Modular Counting of Subgraphs: Matchings, Matching-Splittable Graphs, and Paths

Radu Curticapean
Basic Algorithm Research Copenhagen (BARC), IT University of Copenhagen, Denmark

Holger Dell
Goethe Universität Frankfurt, Germany
Basic Algorithm Research Copenhagen (BARC), IT University of Copenhagen, Denmark

Thore Husfeldt
Basic Algorithm Research Copenhagen (BARC), IT University of Copenhagen, Denmark
Lund University, Sweden

Abstract

We systematically investigate the complexity of counting subgraph patterns *modulo fixed integers*. For example, it is known that the *parity* of the number of k-matchings can be determined in polynomial time by a simple reduction to the determinant. We generalize this to an $n^{f(t,s)}$-time algorithm to compute modulo 2^t the number of subgraph occurrences of patterns that are s vertices away from being matchings. This shows that the known polynomial-time cases of subgraph *detection* (Jansen and Marx, SODA 2015) carry over into the setting of *counting modulo* 2^t. Complementing our algorithm, we also give a simple and self-contained proof that counting k-matchings modulo odd integers q is $\mathrm{Mod}_q\mathrm{W}[1]$-complete and prove that counting k-paths modulo 2 is $\oplus\mathrm{W}[1]$-complete, answering an open question by Björklund, Dell, and Husfeldt (ICALP 2015).

2012 ACM Subject Classification Theory of computation → Fixed parameter tractability; Theory of computation → Problems, reductions and completeness

Keywords and phrases Counting complexity, matchings, paths, subgraphs, parameterized complexity

Digital Object Identifier 10.4230/LIPIcs.ESA.2021.34

Related Version *Full Version*: https://arxiv.org/abs/2107.00629

Funding Supported by VILLUM Foundation grant 16582.

1 Introduction

The last two decades have seen the development of several complexity dichotomies for pattern counting problems in graphs, including full classifications for counting *subgraphs*, *induced subgraphs*, and *homomorphisms* from fixed computable pattern classes $\mathcal{H}$. The input to such problems is a *pattern* graph $H \in \mathcal{H}$ and an unrestricted *host* graph G; the task is to count the relevant occurrences of H in G. Depending on $\mathcal{H}$, these problems are known to be either polynomial-time solvable or #W[1]-hard when parameterized by $|V(H)|$. The latter rules out polynomial-time algorithms under the complexity assumption FPT $\neq$ #W[1].

In this paper, we focus on counting *subgraphs* from any fixed graph class $\mathcal{H}$. On the positive side, given a pattern graph $H \in \mathcal{H}$ whose smallest vertex-cover has size $\mathrm{vc}(H)$ and an n-vertex host graph G, there are known $O(n^{\mathrm{vc}(H)+1})$ time algorithms [40, 29, 9] to count subgraphs of G that are isomorphic to H: First, find a minimum vertex-cover C of H using exhaustive search. Then, iterate over all possible embeddings f of $H[C]$ into G and count the possible extensions of $G[f(C)]$ to a full copy of H. Complementing this algorithm, an almost matching running time lower bound of $n^{\Omega(\mathrm{vc}(H)/\log \mathrm{vc}(H))}$ under the exponential-time hypothesis (ETH) is also known [8]. Thus, assuming ETH or FPT $\neq$ #W[1], the problem

$\#\mathrm{Sub}(\mathcal{H})$ of counting subgraphs from a fixed class $\mathcal{H}$ is polynomial-time solvable if and only if the vertex-cover numbers (or equivalently, the maximum matching sizes) of the graphs in $\mathcal{H}$ are bounded by a constant. The rightmost column of Figure 1 visualizes this situation.

Turning from counting to the problem $\mathrm{Sub}(\mathcal{H})$ of *detecting* subgraphs from fixed classes $\mathcal{H}$, the picture is less clear. Evidence points at three strata of complexity: Define the *matching-split number* of H to be the minimum number of vertices whose deletion turns H into a matching, that is, a graph of maximum degree 1. Jansen and Marx [25] show that, if this number is bounded in a graph class $\mathcal{H}$, then $\mathrm{Sub}(\mathcal{H})$ is polynomial-time solvable. For classes $\mathcal{H}$ of bounded tree-width, it is known [34, 1, 19] that the problem $\mathrm{Sub}(\mathcal{H})$ is fixed-parameter tractable when parameterized by $|V(H)|$. For pattern classes $\mathcal{H}$ of unbounded tree-width, it is conjectured that $\mathrm{Sub}(\mathcal{H})$ is W[1]-hard – so far, this hardness has only been established for cliques, bicliques [30], grids [6], and less natural graph classes. The leftmost column of Figure 1 visualizes the situation.

We propose to study an intermediate setting between decision and counting, namely, counting subgraph patterns *modulo fixed integers* $q \in \mathbf{N}$. Modular counting has a tradition in classical complexity theory, where the complexity classes $\mathrm{Mod}_q\mathrm{P}$ for $q \in \mathbf{N}$ capture problems that ask to count accepting paths of polynomially time-bounded non-deterministic Turing machines modulo q. In particular, the class $\mathrm{Mod}_2\mathrm{P}$ (better known as $\oplus\mathrm{P}$) plays a central role in the proof of Toda's theorem [38]. Several (partial) classification results for frameworks of modular counting problems are known; this includes homomorphisms to fixed graphs [16, 20, 21, 26, 18], constraint satisfaction problems [15, 22], and Holant problems [11].

Figure 1 summarizes our understanding. If the vertex-cover number is bounded, the polynomial-time algorithms (regions 7 and 8) follow from the algorithm for $\#\mathrm{Sub}(\mathcal{H})$ described above and require no further attention. Our paper is concerned with the remaining regions 1–6.

As argued above, matchings play a central role in decision and counting, so it is natural that they reprise their role in modular subgraph counting: On the positive side, there are known polynomial-time algorithms for counting matchings of a given size modulo fixed powers of two. (For bipartite graphs and counting modulo 2, this essentially follows from the fact that determinant and permanent coincide modulo 2.) On the negative side, if q is not a power of two, counting matchings modulo q is known to be $\mathrm{Mod}_p\mathrm{P}$-complete for any odd prime p dividing q. We establish a parameterized analogue of this fact: Let $\mathrm{Mod}_q\mathrm{W}[1]$ be the class of parameterized problems that are fpt-reducible to counting k-cliques modulo q. We show that counting k-matchings (that is, sets of k pairwise disjoint edges) in graphs modulo fixed odd primes $q \in \mathbf{N}$ is $\mathrm{Mod}_q\mathrm{W}[1]$-hard. In our proof, modular counting allows us to sidestep the algebraic machinery from previous works [9, 7, 8], resulting in a surprisingly simple and self-contained argument.

▶ **Theorem 1.** *For any integer $q \in \mathbf{N}$ containing an odd prime factor p, counting k-matchings modulo q is $\mathrm{Mod}_p\mathrm{W}[1]$-hard under Turing fpt-reductions and admits no $n^{o(k/\log k)}$ time algorithm under* ETH.

Known arguments from Ramsey theory (see [10, Section 5]) extend Theorem 1 from matchings to $\#\mathrm{Sub}(\mathcal{H})$ mod q for any hereditary class $\mathcal{H}$ of unbounded vertex-cover number. This suggests that modular subgraph counting may only become tractable when the modulus is a power of two. Indeed, we show that patterns of matching-split number s can be counted modulo $q = 2^t$ in time $n^{O(t4^s)}$. To prove this, we follow the general idea of the bounded vertex-cover number algorithm for $\mathrm{Sub}(\mathcal{H})$ outlined before, and we reduce to counting matchings modulo powers of two. This however requires us to overcome technical complications to avoid unwanted cancellations. Overall, we obtain:

