Counting Problems in Parameterized Complexity

Radu Curticapean
Basic Algorithms Research Copenhagen (BARC) and IT University of Copenhagen, Copenhagen, Denmark
radu.curticapean@gmail.com
https://orcid.org/0000-0001-7201-9905

Abstract

This survey is an invitation to parameterized counting problems for readers with a background in parameterized algorithms and complexity. After an introduction to the peculiarities of counting complexity, we survey the parameterized approach to counting problems, with a focus on two topics of recent interest: Counting small patterns in large graphs, and counting perfect matchings and Hamiltonian cycles in well-structured graphs.

While this survey presupposes familiarity with parameterized algorithms and complexity, we aim at explaining all relevant notions from counting complexity in a self-contained way.

2012 ACM Subject Classification Theory of computation → Parameterized complexity and exact algorithms, Theory of computation → Problems, reductions and completeness

Keywords and phrases counting complexity, parameterized complexity, graph motifs, perfect matchings, graph minor theory, Hamiltonian cycles

Digital Object Identifier 10.4230/LIPIcs.IPEC.2018.1

Category Invited Tutorial

Funding BARC is supported by Villum Foundation Grant No. 16582.

Acknowledgements I wish to thank all my colleagues for the discussions on this topic and the collaborations over the last years.

1 Introduction

Many problems in computer science ask about the *existence* of solutions, such as satisfying assignments or graph structures with certain properties. However, in both practice and theory, it may be equally important to *count* solutions.

In network science, counting problems occur in the context of *graph motifs* [79], which are small patterns in graphs that occur unexpectedly often. For instance, social networks exhibit lower numbers of induced 3-vertex paths than one would expect in comparable random networks. This is a consequence of the *triadic closure* [56], the fact that two people with a common friend are likely to be friends themselves. Such phenomena cannot be observed by merely testing existence of patterns.

In statistical physics, thermodynamic properties of discrete systems are determined by *partition functions* [59]; such functions essentially count the admissible states of systems. As an example, the *dimer model* is a graph-based system that models how atoms can pair up to two-atom molecules [90, 65, 66]. In the 1960s, while attempting to find formulas for the partition function of the dimer model, physicists invented a polynomial-time algorithm for counting perfect matchings in planar graphs.

Counting can also help for problems that do not explicitly ask about numbers: Some of the best known algorithms for Hamiltonicity or coloring problems rely on intermediate counting problems [57, 8, 7].

1.1 Counting complexity

Most interesting counting problems are not known to admit polynomial-time algorithms. Sometimes, this can be explained through NP-hardness: For instance, counting the satisfying assignments of Boolean formulas is clearly NP-hard, since counting is certainly not easier than deciding existence. Such arguments however do *not* show hardness for counting problems that admit polynomial-time solvable decision versions, such as counting perfect matchings in graphs, even though these problems are presumed to be hard.

This called for a complexity theory that is specific to counting problems—such a theory was provided in 1979 by Valiant [93] in a seminal paper that introduced the complexity class #P, which captures most natural counting problems. The problems in #P are the counting versions of problems in NP: Rather than deciding the existence of a witness on a given input, the task is to count the witnesses. Counting problems are #P-hard if they admit polynomial-time Turing reductions from counting satisfying assignments of CNF formulas. As desired, this includes natural counting problems with easy decision versions, such as the problem of counting perfect matchings, as shown in Valiant's paper [93]. The #P-hardness proof for this problem required the introduction of several new reduction techniques, which we survey in Section 1.3, and showed that counting complexity amounts to more than just checking that NP-hardness reductions carry over to the counting setting.

Since the initial #P-hardness results, the study of counting problems advanced to a rich sub-area of complexity theory. In particular, large territories of counting problems were classified successfully by means of *dichotomy theorems* that sort into polynomial-time solvable and #P-hard problems [46, 16, 45, 21, 19]. For instance, a dichotomy theorem for the counting versions of *constraint satisfaction problems* was first shown a decade ago [17, 47], while the analogous result for the decision version was a long-standing open problem that was resolved only last year [18, 99]. The known dichotomy theorems for counting problems show that most interesting problems are indeed #P-hard, with only few polynomial-time solvable exceptions, such as counting perfect matchings in planar graphs [90, 65, 66], counting spanning trees [67], and problems solved by holographic algorithms [94, 20].

