions from #3-Sat to #3-Terminal MinCut cannot be parsimonious.

Not-all-equal-Sat

We show that counting the number of all not-all-equal assignments is hard even for the promise problem in which we only have inputs with at least one such assignment. A truth assignment is a *not-all-equal assignment* if all constraints $\{a, b, c\} \in \varphi$ contain a true *and* a false truth value. Formally, we use the following promise version of #NAE-3-Sat.

Name #NAE-3-Sat$^+$

Input 3-CNF formula φ with at least one not-all-equal assignment.

Output The number of not-all-equal assignments.

Lemma C.2. *There is a polynomial-time reduction from* #3-Sat *to* #NAE-3-Sat$^+$ *that maps formulas with m clauses to formulas with $O(m)$ clauses.*

Proof. Let ψ be a 3-CNF formula with n variables and m clauses. To fulfil the promise, we first plant a satisfying assignment using a popular homework assignment. We obtain a 3-CNF formula φ with $O(m)$ variables and clauses such that $\#\textsc{Sat}(\varphi) = \#\textsc{Sat}(\psi)+1$.

To construct the instance φ' to NAE-3-Sat, we introduce a new variable x for every trivariate clause $(a \vee b \vee c)$ of φ, and we replace that clause with

$$(x \vee \overline{a}) \wedge (x \vee \overline{b}) \wedge (\overline{x} \vee a \vee b) \wedge (x \vee c)\,.$$

These clauses force x to have the same value as $a \vee b$ in any satisfying assignment. It can be checked that these clauses are satisfied exactly if the original clause was satisfied and moreover that the trivariate clause is never all-false or all-true. In total, we increased the number of clauses four-fold without changing the number of satisfying assignments.

Finally, introduce a single fresh variable z and add this variable (positively) to every mono- and bivariate clause. It is well-known that this modification turns φ' into an instance φ'' of NAE-3-Sat [Pap94, Theorem 9.3]: The not-all-equal assignments of φ'' are exactly the satisfying assignments of φ' (if z is set to false) or their complements (if z is set to true).

The reduction computes φ'' from ψ in polynomial time, φ'' has at most $O(m)$ clauses, and we have $\#\text{NAE-3-}\textsc{Sat}(\varphi'') = 2 \cdot (\#\textsc{Sat}(\psi) + 1)$. ∎

Maximum Cut

A *cut* is a set $C \subseteq V(G)$ and its *size* is the number $|E(C, \overline{C})|$ of edges of G that cross the cut. A *maximum cut* is a cut $C \subseteq V(G)$ of maximum size.

Name #MaxCut

Input Simple undirected graph G.

Output The number of maximum cuts.

Jerrum and Sinclair [JS93, Lemma 13] modify a reduction of Garey, Johnson, and Stockmeyer [GJS76, Theorem 1.1 and Theorem 1.2] to show #P-hardness of this problem. The reduction increases the number of edges quadratically, so we cannot use it. Instead, we use the reduction in [Pap94, Theorem 9.5] and compose it with a 3-stretch to make the graph simple. The reduction is from #NAE-3-Sat$^+$ to #MaxCut.

Lemma C.3. *There is a polynomial-time reduction from* #NAE-3-Sat$^+$ *to* #MaxCut *that maps formulas with m clauses to graphs with $O(m)$ edges.*

Proof. We use the same reduction as [Pap94, Theorem 9.5] and we repeat the details here for completeness. Given an instance φ of NAE-3-Sat with n variables and m constraints, we construct a graph G as follows: For every variable x_i, we add adjacent vertices x_i and $\neg x_i$. For every constraint $\{a, b, c\}$ of φ, we further add a triangle between the three involved literals, which possibly leads to multiedges. This multigraph G has $2n$ vertices and $3m + n$ edges.