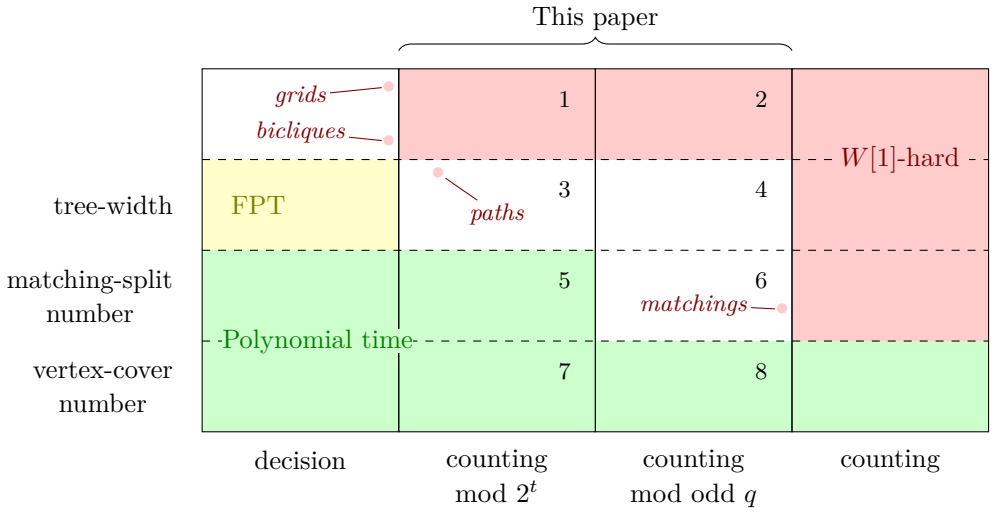


Figure 1 An overview over known results and our new results. The columns correspond, from left to right, to the problem types Sub($\mathcal{H}$), #Sub($\mathcal{H}$) mod 2^t, #Sub($\mathcal{H}$) mod q for $q \neq 2^t$, or #Sub($\mathcal{H}$); our results are depicted in the two middle columns. The rows correspond, from bottom to top, to requiring $\mathcal{H}$ to have bounded vertex-cover number, matching-split number, tree-width, or no requirement at all. The complexity along each row is monotone: By Lemma 4, decision is no harder than modular counting, and modular counting trivially is no harder than counting. Our results are depicted in the middle two columns: Regions 1 and 2 are Lemma 5. Region 5 is Theorem 2. Regions 7 and 8 already follow from [40]. The point in region 6 is Theorem 1, and the point in region 3 is Theorem 3. We view the hardness of these points as evidence to conjecture their enclosing regions to be hard, see Conjecture 14.

▶ **Theorem 2.** *There is an algorithm that, given a graph H of matching-split number $s \in \mathbf{N}$ and an n-vertex graph G, computes the number of H-isomorphic subgraphs of G modulo 2^t in time $n^{O(t4^s)}$.*

We complement this result in two ways: First, we observe that $\oplus$Sub($\mathcal{H}$) is $\oplus$W[1]-complete for pattern classes $\mathcal{H}$ of unbounded tree-width; this follows directly from previous hardness proofs for #Sub($\mathcal{H}$). More interestingly, we establish the $\oplus$W[1]-completeness of counting k-paths modulo 2 in undirected graphs, thus solving an open problem from [3], where this problem was considered in the context of Hamiltonian cycle detection, following [4].

▶ **Theorem 3.** *Counting k-paths modulo 2 is $\oplus$W[1]-complete.*

This result adds to a rich range of previous work on the k-path problem, and is of interest outside our framework. Bodlaender [5] and Monien [32] showed that *finding* a k-path is fixed-parameter tractable. In contrast, Flum and Grohe [17] showed that *exactly counting* k-paths is #W[1]-hard. Nevertheless, Arvind and Raman [2] showed that *approximately counting* k-paths, which corresponds to computing the most significant bit(s) of the number of k-paths, is fixed-parameter tractable. Our Theorem 3 suggests that the *least* significant bit of the number of k-paths is hard to compute. This is surprising, because some of the most influential fpt-algorithms for finding a k-path work over characteristic 2, based on the group algebra framework introduced by Koutis [28].

Let us conclude with a general remark on the techniques used in this paper: Recent works successfully exploited a connection between subgraph counts and (linear combinations of) homomorphism counts to obtain algorithms and hardness results [8, 35, 14, 36, 37]. For

example, the number of k-matchings in a graph G is a linear combination of homomorphism counts from $f(k)$ fixed graphs. Insights on the complexity of counting the homomorphisms occurring in this linear combination then lead to complexity results for counting k-matchings. This connection however does not readily transfer to modular counting, as the relevant linear combinations (which involve rational coefficients) may be *undefined* modulo p. We therefore prove Theorems 1–3 using more combinatorial approaches.

2 Preliminaries

Unless otherwise stated, we consider finite, undirected, simple graphs without self-loops.

Subgraph problems

A *homomorphism* from graph H to graph G is a mapping $\varphi\colon V(H) \to V(G)$ such that $\{\varphi(u), \varphi(v)\} \in E(G)$ for each $\{u, v\} \in E(H)$. An *embedding* is an injective homomorphism, and we let $\mathrm{Emb}(H, G)$ denote the set of embeddings from H to G. An *isomorphism* is a bijective homomorphism, and an *automorphism* is an isomorphism from H to itself. The set of all automorphisms of H is called $\mathrm{Aut}(H)$, and forms a group when endowed with function composition $\circ$.

We let $\mathrm{Sub}(H, G)$ be the set of all H-subgraphs of G, that is, the set of all H' with $V(H') \subseteq V(G)$ and $E(H') \subseteq E(G)$ such that H' is isomorphic to H. This terminology fixes the possible confusion about isomorphic copies of subgraphs: For example, there is exactly one K_k-subgraph in K_k, but there are $k!$ embeddings. The *subgraph problem* Sub is given a pair (H, G) to decide whether G has at least one H-subgraph. The *subgraph counting problem* #Sub is given a pair (H, G) to determine the number of H-subgraphs in G.

For a graph class $\mathcal{H}$, we write $\#\mathrm{Sub}(\mathcal{H})$ for the restricted problem where the input (H, G) is promised to satisfy $H \in \mathcal{H}$. For $q \in \mathbf{Z}_{\geq 2}$, the *modular subgraph counting problem* $\#\mathrm{Sub}(\mathcal{H})$ mod q is the problem to compute the number of H-subgraphs modulo q. In the special case with $q = 2$, we write $\oplus\mathrm{Sub}$.

It will be useful to consider *colorful* subgraph problems, where G is H-colored, that is, there is a given homomorphism $c : V(G) \to V(H)$. Due to the homomorphism property, we allow edges $\{u, v\} \in E(G)$ only if the corresponding colors satisfy $\{c(u), c(v)\} \in E(H)$. A subgraph H' of an H-colored graph G is *vertex-colorful* if c is bijective on $V(H')$. Let $\mathrm{VertexColorfulSub}(H, G)$ be the set of vertex-colorful subgraphs H' for which c is an isomorphism from H' to H. The corresponding computational problems are defined analogously to the uncolored case; the input consists of a graph G together with an H-coloring c.

Background from complexity theory

A *parameterized counting problem* is a pair (f, κ) of functions $f, \kappa : \{0,1\}^* \to \mathbf{N}$ where κ is computable. A *parsimonious fpt-reduction* from a parameterized counting problem (f, κ) to a parameterized counting problem (g, ι) is a function R with the following properties: (i) $f(x) = g(R(x))$ for all $x \in \{0,1\}^*$, (ii) $\iota(R(x))$ is bounded by a computable function in $\kappa(x)$, and (iii) the reduction is computable in time $h(\kappa(x))\,\mathrm{poly}(|x|)$ for some computable function h. A *Turing fpt-reduction* may query the oracle multiple times for instances whose parameter is bounded by a function of the input parameter, and combine the query answers in fpt-time to produce the correct output. Moreover, reductions can also be *randomized*, in which case we require that their error probability is bounded by a small constant.

The *exponential-time hypothesis* (ETH) postulates the existence of some $\varepsilon > 0$ such that no algorithm solves n-variable 3-CNF formulas in time $O(2^{\varepsilon n})$. We write for short that 3-CNF-SAT does not have $2^{o(n)}$-time algorithms, and we also disallow bounded-error randomized algorithms.

Modular counting

For our purposes, we define the class $\text{Mod}_q\text{W}[1]$ as the class of all parameterized problems (f, κ) with $f\colon \Sigma^* \to \{0, \ldots, q-1\}$ such that (f, κ) has a parsimonious fpt-reduction to the problem of counting k-cliques modulo q. For $q = 2$, it was shown in [3] that all problems in W[1] admit randomized fpt-reductions to problems in $\oplus\text{W}[1]$. Another result [39, Lemma 2.1] yields the corresponding generalization for all $q > 2$. We use the following analogous proposition for the vertex-colorful subgraph problem, proven in the full version.

▶ **Lemma 4.** *For any integer $q \geq 2$, there is a randomized Turing fpt-reduction from the problem* VertexColorfulSub *to the problem* #VertexColorfulSub mod q. *On input (H, G), the reduction only queries instances with the same pattern H.*

Our work relies on the following hardness result for parameterized modular subgraph counting, which follows easily from known results on the colorful subgraph decision problem [13, 31]. See the full version for a proof.