Thus, even more so than for NP-hard decision problems, relaxations were needed to cope with #P-hard counting problems. The most popular such coping technique is *approximate counting*, which allows outputs within a multiplicative error of the actual count [38, 24, 54, 55, 64]. Important #P-hard counting problems admit randomized polynomial-time approximation schemes; these famously include counting perfect matchings in bipartite graphs [60]. (The same result for *general* graphs remains a major open problem in approximate counting.) Counting problems were also investigated through *parameterized*, *exponential-time*, and *fine-grained* complexity [48, 77, 40]. In this survey, we focus on these paradigms.

1.2 Parameterized counting complexity

The multivariate paradigm has proven to be very successful for decision and optimization problems [44, 49, 82, 36], and it also enables a deeper understanding of counting problems. One of the first results on parameterized counting actually lies in the three-fold intersection of parameterization, approximation, and counting: Arvind and Raman [6] showed in 2002 how to approximately count H-subgraph copies in a graph G to within a multiplicative error of $1 \pm \epsilon$ in randomized time $f(k, \epsilon) \cdot n^{\mathbf{tw}(H)+O(1)}$. This gives an FPT-algorithm for approximately counting k-paths, k-cycles, and k-matchings.

However, no FPT-algorithm is known for counting such subgraph patterns *exactly*, creating the need for a parameterized analogue of #P: While the problem of counting k-cliques (even approximately) is W[1]-hard through its decision version, we cannot say the same about counting k-paths, k-cycles, or k-matchings, as the decision versions of these problems are FPT [4]. To deal with such issues, Flum and Grohe [48] and independently McCartin [77] introduced the class #W[1], a natural parameterized version of #P: In analogy to W[1], the canonical #W[1]-hard problem is that of counting k-cliques in a graph, and a problem is #W[1]-hard if it admits a parameterized Turing reduction from counting k-cliques.

Apart from introducing the class #W[1], Flum and Grohe also proved an impressive parameterized reduction from counting k-cliques to counting k-cycles and k-paths, thus proving the #W[1]-hardness of these problems and paving the way for a rich complexity theory of parameterized counting problems. Techniques in parameterized counting were later used to classify the complexity of general subgraph counting problems parameterized by pattern size [10, 27, 33, 78, 63, 61, 62, 15, 85, 86, 31], to obtain FPT-approximation results for #P-hard problems [62, 63, 78, 1, 13], and to investigate classical #P-hard problems like counting perfect matchings under structural parameters [28, 35, 32, 34].

As for decision and optimization problems, more precise running time bounds for parameterized counting problems can be obtained through the exponential-time hypothesis ETH [58]. This hypothesis also admits a counting version #ETH, introduced by Dell et al. [40], which asserts that counting satisfying assignments to Boolean 3-CNF-formulas requires time $2^{\Omega(n)}$. A sparsification lemma [40] and counting-specific reduction techniques [29] are known for #ETH, enabling tight lower bounds for a variety of problems [14, 29]. Note that #ETH may be a weaker assumption than ETH.

1.3 Techniques for counting problems

To gain some intuition for the algorithmic boundary of parameterized counting, let us examine the fates of some techniques for parameterized algorithms as we go from decision to counting:

- To start on a positive note, *dynamic programming* often carries over to counting problems. For instance, in treewidth-based DPs, *join* nodes correspond very roughly to multiplication (more realistically, *convolution* [51]) and *forget* nodes correspond roughly to summation over the possible states of the vertex to be forgotten. Most importantly, Courcelle's theorem admits analogous counting versions [5, 26, 74]. There also is an FPT-algorithm for counting the models of FOL formulas on structures of locally bounded treewidth [52].
- On the other hand, *win-win* approaches like *bidimensionality* [42, 50] fail for counting problems when the large-treewidth case "wins" via a guaranteed solution: As an example, knowing that k-paths exist due to the presence of a $\sqrt{k} \times \sqrt{k}$ grid minor does a priori not facilitate counting. However, if the large-treewidth case guarantees the *absence* of the solutions sought for, then we only need to address the low-treewidth case: As mentioned above, treewidth-based DPs for decision problems often support the extension to counting.
- *Color coding* [4] fails, strictly speaking: While every k-set will be colorful under at least one coloring, there is otherwise no guarantee on the number of such colorings. In fact, it is known [2] that a perfectly balanced collection of colorings needs size $\Omega(n^{k/2})$. There are however approximately balanced collections that enable approximate counting [1, 3].
- Most *kernelization* techniques preserve only the existence of solutions. However, under proper definitions of a counting kernel, some techniques like the sunflower kernel for bounded-cardinality k-hitting sets do carry over [91]. In Section 3.1, we will also see an application of protrusion replacement [12] that works for counting.