With $k = 2m + n$, we claim that the number of cuts of size k is equal to the number of not-all-equal assignments of φ. First notice that there are no cuts of size larger than k: every constraint triangle either contributes zero or two edges to any cut C, so every cut has at most $2m$ edges from constraint triangles of G. Except for triangle edges, there are exactly n further edges in the graph, so the cut cannot be larger than $2m + n = k$. Also note that if any x_j and $\neg x_j$ are on the same side of a cut, then the size of that cut cannot exceed $k - 1$. Hence every cut C of size exactly k separates all pairs x_i and $\neg x_i$ and can be seen as a truth assignment to the variables of φ. Furthermore, since C has size exactly k, it cuts every constraint triangle, so it corresponds to a not-all-equal truth assignment of φ. For the other direction, any cut constructed from a not-all-equal assignment separates all x_i and $\neg x_i$, and cuts every triangle, so the size of such cuts is k. In particular, since we reduced from an instance φ that has at least one not-all-equal assignment, the maximum cuts of G have size k. We obtain a parsimonious polynomial-time reduction from #NAE-3-Sat$^+$ to #MaxCut on multigraphs that increases the parameters n and m at most by a constant factor.

We now reduce #MaxCut for multigraphs to simple graphs. Let G be a multigraph with m edges and with a maximum cut of size k. Let G' be the 3-stretch of G, that is, every edge is replaced by a path with three edges. This graph has $3m$ edges, and we claim that $\#\textsc{MaxCut}(G') = 3^{m-k} \cdot \#\textsc{MaxCut}(G)$, which suffices to prove the reduction.

To prove the claim, let C be a maxcut of G. We think of C as a colouring $C : V(G) \to \{0, 1\}$ such that the number of bichromatic edges is maximized. The colouring C can be

extended in 3^{m-k} ways to a maximum cut of G' as follows. We consider an edge $\{u, v\}$ of G that got stretched into a 3-path u, a, b, v.

(1) If $C(u) = C(v)$, then there are exactly three ways to colour a and b such that the number of bichromatic edges on the path u, a, b, v is two. Furthermore, no extension can yield more than two bichromatic edges.

(2) If $C(u) \neq C(v)$, then there is exactly one way in which colouring can be extended to a and b such that the number of bichromatic edges on the path u, a, b, v is three.

Since C has k bichromatic edges and $m-k$ monochromatic edges in G, it can be extended in 3^{m-k} ways to yield a colouring of G' with $2(m-k)+3k = 2m+k = k'$ bichromatic edges. On the other hand, any other extension than the above, as well as any extension of cuts C of size smaller than k lead to cuts of G' that have size smaller than k'. ∎

Minimum cut between three terminals

For convenience, we restate the definition of #3-Terminal MinCut from §4.

Name #3-Terminal MinCut

Input Simple undirected graph $G = (V, E)$ with three distinguished vertices ("terminals") $t_1, t_2, t_3 \in V$.

Output The number of cuts of minimal size that separate t_1 from t_2, t_2 from t_3, and t_3 from t_1.

Lemma C.4. *There is a polynomial-time reduction from the* #MaxCut *problem to* #3-Terminal MinCut *that maps graphs with m edges to graphs with $O(m)$ edges.*

Proof. We follow the reduction of Dahlhaus et al. [DJP+94, Theorem 3]. So let $G = (V, E)$ be a simple graph with n vertices and m edges. It is made explicit in [DJP+94] that the construction builds a graph F with $n' = 3 + n + 4m = O(m)$ vertices. For the number of edges, every $uv \in E$ results in a gadget graph C with 18 edges, so the number of edges in F is $18m = O(m)$. The construction is such that the number of minimum 3-terminal cuts of F equals the number of maximum cuts of G. ∎

Three-colouring

Name #3-Colouring

Input Simple undirected graph G.

Output The number of proper vertex-colourings with three colours.

Impagliazzo, Paturi, and Zane [IPZ01] already observed the hardness of 3-Colouring under ETH. This can be extended to the counting version as follows.

Lemma C.5. *There is a polynomial-time reduction from the* #NAE-3-Sat *problem to* #3-Colouring *that maps formulas with m clauses to graphs with $O(m)$ edges.*

Proof. We follow the proof of [Pap94, Theorem 9.8]. The graph G that is constructed from an NAE-3-SAT-instance φ with n variables and m clauses has $n' = 1 + 2n + 3m$ vertices and $m' = 3n + 6m$ edges. Furthermore, every not-all-equal assignment to the variables of φ gives rise to exactly $3 \cdot 2^m$ proper 3-colourings of G: There are 3 possible colours for a and a variable assignment then uniquely colours the $2n$ vertices that correspond to literals (take the smaller of the remaining colours to mean false and the larger to mean true; since complements of not-all-equal assignments are also not-all-equal assignments, this choice prevents overcounting). Now the colouring can be extended to each clause gadget in exactly two ways. Hence the number of proper 3-colourings of G is equal to $3 \cdot 2^m \cdot \#\text{NAE-3-SAT}(\varphi)$. ∎