▶ **Lemma 5.** *Let $\mathcal{H}$ be a graph family of unbounded tree-width and let q be an integer with $q \geq 2$. Then* #VertexColorfulSub($\mathcal{H}$) mod q *parameterized by $k = |E(H)|$ is* $\text{Mod}_q\text{W}[1]$*-hard under parsimonious fpt-reductions. Moreover, if ETH is true, then the problem does not have an algorithm running in time $n^{o(k/\log k)}$, where $n = |V(G)|$.*

3 Hardness of counting k-matchings

In this section, we prove Theorem 1. We first establish $\text{Mod}_q\text{W}[1]$-hardness of the problem #ColMatch mod q for odd $q \geq 3$: Given a graph G with an edge-coloring $c : E(G) \to \mathcal{C}$ for some set of colors $\mathcal{C}$ with $|\mathcal{C}| = k$, this problem asks to count modulo q the edge-colorful matchings in G. These are the matchings that use each color in $\mathcal{C}$ exactly once.

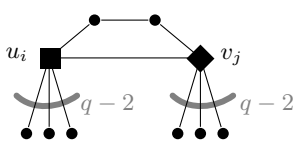
▶ **Lemma 6.** *For any fixed integer p with odd prime factor q, the problem* #ColMatch mod p *is* $\text{Mod}_q\text{W}[1]$*-hard under parsimonious fpt-reductions and has no $n^{o(k/\log k)}$ time algorithm under* ETH.

Proof. The class $\mathcal{H}$ of all 3-regular graphs has unbounded tree-width, and hence by Lemma 5, the problem #VertexColorfulSub($\mathcal{H}$) mod q is $\text{Mod}_q\text{W}[1]$-hard and hard under ETH. We reduce it to #ColMatch mod q, implying the hardness of #ColMatch mod p. Let $H \in \mathcal{H}$ and G be the input for the reduction with $k = |V(H)|$ and H-colored G, and let $\{ V_a : a \in V(H) \}$ be the color classes of G, with edge-sets $E_{a,b}(G)$ for $ab \in E(H)$. Using the gadgets from Figure 2, we construct a graph G':

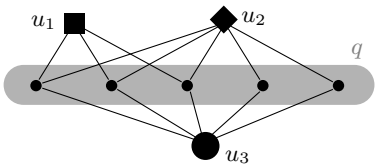
1. Each vertex $u \in V(G)$ is replaced by three vertices u_1, u_2, and u_3. We insert a *consistency gadget* Q_u at these vertices by adding q gadget vertices, connecting u_2 and u_3 to all gadget vertices, and u_1 to the first $(q+1)/2$ gadget vertices. For $S \subseteq \{u_1, u_2, u_3\}$, let m_S count the matchings in Q_u that match precisely S; it can be checked that

$$m_S \equiv_q \begin{cases} 1 & \text{if } S \text{ is } \emptyset \text{ or } \{u_1, u_2, u_3\}, \\ 0 & \text{if } S \text{ is } \{u_2\}, \{u_3\}, \text{ or } \{u_2, u_3\}. \end{cases} \tag{1}$$

   We explicitly ignore the other three cases for S, as they will not be relevant.



(a) The AND-gadget, shown here for $q = 5$. In general, the upper u_i, v_j-path always has 3 edges; both *external vertices* u_i and v_j have $q-2$ neighboring leaves. If exactly 0 or 1 external vertices are removed, this graph has 0 edges modulo q; if both vertices are removed, the graph has 1 edge.



(b) The consistency gadget contains q gadget vertices, shown here for $q = 5$. The number of matchings of size 0 and 3 equals 1 modulo q, and if u_1 is deleted, the number of non-empty edge-colorful matchings equals 0 modulo q.

Figure 2 The two gadgets used in the proof of Lemma 6.

2. For $\{a, b\} \in E(H)$, suppose that a is the jth vertex incident to b, and that b is the ith vertex incident to a. For each edge $\{u, v\} \in E_{a,b}(G)$ with $u \in V_a$ and $v \in V_b$, we insert an AND-gadget A_{uv} at $\{u_i, v_j\}$. Then for any set $S \subsetneq \{u_i, v_j\}$, the number of edges in $A_{uv} - S$ is divisible by q, whereas $A_{uv} - \{u_i, v_j\}$ has exactly one edge.
3. The edge-colors of G' are defined as follows: For $u \in V_a$ with $a \in V(H)$, we assign color $(\texttt{CONS}, a, i)$ to all edges of Q_u incident to vertex u_i, for $i \in \{1, 2, 3\}$. For each $ab \in E(H)$, we assign color $(\texttt{AND}, ab)$ to all edges in AND-gadgets between edges in $E_{a,b}(G)$. Overall, we have $k' = 3k + 3k/2$ colors.

Every H-copy F in G induces a set $\mathcal{M}_F$ of colorful matchings in G'. We describe this set in the following, show that $|\mathcal{M}_F| \equiv_q 1$, and that $\mathcal{M}_F$ and $\mathcal{M}_{F'}$ are disjoint for $F \neq F'$.

- For each $v \in V(F)$, match all of $\{v_1, v_2, v_3\}$ within Q_v. For fixed v, the number of possible matchings in Q_v is $m_{\{v_1,v_2,v_3\}} \equiv_q 1$ by (1). Let $\mathcal{Q}_F$ denote the set of all matchings that can be obtained by the previous step. Since matchings can be chosen independently for distinct Q_v, we obtain $|\mathcal{Q}_F| \equiv_q 1^{|V(F)|} \equiv_q 1$.
- Any $M \in \mathcal{Q}_F$ can be extended to several colorful matchings by choosing one edge from each color $(\texttt{AND}, ab)$ for $\{a, b\} \in E(H)$. For each $\{u, v\} \in E(F)$, the AND-gadget A_{uv} has exactly one such edge, while the other AND-gadgets of color $(\texttt{AND}, ab)$ have 0 such edges modulo q. Hence, the number edges of color $(\texttt{AND}, ab)$ that can extend M is 1 modulo q. This implies that the overall number r_M of matchings extending M into a colorful matching is also $r_M \equiv_q 1^{|E(F)|} \equiv_q 1$.

Overall, every H-copy F induces $\sum_{M \in \mathcal{Q}_F} r_M \equiv_q \sum_{M \in \mathcal{M}_F} 1 \equiv_q 1$ colorful matchings, so we indeed have $|\mathcal{M}_F| \equiv_q 1$. We also observe from the construction that $\mathcal{M}_F \cap \mathcal{M}_{F'} = \emptyset$ for distinct H-copies F and F'. In the full version, we use properties of the gadgets to prove that colorful matchings $M \notin \bigcup_F \mathcal{M}_F$ cancel modulo q.

▷ Claim 7. The number of colorful matchings M that are not contained in $\mathcal{M}_F$ for any H-copy F is divisible by q.

Overall, we have shown that the number of H-copies in G and the number of colorful matchings in G' agree modulo q. As G' can be computed in polynomial time and the parameter is increased only by a constant factor, the claimed hardness results follow. ◀

To prove Theorem 1, it suffices to give an fpt-reduction from #ColMatch mod q to counting k-matchings modulo q. This is achieved by a standard inclusion-exclusion argument that can be found in the full version.

4 Counting matching-splittable subgraphs modulo 2^t

In this section, we prove Theorem 2 by describing an $n^{O(t4^s)}$-time algorithm for counting modulo 2^t the subgraphs of matching-split number s. Our algorithm builds upon known algorithms for the decision and counting versions of subgraph problems; we first review their underlying ideas and sketch our algorithm for Theorem 2.

Counting subgraphs of bounded vertex-cover number

The basic structure of our algorithm is similar to a known $O(n^{s+1})$ time algorithm [40, 29, 9] for counting embeddings from H to G if H has a vertex-cover $S \subseteq V(H)$ of size $s \in \mathbf{N}$. Counting embeddings is sufficient for counting subgraph copies, as we can first compute the number $\#\mathrm{Aut}(H)$ of automorphisms on H as $\#\mathrm{Emb}(H, H)$, and then use

$$\#\mathrm{Sub}(H, G) = \frac{\#\mathrm{Emb}(H, G)}{\#\mathrm{Aut}(H)}\,. \tag{2}$$

Given (H, G) with $h = |V(H)|$ and $n = |V(G)|$, the algorithm for computing $\#\mathrm{Emb}(H, G)$ first finds a minimum vertex-cover S of H in time $h^{O(s)}$; then $I := V(H) \setminus S$ is an independent set. Then the algorithm enumerates all partial embeddings f from $H[S]$ to G, which takes time at most $n^{O(s)}$. Finally, for each f, it remains to count all functions $g\colon I \to V(G)$ that extend f to a full embedding from H to G. We observe that g extends f to a full embedding if and only if every vertex $u \in I$ maps via g to a vertex $v = g(u) \in V(G) \setminus f(S)$ that satisfies the *neighborhood constraint* $N_G(v) \cap f(S) \supseteq f(N_H(u))$. Counting functions g with this property can be achieved (in a not completely obvious way) with dynamic programming; we only need to know the number of vertices $v \in V(G) \setminus f(S)$ that have a specific neighborhood $N_G(v) \cap f(S)$, and for each f, there are at most 2^s different possible such neighborhoods. Overall, in $n^{O(s)}$ time, we can compute the number $\#\mathrm{Emb}(H, G)$.