What we lose in algorithmic techniques when going into the counting world, we gain in new reduction techniques for proving hardness. In the following, we survey some reduction techniques for #P- and #W[1]-hardness that are unavailable for NP- and W[1]-hardness.

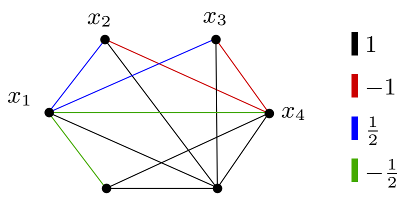
Cancellations in gadgets

Gadgets in NP- or W[1]-hardness proofs typically serve the purpose of testing local states: They can be extended by an internal solution iff their external state is *admissible* in a problem-specific sense. In counting problems with easy decision versions, gadgets with this property may not necessarily exist, as they might imply hardness of the decision version. We must actually use the power of counting: As first done by Valiant [93], we can design gadgets that do admit unwanted internal extensions for non-admissible external states, but such that the (weighted) contributions of unwanted internal extensions *cancel out to zero*.

As an example, let us consider a weighted version of counting perfect matchings: Given an edge-weighted graph $G = (V, E, w)$, we define

$$\mathrm{PerfMatch}(G) = \sum_{M} \prod_{e \in M} w(e),$$

where M ranges over the perfect matchings of G. The following edge-weighted gadget Γ with special vertices $X = \{x_1, \ldots, x_4\}$ acts as an "equality gadget" for PerfMatch: For $S = \emptyset$ and $S = X$, we have $\mathrm{PerfMatch}(\Gamma - S) = 1$, and we also have $\mathrm{PerfMatch}(\Gamma - S) = 0$ for all other $S \subseteq X$. (The gadget is from [34] and was found with a computer algebra system.)



Summarizing, if the gadget Γ is used in a larger construction, then *effectively* either all or none of X must be matched by edges outside of Γ. Even though Γ does admit internal matchings in some of the other cases, the weighted number of such matchings cancels to zero.

Interpolation

In the counting world, Turing reductions become very powerful, as the results of oracle queries can be added, multiplied, subtracted—a luxury we do not have with Boolean values. This enables linear algebra on oracle outputs, in particular, *polynomial interpolation* [93, 92, 98, 29]. As an example of this prevalent technique, we show a classical reduction that computes $\mathrm{PerfMatch}(G)$ for an n-vertex graph G with edge-weights ± 1 when given an oracle for counting perfect matchings in unweighted graphs.

Let x be an indeterminate and define a graph G_x by replacing each edge of weight -1 in G by an edge of weight x. Then $p := \mathrm{PerfMatch}(G_x)$ is a polynomial in x of degree at most $\frac{n}{2}$ such that $p(-1) = \mathrm{PerfMatch}(G)$. If we knew the $\frac{n}{2} + 1$ values $p(0), \ldots, p(\frac{n}{2})$, we could recover all coefficients of $p(G)$ via Lagrange interpolation and could thus evaluate $p(-1)$. But the evaluation $p(i)$ for $0 \leq i \leq \frac{n}{2}$ can be obtained as $\mathrm{PerfMatch}(G_i)$ for the graph G_i in which each edge of weight x is replaced by an edge of weight i. Each such edge in turn can be simulated by a bundle of i parallel unweighted edges.

Overall, we can recover the value of $\mathrm{PerfMatch}(G)$ for the ± 1-weighted graph G by counting perfect matchings in the unweighted graphs $G_0, \ldots, G_{n/2}$ via oracle calls and performing linear algebra on the oracle results to interpolate the polynomial p.

Inclusion-exclusion

Also taking linear combinations of oracle outputs, the inclusion-exclusion principle is a very useful tool for *parameterized* hardness results [78, 39, 25, 33]. Given a universe Ω and "bad" subsets $A_1, \ldots, A_k \subseteq \Omega$, inclusion-exclusion allows us to count those elements of Ω that avoid all bad subsets, provided that we know the sizes of intersections of bad subsets:

$$\left|\Omega \setminus \bigcup_{i=1}^{k} A_i\right| = \sum_{S \subseteq \{1,\ldots,k\}} (-1)^{|S|} \, |A_S| \tag{1}$$

with $A_\emptyset := \Omega$ and $A_S := \bigcap_{i \in S} A_i$ for $\emptyset \subset S \subseteq [k]$. We can thus determine the left-hand side with 2^k oracle calls to numbers $|A_S|$ for $S \subseteq [k]$.