Proof (of Theorem C.1). Assume one of the problems can be solved in time $\exp(cm)$ for every $c > 0$. Then #3-SAT can be solved by first applying the applicable reductions of the preceding lemmas and then invoking the assumed algorithm. This gives for every $c > 0$ an algorithm for #3-SAT that runs in time $\exp(O(cm))$, which implies that #ETH fails. ∎

C.1. Hardness of Colouring and Other Individual Points on the Chromatic Line

Theorem 1.4(ii) cannot be handled by the proof of Proposition 5.1 because thickenings do not produce enough points for interpolation. Instead, we use a reduction for the chromatic line that was discovered by Linial [Lin86].

The chromatic polynomial $\chi(G; q)$ of G is the polynomial in q with the property that, for all $c \in \mathbb{N}$, the value $\chi(G; c)$ is the number of proper c-colourings of the vertices of G. We write $\chi(q)$ for the function $G \mapsto \chi(G; q)$. The Tutte polynomial specializes to the chromatic polynomial for $y = 0$:

$$\chi(G; q) = (-1)^{n(G)-k(G)} q^{k(G)} T(G; 1 - q, 0)\,. \tag{26}$$

The following two propositions establish Theorem 1.4(ii).

Proposition C.6. *Let $x \in \{-2, -3, \ldots\}$.*
If #ETH *holds, then* $\text{TUTTE}^{0,1}(x, 0)$ *cannot be computed in time* $\exp(o(m))$.

Proof. Set $q = 1 - x$. Since $q \neq 0$, it follows from (26) that evaluating $\text{TUTTE}(x, 0)$ is equivalent to evaluating the chromatic polynomial $\chi(q)$ at point q. In particular, $\chi(3)$ is the number of 3-colourings. By Theorem C.1, if #ETH is true, $\chi(3)$ cannot be computed in time $\exp(o(m))$ even for simple graphs. For $i \in \{1, 2, \ldots\}$ and all real r, Linial's identity is

$$\chi(G + K_i; r) = r(r-1)\ldots(r-i+1) \cdot \chi(G; r-i)\,, \tag{27}$$

where $G + K_i$ is the simple graph consisting of G and a clique K_i on i vertices, each of which is adjacent to every vertex of G.

For $q \in \{4, 5, \ldots\}$, we can set $i = q - 3$ and directly compute $\chi(G; 3) = \chi(G; q - i) = \chi(G + K_i; q)/[q(q-1)\cdots 4]$. Since $m(G + K_i) = m(G) + i \cdot n(G) + \binom{i}{2} \leq O(m(G))$, it follows that $\chi(q)$ cannot be computed in time $\exp(o(m))$ under #ETH, even for simple graphs. ∎

Proposition C.7. *Let $x \notin \mathbb{Q} \setminus \{1, 0, -1, -2, -3, \ldots\}$.*
If #ETH holds, then $\textsc{Tutte}^{0,1}(x, 0)$ cannot be computed in time $\exp(o(n))$.

Proof. Set $q = 1-x$. We show that $\textsc{Tutte}^{0,1}(x, 0)$ cannot be computed in time $\exp(o(n))$ under #ETH. Indeed, with access to $\chi(q)$, we can compute $\chi(G; q-i)$ for all $i = 0, \ldots, n$, noting that all prefactors in (27) nonzero. From these $n + 1$ values, we interpolate to get the coefficients of the polynomial $r \mapsto \chi(G; r)$, which in turn allows us evaluate $\chi(G; 3)$. In this case, the size of the oracle queries depends non-linearly on the size of G, in particular $m(G + K_n) \sim n^2$. However, the number of vertices is $n(G + K_i) \leq 2n \leq O(m(G))$. Thus, since $\chi(3)$ cannot be computed in time $\exp(o(n))$ under #ETH, this also holds for $\chi(q)$, even for simple graphs. ∎

The only points on the x-axis not covered here are $x \in \{1, 0, -1\}$. Two of these admit polynomial-time algorithms, so we expect no hardness result. By Theorem 1.4(iii), the Tutte polynomial at the point $(1, 0)$ cannot be evaluated in time $\exp(o(m/\log^2 m))$ under #ETH.