Detecting subgraphs of bounded matching-split number

Jansen and Marx [25] extend the above approach and obtain an $n^{O(s)}$ time algorithm for the *decision* problem $\mathrm{Sub}(H, G)$ when H has matching-split number s. In this case, we consider a *splitting set* S of size s instead of a vertex-cover, that is, the graph $M = H - S$ may have isolated edges besides isolated vertices. Now the idea is to not only classify the vertices $v \in V(G) \setminus f(S)$ by their neighborhoods $N_v = N_G(v) \cap f(S)$, but to also classify the edges $\{u, v\} \in E(G - f(S))$ by their neighborhoods $\{N_u, N_v\}$. It then remains to find a matching in $G - f(S)$ that has as many isolated vertices and isolated edges as $H - S$, such that these vertices and edges satisfy the neighborhood constraints in $f(S)$. Jansen and Marx achieve this by reduction to a colored matching problem.

Our algorithm

In our algorithm for Theorem 2, we need to overcome two challenges:

(a) Since counting embeddings is algorithmically more straight-forward than counting subgraphs, we would like to count embeddings and divide by the number of automorphisms $\#\mathrm{Aut}(H)$ as in Equation (2). However, since we are counting modulo 2^t, the number $\#\mathrm{Aut}(H) \bmod 2^t$ may be 0, and so the division in Equation (2) is impossible. (In fact, even even numbers $\#\mathrm{Aut}(H)$ have no inverse modulo 2^t.)

(b) When mimicking Jansen and Marx's detection algorithm, we cannot just *count* the relevant matchings in $G - f(S)$, since counting perfect matchings is #P-hard.

Most of our effort focuses on overcoming (a): In Section 4.1, we show that every graph H of matching-split number s has a splitting set R of size $O(s^2)$ that remains rigid under automorphisms, i.e., any automorphism f of H must satisfy $f(R) = R$. In Section 4.2, we show how to compute $\#\mathrm{Sub}(H, G)$ if such a rigid splitting set R for H is given. Rather than counting H-embeddings and attempting a division by $\#\mathrm{Aut}(H)$, we use the rigidity of R to keep track of the automorphisms of H in a more explicit way.

To overcome (b), we use a determinant-based algorithm [23] to compute the Hafnian over a polynomial ring modulo 2^t. We then reduce our constrained matching counting problem to computing such Hafnians. This part of the algorithm can be found in the full version.

4.1 Rigidizing the splitting set

Let H be a graph with a splitting set S of size s, and let $M = H - S$ be the remaining graph of maximum degree 1; we speak of M as a matching, even though it may contain isolated vertices. An automorphism f of H may map a vertex $v \in S$ in the splitting set to $f(v) \notin S$. We show that if a splitting set of size s exists then there is also a *rigid* splitting set R of size $O(s^2)$, i.e., such that every $f \in \mathrm{Aut}(H)$ satisfies $f(R) = R$. In fact, the following algorithm can find such a set R.

Algorithm Rigidize(H) *Given a graph H of matching-split number s, this algorithm computes a rigid splitting set $R \subseteq V(H)$ of size $O(s^2)$.*

R1 (Find small splitting set.) Using brute-force, compute a set $S \subseteq V(H)$ of size s such that $H - S$ is a matching.

R2 (Extend it to neighbors of low-degree vertices.) Let $D \subseteq V(H)$ be the set of all vertices whose degree in H is at most $s + 1$. Set $T := S$. While there is an edge $\{u, v\}$ with $u \in T \cap D$ and $v \in \overline{T}$, add v to T.

R3 (Refine it.) Set $R := T$. For each component C of $H[T \cap D]$ with at most two vertices, remove $V(C)$ from R.

The following lemma captures useful properties of Rigidize. See the full version for a proof.

▶ **Lemma 8.** *The algorithm Rigidize runs in time $h^{O(s)}$ where $h = |V(H)|$, and the output set $R \subseteq V(H)$ has the properties that $|R| \leq O(s^2)$, that $H - R$ is a matching, and that every $f \in \mathrm{Aut}(H)$ satisfies $f(R) = R$.*

4.2 Counting subgraphs with rigid splitting sets

We use the rigid splitting set R from Lemma 8 to compute the number of times H occurs as a subgraph modulo a power of two. As a subroutine, we use an algorithm for counting colored matchings modulo a power of two in a setting involving particular "color demands".

▶ **Definition 9.** *Let G be a graph, let C be a finite set of colors, and let $c\colon V(G) \cup E(G) \to 2^C$ be a function that labels each vertex and edge with a subset of C. For any matching M, let $I(M)$ be the set of its isolated vertices. For a coloring $c_M : I(M) \cup E(M) \to C$, the colored matching (M, c_M) is* permissible *if $c_M(t) \in c(t)$ holds for all $t \in I(M) \cup E(M)$.*

Color demands *are functions $D_I, D_E\colon C \to \mathbf{N}$. The pair (M, c_M)* satisfies the demands D_I, D_E *if, for each $i \in C$, the graph M contains exactly $D_I(i)$ isolated vertices v with $c_M(v) = i$ and exactly $D_E(i)$ edges with $c_M(e) = i$. Let $\mathcal{M}(G, c, D_I, D_E)$ be the set of all permissible matchings (M, c_M) that satisfy the demand D.*

As shown in the full version, we obtain the following algorithm as a corollary to Hirai and Namba's algorithm [23] for computing the Hafnian over polynomial rings modulo 2^t.

▶ **Lemma 10.** *Given a graph G, permissible colors $c\colon V(G) \cup E(G) \to 2^C$, color demands $D_I, D_E\colon C \to \mathbf{N}$, and $t \in \mathbf{N}_{\geq 1}$, there is an algorithm that computes the number $|\mathcal{M}(G, c, D_I, D_E)| \bmod 2^t$ in time $n^{O(t|C|)}$.*

Before we state the main algorithm, we introduce some basic group-theoretic notation. Let R be a splitting set of H that satisfies $f(R) = R$ for all $f \in \mathrm{Aut}(H)$. Let G be a graph and let $S \subseteq V(G)$ be a set with $|S| = |R|$. For an embedding $\sigma \in \mathrm{Emb}(H[R], G[S])$ and an automorphism $\varphi \in \mathrm{Aut}(H)$, we note that the function $\sigma \circ (\varphi\restriction_R)$ is again an embedding in $\mathrm{Emb}(H[R], G[S])$. Indeed, we view this operation as a right-action of the group $\mathrm{Aut}(H)$ on the set $\mathrm{Emb}(H[R], G[S])$. We call two embeddings $\sigma, \sigma' \in \mathrm{Emb}(H[R], G[S])$ *equivalent* if there exists $\varphi \in \mathrm{Aut}(H)$ such that $\sigma' = \sigma \circ (\varphi\restriction_R)$; this clearly defines an equivalence relation. The equivalence class $\sigma\mathrm{Aut}(H)$ is called the *orbit* of σ under $\mathrm{Aut}(H)$. All orbits have the same size. Let E_S be a set of representatives for each orbit, that is, a maximal set of mutually non-equivalent embeddings in $\mathrm{Emb}(H[R], G[S])$.

We are ready to state the modular counting algorithm for s-matching-splittable subgraphs.

Algorithm ModCount(H, G, t) *Given an s-matching-splittable graph H, a host graph G, and an integer $t \geq 2$, this algorithm computes the number $\#\mathrm{Sub}(H, G) \bmod 2^t$.*

C1 (Compute rigid splitting set.) Call Rigidize(H) to compute the set R.