We use (1) for a parameterized reduction from the colorful subgraph isomorphism problem to the uncolored version [78, 33]. Let H and G be graphs such that G is (not necessarily properly) vertex-colored with colors from $\{1, \ldots, k\}$ for $k = |V(H)|$. A subgraph copy of H in G is *colorful* if it does not avoid any color. Thus, writing Ω for the set of all H-copies in G, and defining A_i for $i \in [k]$ to be the set of H-copies that *do avoid* color i, we can invoke (1) to obtain the number of colorful H-copies if we knew the cardinalities $|A_S|$ for $S \subseteq [k]$. But $|A_S|$ is just the number of H-copies avoiding all colors in S. These are precisely the H-copies in the graph G_S obtained from G by removing all vertices with a color among S and ignoring colors. The numbers of H-copies in G_S can then be obtained by oracle calls to counting uncolored H-copies, and we only need 2^k such numbers to evaluate (1).

Somewhat peculiarly, this reduction goes the opposite way as in the decision problem: In the decision version of subgraph isomorphism, *color-coding* reduces from the uncolored to the colorful setting, while the reverse direction is generally not known. In the counting version however, we have just seen a reduction from the uncolored to the colorful setting. Furthermore, the converse reduction is generally false: Counting k-paths is #W[1]-hard [48], whereas a simple FPT-algorithm counts colorful k-paths. See also an article by Meeks [78] for a nice overview of the relationship between colored and uncolored subgraph problems.

Organization of this survey

After this brief introduction, we apply the multivariate perspective on counting to two specific problems: In Section 2, we count small patterns in large graphs. Complementary to that, in Section 3, we then count large patterns like perfect matchings and Hamiltonian cycles in graphs excluding fixed minors or bounded treewidth.

2 Counting small patterns

Large networks can often be approached by counting small patterns [79, 73]. This naturally leads to parameterized pattern counting problems: Given as input a pattern graph H and a host graph G, determine the number of occurrences of H in G, parameterized by the size of H. Such problems have been intensively studied in different communities; we survey some results in Section 2.1 and present recent developments in Section 2.2.

2.1 Results for individual pattern types

There are different ways of formalizing an *occurrence* of a pattern in a graph, and they can lead to stark differences in complexity. Below are some of the most common notions:

- Let us write $\mathrm{sub}(H, G)$ for the number of (not necessarily induced) *subgraphs* of G that are isomorphic to H. Closely related is $\mathrm{emb}(H, G)$, the number of *embeddings* from H to G: An embedding from H to G is an injective function $f : V(H) \to V(G)$ such that $uv \in E(H)$ implies $f(u)f(v) \in E(G)$. We have $\mathrm{emb}(H, G) = \mathrm{aut}(H) \cdot \mathrm{sub}(H, G)$, where $\mathrm{aut}(H)$ is the number of automorphisms of H.
- Depending on the application, we may also ask for the number $\mathrm{ind}(H, G)$ of *induced subgraphs* isomorphic to H. Extending this, Jerrum and Meeks introduced a more general model [62, 61]: For a fixed property Φ of k-vertex graphs, we can ask to determine the number $\mathrm{ind}(\Phi, G)$ of k-vertex subsets inducing a subgraph with property Φ.
- We could also ask for the number $\mathrm{hom}(H, G)$ of *homomorphisms* from H to G. These are essentially embeddings that need not be injective, that is, functions $f : V(H) \to V(G)$ such that $uv \in E(H)$ implies $f(u)f(v) \in E(G)$. Homomorphism counts appear in the setting of database queries [22], they are a fundamental notion in the theory of graph limits and graph algebras [73], and we will see in Section 2.2 that they enable a depeer understanding of subgraph and induced subgraph counting problems.

Improved algorithms

All of the above numbers can be computed in time $O(f(k) \cdot n^k)$ by brute force; we write $k = |V(H)|$ and $n = |V(G)|$ throughout. As even a moderate value of k can be prohibitively large for large host graphs G, the possibility of improved algorithms was investigated:

- Fast matrix multiplication solves the above problems in time $f(k) \cdot n^{\omega k/3+O(1)}$, where $\omega < 2.373$ is the exponent of matrix multiplication [81].
- For computing $\mathrm{sub}(H, G)$ and thin patterns H, an additional improvement to $f(k) \cdot n^{0.46k+2\mathbf{pw}(H)+O(1)}$ time is possible, where $\mathbf{pw}(H)$ denotes the pathwidth of H [9].
- Again for $\mathrm{sub}(H, G)$, substantial improvements can be made when H has bounded vertex-cover number $\mathbf{vc}(H)$: As an example, if $H = K_{1,k}$ is a star with $k \geq 2$ rays, then $\mathrm{sub}(H, G) = \sum_{v \in V(G)} \binom{d(v)}{k}$, where $d(v)$ denotes the degree of v. This formula can be evaluated in linear time. More generally, if H has a vertex-cover C, then we can iterate over all $n^{|C|}$ maps from C into G and determine, for each map, the number of ways to extend the image of C to a full copy of H. This last step can be performed in polynomial time, enabling an overall running time of $f(k) \cdot n^{\mathbf{vc}(H)+O(1)}$ [69, 97].
- We will show in Section 2.2 that $\mathrm{sub}(H, G)$ can be computed in $f(k) \cdot n^{0.174\ell+O(1)}$ time when H has ℓ edges; this is useful for counting matchings, paths, or cycles.

It is natural to ask how far such improvements can be pushed. For fixed graphs H, let us write $\mathrm{sub}(H, \cdot)$, $\mathrm{ind}(H, \cdot)$ and $\mathrm{hom}(H, \cdot)$ for the graph parameters[1] that map input graphs G to the numbers $\mathrm{sub}(H, G)$, $\mathrm{ind}(H, G)$ and $\mathrm{hom}(H, G)$. As it is interesting to quantify running time improvements even for algorithms that are not FPT, we define the *complexity exponent* of graph parameters like $\mathrm{sub}(H, \cdot)$ to be the infimum over all $C \in \mathbb{R}$ such that $\mathrm{sub}(H, G)$ can be evlauated in time $O(n^C)$ on input G [30]. Note that the complexity exponent of $\mathrm{sub}(H, \cdot)$, $\mathrm{ind}(H, \cdot)$ and $\mathrm{hom}(H, \cdot)$ for fixed H is always bounded from above by k, but it may be well below k, say, for $\mathrm{sub}(H, \cdot)$ where H has low vertex-cover number.

[1] A graph parameter f is a function from graphs into $\mathbb{Q}$. Please note that *graph parameters* are not (necessarily) *parameters* in the sense of parameterized complexity—it is a bit awkward to talk about the “parameterized complexity of graph parameters”, but graph parameters are a well-established notion.

Hardness results

We want to understand the optimal complexity exponents of graph parameters that count small patterns. Parameterized complexity allows us to aggregate graph parameters into *classes* $\mathcal{H}$ and prove statements on the asymptotic behaviour of the complexity exponents: Given a graph class $\mathcal{H}$, we define the problem $\mathrm{sub}(\mathcal{H},\cdot)$, where the task is to compute $\mathrm{sub}(H,G)$ on input a pattern graph $H\in\mathcal{H}$ and an arbitrary host graph G. An analogous definition gives the problems $\mathrm{ind}(\mathcal{H},\cdot)$ and $\mathrm{hom}(\mathcal{H},\cdot)$.

If a problem $\mathrm{sub}(\mathcal{H},\cdot)$ is FPT or even polynomial-time solvable, then there is a fixed constant C that bounds all complexity exponents of $\mathrm{sub}(H,\cdot)$ for $H\in\mathcal{H}$. If $\mathrm{sub}(\mathcal{H},\cdot)$ is #W[1]-hard, then we believe that no such constant exists. Furthermore, conditional lower bounds under #ETH enable lower bounds on the asymptotic growth of complexity exponents. Indeed, such lower bounds are known for counting subgraphs, induced subgraphs, and homomorphisms for restricted pattern classes $\mathcal{H}$ that are recursively enumerable.

- If $\mathcal{H}$ is finite, then $\mathrm{ind}(\mathcal{H},\cdot)$ is polynomial-time solvable, otherwise it is #W[1]-hard [25]. Furthermore, if $\mathcal{H}$ is infinite, then there is no $f(k)\cdot n^{o(k)}$ time algorithm for $\mathrm{ind}(\mathcal{H},\cdot)$ unless ETH fails. Note that this result holds for *every* fixed recursively enumerable class $\mathcal{H}$; it shows that no pattern structure whatsoever can be exploited for counting induced subgraphs. Jerrum and Meeks generalized this result and proved that counting k-vertex subgraphs with fixed properties Φ is #W[1]-hard for certain well-behaved properties Φ like connectedness or having even/odd number of edges [61, 62, 63].
- For counting not necessarily induced subgraphs, we have seen above that pattern structure *can* be exploited: If the maximum vertex-cover number of graphs in $\mathcal{H}$ is finite, then $\mathrm{sub}(\mathcal{H},\cdot)$ is polynomial-time solvable. Otherwise the problem is known to be #W[1]-hard [33]. We will see in the next section that an $f(k)\cdot n^{o(\mathbf{vc}(H)/\log\mathbf{vc}(H))}$ time algorithm would refute #ETH, even for patterns from fixed classes $\mathcal{H}$.
- For counting homomorphisms, we may even allow bounded-treewidth patterns: If the treewidth of $\mathcal{H}$ is finite, then $\mathrm{hom}(\mathcal{H},\cdot)$ is polynomial-time solvable; otherwise the problem is #W[1]-hard [39]. As a consequence of Marx's cornerstone "Can you beat treewidth" paper, #ETH rules out an $f(k)\cdot n^{o(\mathbf{tw}(H)/\log\mathbf{tw}(H))}$ time algorithm [76].