C2 (Reduce to counting colored matchings.) For each $S \subseteq V(G)$ with $|S| = |R|$ (that is, a possible image of R) and each representative embedding $\sigma \in E_S$ from $H[R]$ to $G[S]$, we construct an instance $(G - S, c_\sigma, D_I, D_E)$ of colored matching with demands and use Lemma 10 to obtain the number $|\mathcal{M}(G - S, c_\sigma, D_I, D_E)| \bmod 2^t$:

- **a.** (Set permitted colors.) Let $C = 2^R \cup \binom{2^R}{1} \cup \binom{2^R}{2}$. For each vertex $v \in V(G) \setminus S$, let $N_v \subseteq R$ be the vertices of R that hit $N_G(v) \cap S$ under σ, that is, $N_v = \sigma^{-1}(N_G(v) \cap S)$. Define $c_\sigma(v) = \{\, N \;:\; N \subseteq N_v \,\}$. Moreover, for each $\{u, v\} \in E(G - S)$, define $c_\sigma(\{u, v\}) = \{\, \{N, N'\} \;:\; N \subseteq N_u,\ N' \subseteq N_v \,\}$.
- **b.** (Make demands.) The demands $D_I, D_E : C \to \mathbf{N}$ depend only on H and R. For each $N \subseteq R$, we let $D_I(N)$ be the number of isolated vertices v in $H - R$ whose neighborhood in H satisfies $N_H(v) \cap R = N$. Moreover, for all $N, N' \subseteq R$, we let $D(\{N, N'\})$ be the number of edges $\{u, v\} \in E(H - R)$ with $\{N_H(u) \cap R, N_H(v) \cap R\} = \{N, N'\}$.

C3 (Sum up.) Output the sum modulo 2^t of all integers returned by the queries in C2.

In the full version, we prove that ModCount satisfies the properties stated in Theorem 2.

5 Hardness of counting paths modulo two

In this section, we prove Theorem 3, that counting k-paths modulo 2 is $\oplus$W[1]-hard. We first formally introduce this and some intermediate problems.

The *length* of a path is the number of its edges. For a graph G and vertices $s, t \in V(G)$, an *s,t-path* is a simple path from s to t. For a computable, strictly increasing function $f\colon \mathbf{N} \to \mathbf{N}$, we define f-Flexible Path to be the problem that is given (G, s, t, k) to decide whether there exists any s,t-path in G whose length ℓ satisfies $k \leq \ell \leq f(k)$. When id denotes the identity function, then Path (also known as k-Path or Longest Path) is defined as id-Flexible Path. We similarly define Directed f-Flexible Path and Directed Path for directed graphs, and we define the counting and parity versions of these problems in the canonical manner.

We start our reduction at the vertex-colorful subgraph problem $\oplus$VertexColorfulSub$(\mathcal{H})$ for a class $\mathcal{H}$ of unbounded tree-width, which is $\oplus$W[1]-hard by Lemma 5. The class we choose consists of connected, almost 4-regular graphs without non-trivial automorphisms;

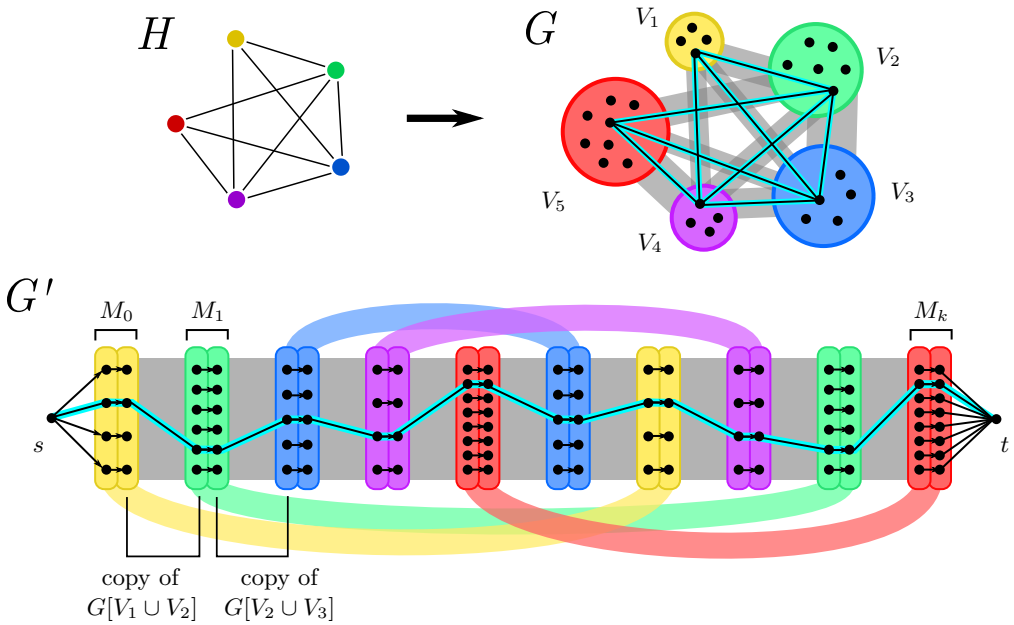


Figure 3 The construction of the graph G' from the graphs H and G. The homomorphism from G to H is indicated by colors. A colorful H-subgraph H' in G and the canonical path corresponding to H' in G' are highlighted in turquoise. Gadget edges are hinted at as semi-transparent curves. Except for (half of the) gadget edges, all edges are oriented from left to right.

here, we say that a graph is *almost* 4*-regular* if it can be obtained from a 4-regular graph by deleting one edge. The core part of the reduction is in Lemma 11, where we reduce to counting paths of somewhat flexible length in a directed graph (modulo 2). From there, we reduce to the familiar k-path problem in undirected graphs using standard tricks.

▶ **Lemma 11.** *For any class $\mathcal{H}$ of connected, almost 4-regular graphs without non-trivial automorphisms, there is a computable, strictly increasing function f such that there is parsimonious polynomial-time fpt-reduction from* $\oplus$VertexColorfulSub($\mathcal{H}$) *to* $\oplus$Directed f-Flexible Path.

Proof. Let $H \in \mathcal{H}$ and G be the undirected graphs that are given as input, where G is given with disjoint color classes V_u for $u \in V(H)$. Let $k = |E(H)|$. Since H is connected and almost 4-regular, it has an Eulerian path $u_0, u_1, \ldots, u_{k-1}, u_k$ such that $E(H) = \{\, \{u_i, u_{i+1}\} : i \in \{0, \ldots, k-1\} \,\}$. Every vertex of H appears exactly twice on the Eulerian path, and u_0 and u_k are the two different degree-3 vertices of H. Our goal is to construct a directed graph G', such that $\oplus$VertexColorfulSub$(H, G) = \oplus$Directed f-Flexible Path(G') holds for a suitable f.

Intuitively, the graph G' "visits" every color class V_{u_i} of G two times according to the Eulerian path in H. Before we give a formal construction, we give an overview; see also Figure 3. Essentially, the graph G' is a sequence of directed bipartite graphs $B_0, \ldots, B_\ell$ whose edges are all directed from left to right. For a bipartite graph B, we write $L(B)$ and $R(B)$ for its left and right part, respectively. We have $R(B_j) = L(B_{j+1})$ for all $j \in \{0, \ldots, \ell-1\}$. Each B_j is either a perfect matching M_i or a graph G_i that is a directed copy of $G[V_{u_{i-1}} \cup V_{u_i}]$. (Note that G_i is indeed bipartite, since G is H-colored and H contains no self-loops.) Pictorially, the sequence of bipartite graphs is $M_0 G_1 M_1 \ldots M_{k-1} G_k M_k$. We also add some additional *gadget edges* between all M_i and M_j with $i \neq j$ and $u_i = u_j$; note that every M_i is

paired with exactly one M_j in this way, because the Eulerian path visits every vertex exactly twice. The gadget edges are the only edges that may be directed from right to left and that connect non-adjacent layers.

The paths p in G that we wish to count modulo two should correspond to the colourful H-subgraphs of G. The path p is supposed to run from left to right through G'; intuitively, the edge that the path picks at layer M_i corresponds to the vertex of $V_{u_i} \subseteq V(G)$ that $u_i \in V(H)$ is mapped to in the subgraph embedding, and the edge that the path picks at layer G_i corresponds to the edge of G that $u_{i-1}u_i \in E(H)$ is mapped to in the subgraph embedding. The gadget edges ensure that those paths cancel modulo two that do not consistently select the "same" vertex in V_{u_i} and V_{u_j} when $u_i = u_j$.

We now describe the construction of G' in detail.