2.2 Graph motif parameters

Despite the similar appearance of the problems $\mathrm{ind}(\mathcal{H},\cdot)$, $\mathrm{sub}(\mathcal{H},\cdot)$, and $\mathrm{hom}(\mathcal{H},\cdot)$, their dichotomy theorems were each proven about five years apart, by different people, using somewhat different techniques. Recently, it was shown that these problems *can* be studied in a uniform way: From the perspective of parameterized complexity, these problems become *the same problem* when "extended linearly".

More formally, graph parameters like $\mathrm{sub}(H,\cdot)$ or $\mathrm{ind}(H,\cdot)$ for fixed H are special cases of so-called *graph motif parameters*, a notion introduced in [30] that adapts early works by Lovász [71, 73] to study counting small patterns from a parameterized perspective. Apart from $\mathrm{sub}(H,\cdot)$ and $\mathrm{ind}(H,\cdot)$, graph motif parameters include the graph parameters $\mathrm{ind}(\Phi,\cdot)$ that count induced k-vertex subgraphs with a fixed property Φ, and generally, every graph parameter that depends only on the numbers of constant-sized subgraphs.

▶ **Definition 1.** A *graph motif parameter* is any graph parameter f that can be written as

$$f(\cdot)=\sum_{F\in\mathcal{F}}\alpha_F\cdot\mathrm{ind}(F,\cdot) \qquad (2)$$

for a finite set of graphs $\mathcal{F}$ and coefficients α_F for $F \in \mathcal{F}$. In other words, f is a (point-wise) linear combination of a finite set of functions $\mathrm{ind}(F, \cdot)$ for fixed patterns F. Recall that the function $\mathrm{ind}(F, \cdot)$ takes as input a graph G and outputs $\mathrm{ind}(F, G)$.

The graph motif parameter evaluation problem asks to evaluate $f(G)$ when given as input a graph G and a representation of f via $\mathcal{F} = \{F_1, \dots, F_t\}$ and coefficients $\alpha_1, \dots, \alpha_t$. We parameterize the problem by the description length of f.

Note that the functions $\mathrm{ind}(H, \cdot)$ for fixed H are graph motif parameters themselves, but using linear combinations of such functions allows us to express much more counting functions as graph motif parameters. As an example, for any H, we have

$$\mathrm{sub}(H, \cdot) = \sum_{F \text{ extends } H} \beta_F \cdot \mathrm{ind}(F, \cdot), \tag{3}$$

where an unlabeled graph F extends H if F is a supergraph of H on $|V(H)|$ vertices, and the coefficient β_F equals $\mathrm{sub}(F, H)$. Since only finitely many graphs F occur in (3), the graph parameter $\mathrm{sub}(H, \cdot)$ for fixed H is a graph motif parameter. Moreover, the graphs and coefficients can be computed in time depending only on H. It follows that the problem of computing $\mathrm{sub}(H, G)$ on input H and G, parameterized by H, admits a parameterized reduction to the graph motif evaluation problem.

Interestingly, (3) admits a converse form that expresses the number of induced subgraphs $\mathrm{ind}(H, \cdot)$ as a finite linear combination of subgraph counts $\mathrm{sub}(F, \cdot)$, see [73, (5.17)]. Thus, subgraph counts are finite linear combinations of induced subgraph counts, *and vice versa.* We could thus replace ind with sub in Definition 1 and still end up defining the same objects: Any graph motif parameter $f = \sum_{F \in \mathcal{F}} \alpha_F \cdot \mathrm{ind}(F, \cdot)$ can also be written as

$$f(\cdot) = \sum_{F \in \mathcal{F}'} \beta_F \cdot \mathrm{sub}(F, \cdot). \tag{4}$$

for a finite set of graphs $\mathcal{F}'$ and coefficients β_F for $F \in \mathcal{F}'$ that can be computed from $\mathcal{F}$ and the coefficients α_F for $F \in \mathcal{F}$. We call (4) the *sub-expansion* of f and (2) its *ind-expansion.*[2]