1. **Graph edges.** For each $i \in \{1, \dots, k\}$, let G_i be a fresh copy of $G[V_{u_{i-1}} \cup V_{u_i}]$, renamed so that $L(G_i) = \{i\} \times V_{u_{i-1}}$ and $R(G_i) = \{i\} \times V_{u_i}$, and directed from left to right.
2. **Matching edges.** For each $i \in \{0, \dots, k\}$, let M_i be the canonical perfect matching between $L(M_i) = \{i\} \times V_{u_i}$ and $R(M_i) = \{i+1\} \times V_{u_i}$, and directed from left to right. Note that $L(M_i) = R(G_i)$ holds for $i \in \{1, \dots, k\}$ and $R(M_i) = L(G_{i+1})$ holds for $i \in \{0, \dots, k-1\}$.
3. **Gadget edges.** For all $i, j \in \{1, \dots, k\}$ with $i < j$ and $u_i = u_j$, note that $L(M_i) = \{i\} \times V_{u_i}$ and $L(M_j) = \{j\} \times V_{u_i}$. We add the canonical *bidirected* perfect matching between $L(M_i)$ and $L(M_j)$. Similarly, we add the canonical bidirected perfect matching between $R(M_i)$ and $R(M_j)$.
4. **Source/sink.** Let s be a new vertex and add all edges (s, v) for $v \in L(M_0)$. Let t be a new vertex and add all edges (v, t) for $v \in R(M_{k+1})$.
5. **Parameters.** Finally, we set $k' = 2k + 2$ and $f(k') = 6k + 3$ so that we are counting all s, t-paths whose length is between k' and $f(k')$.

This finishes the construction of G'. Note that $G' \setminus \{s, t\}$ is indeed a sequence $B_0 \dots B_\ell$ of bipartite graphs with some additional gadget edges, where $\ell = 2k$. We define the *j-th layer of G'* as the set $L_j = L(B_j)$ for $j \leq \ell$ and $L_{\ell+1} = R(B_\ell)$. Recall that $L_j = R(B_{j-1})$ holds for $j > 0$.

We describe the *canonical* solutions in the output of the reduction.

To this end, let H' be an H-subgraph of G that is colorful. This means that H is isomorphic to H' and that the "coloring" homomorphism $c\colon V(G) \to V(H)$ is bijective on $V(H')$. Moreover, c restricted to $V(H')$ is in fact an isomorphism: It must map non-edges to non-edges because $E(H)$ and $E(H')$ have the same size. Let $\phi\colon V(H) \to V(H')$ be the inverse of c restricted to $V(H')$ and note that ϕ is an isomorphism from H to H'. Moreover, because H has non-trivial automorphisms, this isomorphism ϕ is unique for H'.

We define the *canonical s, t-path spt in G' corresponding to H'*: The path p visits exactly one vertex from each layer from left to right; each layer has the form $L_j = \{i\} \times V_u$ for some i and u, and p chooses the vertex $(i, \phi(u)) \in L_j$ in this layer. Note that this determines all vertices of p. We claim that $V(p)$ indeed induces a path on the graph and matching edges.

To show that p is a path, let $j \in \{0, \dots, \ell\}$. We claim that the two vertices in $V(p) \cap V(B_j)$ are adjacent in B_j. If B_j is one of the matching graphs, then $L(B_j) = \{i\} \times V_{u_i}$ and $R(B_j) = \{i+1\} \times V_{u_i}$ for some i. Since the perfect matching is canonical, there is indeed an edge from $(i, \phi(u_i))$ to $(i+1, \phi(u_i))$ in B_j. Otherwise, B_j is one of the graph copies, say $L(B_j) = \{i\} \times V_{u_{i-1}}$ and $R(B_j) = \{i\} \times V_{u_i}$. Recall that $u_{i-1}u_i$ is part of the Eulerian path and thus an edge of H. Since ϕ is a graph homomorphism from H to G respecting the coloring, we have that $\phi(u_{i-1})\phi(u_i)$ is an edge in $G[V_{u_{i-1}} \cup V_{u_i}]$. Since B_j was a copy of this graph by construction, there is an edge from $(i, \phi(u_{i-1}))$ to $(i, \phi(u_i))$. Overall, we get that spt is an s, t-path in G', and its length is $\ell + 2 = 2k + 2 = k'$; this is the canonical path corresponding to H'.

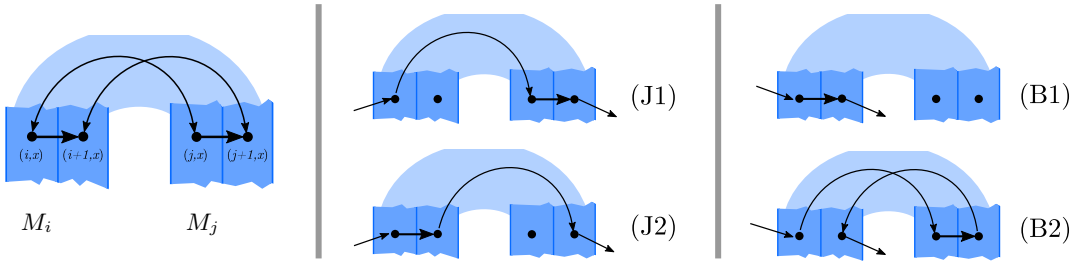


Figure 4 The left drawing shows the local configuration around the four vertices of G' that represent a vertex $x \in V_u$ from G. These four vertices are contained in two matchings M_i and M_j for $u_i = u_j = u$. Thick edges are contained in M_i or M_j while light edges are gadget edges. Note that (i, x) and (j, x) have only the depicted outgoing edges in the entire graph G', and $(i+1, x)$ and $(j+1, x)$ have only the depicted incoming edges. The four drawings on the right depict (up to the symmetry of exchanging i and j) all different ways in which paths might not be canonical: Either they illegally use the gadget edges to jump from M_i to M_j as in (J1) or (J2) and continue from there, or they do not consistently visit the corresponding edges in M_i and M_j, which allows them to use the gadget edges to either take (B2) or not take (B1) a short detour from M_i to M_j.

In summary, every vertex-colorful H-subgraph H' in G defines a unique canonical s, t-path in G', which implies that the number of canonical paths is equal to the number of H-subgraphs. We now characterize canonical paths slightly differently: Let p be any s, t-path in G' that picks exactly one vertex of each layer from left to right with the additional property that it consistently picks the "same" vertex from each color class. That is, whenever p picks (i, x) in layer $\{i\} \times V_u$ and (j, y) in a layer $\{j\} \times V_u$ (with the same V_u), then $x = y$. Such a path p describes a set of $|V(H)|$ vertices and k edges in G that make up a colorful H-subgraph H' of G, which means that every such path is canonical.

Let $\mathcal{P}$ be the set of all s, t-paths whose length r satisfies $k' \leq r \leq f(k')$. The central claim is that the number of non-canonical paths in $\mathcal{P}$ is even. For this, we construct a fixed-point free involution π on non-canonical paths.

First we focus on paths that are *jumpy* (cf. Figure 4): Let $i, j \in \{0, \ldots, k\}$ with $i \neq j$ and $u_i = u_j$. By construction, we added gadget edges between M_i and M_j. Recall that vertices have the form

$$(i, x) \in L(M_i),\ (i+1, x) \in R(M_i), \qquad (j, x) \in L(M_j),\ (j+1, x) \in R(M_j).$$

A path $p \in \mathcal{P}$ is jumpy at i, j, x if
(J1) p uses the edge from (i, x) to (j, x) but not the edge from $(j+1, x)$ to $(i+1, x)$, or
(J2) p uses the edge from $(i+1, x)$ to $(j+1, x)$ but not the edge from (j, x) to (i, x).

If p is jumpy (for some choice of i, j, x), we define $\pi(p)$ as follows: First we identify the lexicographically first position i, j, x where p is jumpy. Then we exchange state (J1) with state (J2) at that position. Note that (J1) implies that p uses the M_j-edge from (j, x) to $(j+1, x)$, since (j, x) has no other outgoing edges, and (J2) implies that p uses the M_i-edge from (i, x) to $(i+1, x)$, because $(i+1, x)$ has no other incoming edges; we swap these edges from M_i and M_j too when applying π. Now π is a fixed-point free involution on jumpy paths, and note that $\pi(p)$ has the same length as p.

Since jumpy paths will cancel out when counting modulo two, we can focus on non-canonical paths that are not jumpy. Paths p that are not jumpy have the following property: A gadget edge from (i, x) on the left side of M_i to (j, x) on the left side of M_j is used by p if and only if the corresponding edge from $(j+1, x)$ to $(i+1, x)$ on the right side is used.