Homomorphism counts count

There are actually many equivalent ways of expanding graph motif parameters into finite linear combinations of basis functions, with explicit formulas for changing bases between such expansions [73, Chapter 5.2]. As it turns out, such basis changes enable a much better understanding of the *complexity* of graph motif parameters. In particular, expanding graph motif parameters into linear combinations of the previously neglected *homomorphism* counts will prove to be extremely useful for algorithmic purposes.

Let us have a more detailed look at the relation between embeddings[3] and homomorphisms: We can easily express the number of homomorphisms $\mathrm{hom}(H, \cdot)$ as a finite linear combination of embeddings $\mathrm{emb}(F, \cdot)$. A graph F is a *quotient* of H if F can be obtained from H by repeated identifications of vertex pairs. These identifications can be performed all at once: If π is a partition of $V(H)$, then the quotient H/π is obtained from H by identifying, for each

[2] We could have chosen to call the parameterized counting problem from Definition 1 the "evaluation problem for graph motif parameters in the ind-expansion" and could have defined an analogous version for the sub-expansion, but the fact that the sub-expansion and the ind-expansion can be transformed into each other algorithmically implies that these problems are FPT-interreducible.

[3] Recall that an embedding is an injective homomorphism, and that $\mathrm{emb}(H, \cdot) = \mathrm{aut}(H) \cdot \mathrm{sub}(H, \cdot)$.

block B of π, the vertices in B to a single vertex. Quotients of graphs H correspond to the images that H can attain under homomorphisms, and we obtain

$$\mathrm{hom}(H,\cdot) = \sum_{\pi} \mathrm{emb}(H/\pi,\cdot), \tag{5}$$

where π ranges over all partitions of $V(H)$. Different partitions can yield isomorphic quotients, and after collecting for isomorphic copies in (5) and rewriting emb $=$ aut $\cdot$ sub, we obtain a formula for the sub-expansion of $\mathrm{hom}(H,\cdot)$.

More interestingly, as shown in [73, (5.18)], the formula (5) can be inverted[4] to read

$$\mathrm{emb}(H,\cdot) = \sum_{\pi} \underbrace{(-1)^{k-|\pi|} \prod_{B\in\pi} (|B|-1)!}_{=:\mu(\pi)} \cdot \mathrm{hom}(H/\pi,\cdot), \tag{6}$$

where we write $|\pi|$ for the number of blocks in π. The precise form of (6) and $\mu(\pi)$ will become important later on, but for now let us just use it to conclude that every graph motif parameter f can be expressed as a linear combination of homomorphism counts

$$f(\cdot) = \sum_{F\in\mathcal{F}''} \gamma_F \cdot \mathrm{hom}(F,\cdot) \tag{7}$$

for a finite set of graphs $\mathcal{F}''$ and coefficients γ_F for $F \in \mathcal{F}''$. We call (7) the *hom-expansion* of f. This works because every graph motif parameter has a sub-expansion via (4), and because subgraph counts are rescaled embedding counts, which in turn are linear combinations of homomorphism counts by (6).

Complexity monotonicity of the hom-expansion

Lovász showed that homomorphism counts enjoy a wealth of mathematical properties that make them more well-behaved than subgraph or induced subgraph counts [73], and some of these properties translate almost directly into complexity-theoretic properties. Most importantly for us, the hom-expansion of graph motif parameters enjoys a monotonicity property with respect to complexity exponents: Any linear combination of homomorphism counts, as in (7), is unconditionally at least as hard to evaluate as its hardest terms.

▶ **Theorem 2** ([30, 22]). *Let f be a graph motif parameter with $f = \sum_{F\in\mathcal{F}} \alpha_F \cdot \mathrm{hom}(F,\cdot)$ for a finite set of pairwise non-isomorphic graphs $\mathcal{F}$ with $\alpha_F \neq 0$ for all $F \in \mathcal{F}$. Then the complexity exponent of f is exactly the maximum over all complexity exponents of the functions* $\mathrm{hom}(F,\cdot)$ *for $F \in \mathcal{F}$.*

▶ Remark. A result similar to Theorem 2 cannot be obtained for the sub-expansion or the ind-expansion. In these cases, we can "hide" cliques in easy linear combinations: For any $k \in \mathbb{N}$, summing over the induced subgraph counts from all k-vertex graphs H gives $\sum_H \mathrm{ind}(H,G) = \binom{|V(G)|}{k}$, which can be evaluated in linear time for any graph G, even though the left-hand side contains a k-clique. Theorem 2 rules this out for homomorphisms.