Next we consider paths that are *bad* (again, cf. Figure 4): A path $p \in \mathcal{P}$ is bad at i, j, x with $u_i = u_j$ if

(B1) p uses the M_i-edge from (i, x) to $(i+1, x)$ but not the M_j-edge from (j, x) to $(j+1, x)$, or

(B2) p does not use the M_i-edge from (i, x) to $(i+1, x)$ but does use the M_j-edge from (j, x) to $(j+1, x)$.

We define $\pi(p)$ for a bad path p by finding the first position i, j, x at which p is bad, and switching between these two states. Say, p was in state (B1) at i, j, x as depicted in the figure, then $\pi(p)$ is in state (B2) at i, j, x, and $\pi(p)$ is exactly two edges longer than p.

We claim that all bad paths that are not jumpy pair up in this manner, without having to consider arbitrarily long paths. Indeed, since p is not jumpy, when we look at the vertices that p traverses, we are for the most part traversing the layers in a monotone order, except for potential short two-vertex detours in bad positions as depicted in the figure as (B2). More precisely, for every vertex v on p, if $v \in L_j$ for $j < \ell$, then either the next vertex is in L_{j+1} or the third vertex after it is in L_{j+1}. This means that the path moves to the right by one layer at least once every 3 vertices, and thus paths that are not jumpy have length at most $3\ell + 3 = 6k + 3 = f(k')$, accounting for $\ell + 1$ matching or graph edges and up to 2ℓ gadget edges that p might take in the short detours, and the two edges at the source and sink.

Finally, if an s, t-path is neither jumpy nor bad, then it does not use any gadget edges, and thus is canonical. Since all jumpy or bad paths cancel, the number of s, t-paths of length between k' and $f(k')$ in G' is the number of canonical paths modulo two. ◀

For completeness, we include two simple reductions: First from the flexible-length to the fixed-length problem in directed graphs, then from the directed to undirected problem.

▶ **Lemma 12.** *Let $f : \mathbf{N} \to \mathbf{N}$ be computable and strictly increasing. There is a parsimonious polynomial-time fpt-reduction from* #Directed f-Flexible Paths *to* #Directed Paths*.*

▶ **Lemma 13.** #Directed Paths *admits a parsimonious poly-time fpt-reduction to* #Paths*.*

With all these prerequisites collected, we can complete the proof.

Proof of Theorem 3. Let $\mathcal{H}$ be the class of all graphs H that are connected, almost 4-regular, and whose automorphism group has size one. We use the probabilistic method to argue that the tree-width of graphs in $\mathcal{H}$ is not bounded. With probability $1 - o(1)$ as $h \to \infty$, random 4-regular graphs with h vertices are connected [41, Theorem 2.10], they have no nontrivial automorphisms [27], and they are almost Ramanujan [24, Theorem 7.10], that is, their second-largest eigenvalue in absolute value satisfies $\lambda \leq 2\sqrt{3} + o(1) < 3.5$. By a union bound, H has all three properties simultaneously with probability $1 - o(1)$. By Cheeger's inequality [24, Theorem 4.11], we have

$$\min_{S \subseteq V(H), |S| \leq \frac{1}{2}h} \frac{|E(S, \overline{S})|}{|S|} \geq \frac{1}{2}(4 - \lambda) > 0.1\,,$$

that is, the edge expansion is bounded away from zero, which implies that the tree-width of H is at least linear in h (see, e.g., [12, Exercise 7.34]). Now, if we remove an arbitrary edge e from H, we obtain an almost 4-regular graph that remains connected (since H is Eulerian) and whose tree-width has decreased by at most 1. Moreover, suppose that π is an automorphism of $H - e$. Since π preserves degrees, it has to map the vertex set e to e. But then π is an automorphism of H, too, which implies that π is the trivial automorphism and the automorphism group of $H - e$ has size one as required. Thus, $\mathcal{H}$ contains graphs of arbitrarily large tree-width.

By Lemma 5, $\oplus$VertexColorfulSub($\mathcal{H}$) is $\oplus$W[1]-hard under parsimonious fpt-reductions. If there is a parsimonious fpt-reduction from problem $\#A$ to problem $\#B$ then in particular the parity version $\oplus A$ reduces to $\oplus B$. Writing $\oplus A \leq \oplus B$ we can summarize the chain of reductions in Lemmas 11–13 as

$$\oplus\text{VertexColorfulSub}(\mathcal{H}) \leq \oplus\text{Directed } f\text{-Flexible Paths} \leq \oplus\text{Directed Paths} \leq \oplus\text{Paths}\,.$$

This proves the $\oplus$W[1]-hardness of $\oplus$Paths. The containment follows from the standard fpt-reduction from #Paths to #Clique, which is parsimonious. Overall, the claim follows. ◀

6 Conclusion

We conducted an initial investigation of modular subgraph counting, leading to the partial classification depicted in Figure 1. To obtain a complete picture, the following conjecture needs to be addressed.

▶ **Conjecture 14.** *For any computable pattern class $\mathcal{H}$:*

- *If $\mathcal{H}$ has unbounded matching-split number, then the problem $\oplus\text{Sub}(\mathcal{H})$ is $\oplus$W[1]-complete.*
- *If $\mathcal{H}$ has unbounded vertex-cover number, then $\#\text{Sub}(\mathcal{H}) \bmod q$ for fixed $q \in \mathbf{N}$ is Mod_pW[1]-complete for any odd divisor p of q.*

An appropriate transfer of the subgraph-homomorphism framework to modular counting is likely to help in settling this conjecture. Partial results towards this have been obtained by Peyerimhoff et al. [33].

References

1. Noga Alon, Raphael Yuster, and Uri Zwick. Finding and counting given length cycles. *Algorithmica*, 17(3):209–223, 1997. doi:10.1007/BF02523189.
2. Vikraman Arvind and Venkatesh Raman. Approximation algorithms for some parameterized counting problems. In Prosenjit Bose and Pat Morin, editors, *Algorithms and Computation, 13th International Symposium, ISAAC 2002 Vancouver, BC, Canada, November 21-23, 2002, Proceedings*, volume 2518 of *Lecture Notes in Computer Science*, pages 453–464. Springer, 2002. doi:10.1007/3-540-36136-7_40.
3. Andreas Björklund, Holger Dell, and Thore Husfeldt. The parity of set systems under random restrictions with applications to exponential time problems. In Magnús M. Halldórsson, Kazuo Iwama, Naoki Kobayashi, and Bettina Speckmann, editors, *Automata, Languages, and Programming - 42nd International Colloquium, ICALP 2015, Kyoto, Japan, July 6-10, 2015, Proceedings, Part I*, volume 9134 of *Lecture Notes in Computer Science*, pages 231–242. Springer, 2015. doi:10.1007/978-3-662-47672-7_19.
4. Andreas Björklund and Thore Husfeldt. The parity of directed Hamiltonian cycles. In *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013, 26-29 October, 2013, Berkeley, CA, USA*, pages 727–735. IEEE Computer Society, 2013. doi:10.1109/FOCS.2013.83.
5. Hans L. Bodlaender. On linear time minor tests with depth-first search. *Journal of Algorithms*, 14(1):1–23, January 1993. doi:10.1006/jagm.1993.1001.
6. Yijia Chen, Martin Grohe, and Bingkai Lin. The hardness of embedding grids and walls. In Hans L. Bodlaender and Gerhard J. Woeginger, editors, *Graph-Theoretic Concepts in Computer Science - 43rd International Workshop, WG 2017, Eindhoven, The Netherlands, June 21-23, 2017, Revised Selected Papers*, volume 10520 of *Lecture Notes in Computer Science*, pages 180–192. Springer, 2017. doi:10.1007/978-3-319-68705-6_14.

7 Radu Curticapean. *The simple, little and slow things count: on parameterized counting complexity*. PhD thesis, Saarland University, 2015. URL: http://scidok.sulb.uni-saarland.de/volltexte/2015/6217/.

8 Radu Curticapean, Holger Dell, and Dániel Marx. Homomorphisms are a good basis for counting small subgraphs. In Hamed Hatami, Pierre McKenzie, and Valerie King, editors, *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 210–223. ACM, 2017. doi:10.1145/3055399.3055502.

9 Radu Curticapean and Dániel Marx. Complexity of counting subgraphs: Only the boundedness of the vertex-cover number counts. In *55th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2014, Philadelphia, PA, USA, October 18-21, 2014*, pages 130–139. IEEE Computer Society, 2014. doi:10.1109/FOCS.2014.22.

10 Radu Curticapean and Dániel Marx. Complexity of counting subgraphs: Only the boundedness of the vertex-cover number counts. *CoRR*, abs/1407.2929, 2014. arXiv:1407.2929.

11 Radu Curticapean and Mingji Xia. Parameterizing the permanent: Genus, apices, minors, evaluation mod 2k. In Venkatesan Guruswami, editor, *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*, pages 994–1009. IEEE Computer Society, 2015. doi:10.1109/FOCS.2015.65.

12 Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015. doi:10.1007/978-3-319-21275-3.

13 Víctor Dalmau and Peter Jonsson. The complexity of counting homomorphisms seen from the other side. *Theor. Comput. Sci.*, 329(1-3):315–323, 2004. doi:10.1016/j.tcs.2004.08.008.

14 Julian Dörfler, Marc Roth, Johannes Schmitt, and Philip Wellnitz. Counting induced subgraphs: An algebraic approach to #W[1]-hardness. In Peter Rossmanith, Pinar Heggernes, and Joost-Pieter Katoen, editors, *44th International Symposium on Mathematical Foundations of Computer Science, MFCS 2019, August 26-30, 2019, Aachen, Germany*, volume 138 of *LIPIcs*, pages 26:1–26:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPIcs.MFCS.2019.26.

15 John D. Faben. The complexity of counting solutions to generalised satisfiability problems modulo k. *CoRR*, abs/0809.1836, 2008. arXiv:0809.1836.

16 John D. Faben and Mark Jerrum. The complexity of parity graph homomorphism: An initial investigation. *Theory Comput.*, 11:35–57, 2015. doi:10.4086/toc.2015.v011a002.

17 Jörg Flum and Martin Grohe. The parameterized complexity of counting problems. *SIAM J. Comput.*, 33(4):892–922, 2004. doi:10.1137/S0097539703427203.

18 Jacob Focke, Leslie Ann Goldberg, Marc Roth, and Stanislav Živný. Counting homomorphisms to K4-minor-free graphs, modulo 2. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2303–2314. Society for Industrial and Applied Mathematics, January 2021. doi:10.1137/1.9781611976465.137.

19 Fedor V. Fomin, Daniel Lokshtanov, Venkatesh Raman, Saket Saurabh, and B. V. Raghavendra Rao. Faster algorithms for finding and counting subgraphs. *J. Comput. Syst. Sci.*, 78(3):698–706, 2012. doi:10.1016/j.jcss.2011.10.001.

20 Andreas Göbel, Leslie Ann Goldberg, and David Richerby. The complexity of counting homomorphisms to cactus graphs modulo 2. *ACM Trans. Comput. Theory*, 6(4):17:1–17:29, 2014. doi:10.1145/2635825.

21 Andreas Göbel, Leslie Ann Goldberg, and David Richerby. Counting homomorphisms to square-free graphs, modulo 2. *ACM Trans. Comput. Theory*, 8(3):12:1–12:29, 2016. doi:10.1145/2898441.

22 Heng Guo, Sangxia Huang, Pinyan Lu, and Mingji Xia. The complexity of weighted Boolean #CSP modulo k. In Thomas Schwentick and Christoph Dürr, editors, *28th International Symposium on Theoretical Aspects of Computer Science, STACS 2011, March 10-12, 2011,*

Dortmund, Germany, volume 9 of *LIPIcs*, pages 249–260. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2011. doi:10.4230/LIPIcs.STACS.2011.249.

23 Hiroshi Hirai and Hiroyuki Namba. Shortest (a+b)-path packing via Hafnian. *Algorithmica*, 80(8):2478–2491, 2018. doi:10.1007/s00453-017-0334-0.

24 Shlomo Hoory, Nathan Linial, and Avi Wigderson. Expander graphs and their applications. *Bulletin of the American Mathematical Society*, 43(04):439–562, August 2006. doi:10.1090/s0273-0979-06-01126-8.

25 Bart M. P. Jansen and Dániel Marx. Characterizing the easy-to-find subgraphs from the viewpoint of polynomial-time algorithms, kernels, and Turing kernels. In Piotr Indyk, editor, *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, San Diego, CA, USA, January 4-6, 2015*, pages 616–629. SIAM, 2015. doi:10.1137/1.9781611973730.42.

26 Amirhossein Kazeminia and Andrei A. Bulatov. Counting homomorphisms modulo a prime number. In Peter Rossmanith, Pinar Heggernes, and Joost-Pieter Katoen, editors, *44th International Symposium on Mathematical Foundations of Computer Science, MFCS 2019, August 26-30, 2019, Aachen, Germany*, volume 138 of *LIPIcs*, pages 59:1–59:13. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPIcs.MFCS.2019.59.

27 Jeong Han Kim, Benny Sudakov, and Van H. Vu. On the asymmetry of random regular graphs and random graphs. *Random Struct. Algorithms*, 21(3-4):216–224, 2002. doi:10.1002/rsa.10054.

28 Ioannis Koutis. Faster algebraic algorithms for path and packing problems. In Luca Aceto, Ivan Damgård, Leslie Ann Goldberg, Magnús M. Halldórsson, Anna Ingólfsdóttir, and Igor Walukiewicz, editors, *Automata, Languages and Programming, 35th International Colloquium, ICALP 2008, Reykjavik, Iceland, July 7-11, 2008, Proceedings, Part I: Tack A: Algorithms, Automata, Complexity, and Games*, volume 5125 of *Lecture Notes in Computer Science*, pages 575–586. Springer, 2008. doi:10.1007/978-3-540-70575-8_47.

29 Miroslaw Kowaluk, Andrzej Lingas, and Eva-Marta Lundell. Counting and detecting small subgraphs via equations. *SIAM J. Discret. Math.*, 27(2):892–909, 2013. doi:10.1137/110859798.

30 Bingkai Lin. The parameterized complexity of the k-biclique problem. *J. ACM*, 65(5):34:1–34:23, 2018. doi:10.1145/3212622.

31 Dániel Marx. Can you beat treewidth? *Theory Comput.*, 6(1):85–112, 2010. doi:10.4086/toc.2010.v006a005.

32 Burkhard Monien. How to find long paths efficiently. In *Analysis and Design of Algorithms for Combinatorial Problems*, pages 239–254. Elsevier, 1985. doi:10.1016/s0304-0208(08)73110-4.

33 Norbert Peyerimhoff, Marc Roth, Johannes Schmitt, Jakob Stix, and Alina Vdovina. Parameterized (modular) counting and Cayley graph expanders. *CoRR*, abs/2104.14596, 2021. arXiv:2104.14596.

34 Jürgen Plehn and Bernd Voigt. Finding minimally weighted subgraphs. In Rolf H. Möhring, editor, *Graph-Theoretic Concepts in Computer Science, 16rd International Workshop, WG '90, Berlin, Germany, June 20-22, 1990, Proceedings*, volume 484 of *Lecture Notes in Computer Science*, pages 18–29. Springer, 1990. doi:10.1007/3-540-53832-1_28.

35 Marc Roth. Counting restricted homomorphisms via Möbius inversion over matroid lattices. In Kirk Pruhs and Christian Sohler, editors, *25th Annual European Symposium on Algorithms, ESA 2017, September 4-6, 2017, Vienna, Austria*, volume 87 of *LIPIcs*, pages 63:1–63:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017. doi:10.4230/LIPIcs.ESA.2017.63.

36 Marc Roth and Johannes Schmitt. Counting induced subgraphs: A topological approach to #W[1]-hardness. *Algorithmica*, 82(8):2267–2291, 2020. doi:10.1007/s00453-020-00676-9.

37 Marc Roth, Johannes Schmitt, and Philip Wellnitz. Counting small induced subgraphs satisfying monotone properties. In *61st IEEE Annual Symposium on Foundations of Computer*

Science, FOCS 2020, Durham, NC, USA, November 16-19, 2020, pages 1356–1367. IEEE, 2020. doi:10.1109/FOCS46700.2020.00128.

38 Seinosuke Toda. PP is as hard as the polynomial-time hierarchy. *SIAM J. Comput.*, 20(5):865–877, 1991. doi:10.1137/0220053.

39 Virginia Vassilevska Williams, Joshua R. Wang, Richard Ryan Williams, and Huacheng Yu. Finding four-node subgraphs in triangle time. In Piotr Indyk, editor, *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, San Diego, CA, USA, January 4-6, 2015*, pages 1671–1680. SIAM, 2015. doi:10.1137/1.9781611973730.111.

40 Virginia Vassilevska Williams and Ryan Williams. Finding, minimizing, and counting weighted subgraphs. *SIAM J. Comput.*, 42(3):831–854, 2013. doi:10.1137/09076619X.

41 Nicholas C. Wormald. Models of random regular graphs. *London Mathematical Society Lecture Note Series*, pages 239–298, 1999.