Via Theorem 2, we obtain a good grip on the complexity of a graph motif parameter f if we manage to understand (i) what patterns F occur in the hom-expansion of f, and (ii) how hard counting homomorphisms $\mathrm{hom}(F,\cdot)$ is for these patterns. For the first item, we can use enumerative combinatorics to obtain formulas like (6). For the second item, we can use the known lower bounds under ETH for cliques [23] or patterns of large treewidth [76].

[4] This is done via Möbius inversion on the partition lattice, a generalization of the inclusion-exclusion principle, which can be viewed as Möbius inversion on the subset lattice [88].

Understanding subgraph counting via homomorphisms

In the following, we use the two-step approach to give upper and lower bounds on the complexity of counting subgraphs $\mathrm{sub}(H, \cdot)$ for fixed H. Let us start with an upper bound.

▶ **Theorem 3** ([30]). *On input H and G, the number $\mathrm{sub}(H, G)$ can be determined in time $k^{O(k)} \cdot n^{0.174\ell+O(1)}$ with $\ell = |E(H)|$.*

Proof. Consider the formula (6) for transforming embedding counts into homomorphism counts. The right-hand side sums over all partitions π of $V(H)$, of which there are at most $k^{O(k)}$, and requires homomorphism numbers from quotients H/π into G. Any quotient H/π has at most ℓ edges, and the treewidth of ℓ-edge graphs F is known to be at most $\mathbf{tw}(F) \leq 0.174\ell + o(\ell)$ [87]. By a standard DP approach, we can determine $\mathrm{hom}(H/\pi, G)$ in time $2^{O(k)} \cdot n^{0.174\ell+O(1)}$ for every fixed π, and the overall running time bound follows. ◀

It is not much harder to prove hardness results for subgraph counting via Theorem 2:

▶ **Theorem 4.** *For any fixed recursively enumerable class $\mathcal{H}$, the problem $\mathrm{sub}(\mathcal{H}, \cdot)$ cannot be solved in time $f(k) \cdot n^{o(\mathbf{vc}(H)/\log \mathbf{vc}(H))}$ for patterns $H \in \mathcal{H}$ unless ETH fails.*

Proof. Consider $\mathrm{sub}(H, \cdot)$ for $H \in \mathcal{H}$. If H has vertex-cover number $b \in \mathbb{N}$, then H contains a matching M with $t = b/2$ edges. We show that for every t-edge graph S, the hom-expansion of $\mathrm{sub}(H, \cdot)$ contains a supergraph of S with non-zero coefficient. In particular, this holds for bounded-degree expanders S of treewidth $\Omega(t)$. Supergraphs have only larger treewidth.

Every t-edge graph S is some quotient of the t-matching M in H. It follows that some supergraph of S is a quotient of H. Using (6), we can see that every quotient H/π appears with non-zero coefficient in the hom-expansion of $\mathrm{sub}(H, \cdot)$ even after collecting for isomorphic graphs: While different partitions π, π' may lead to isomorphic quotients, this requires $|\pi| = |\pi'|$. But since the sign of $\mu(\pi)$ in (6) depends only on $|\pi|$, the contributions of π and π' cannot cancel. ◀

The hom-expansion of graph motif parameters is a fascinating object of study on its own that connects various mathematical areas: After the results for subgraph counting, Theorem 2 was used by Roth [85] to classify the complexity of counting homomorphism variants such as locally injective homomorphisms. This was obtained by finding large-treewidth patterns in the relevant hom-expansions through matroid theory. Very recently, an almost exhaustive #W[1]-hardness proof for counting induced k-vertex subgraphs with a fixed monotone property Φ was given by Roth and Schmitt [86] by using topological properties of abstract simplicial complexes. In unpublished work by the author, connections to finite model theory and chromatic polynomials were also uncovered.

3 Counting large patterns in "simple" graphs

In this last and shorter part of the survey, we count large objects in graphs of simple structure: In Section 3.1, we attempt to count perfect matchings in graphs that exclude fixed minors, and in Section 3.2, we count Hamiltonian cycles in graphs of bounded pathwidth.

3.1 Counting perfect matchings in minor-free graphs

Counting perfect matchings was the first problem shown to be #P-hard for interesting reasons, and it admits a beautiful polynomial-time algorithm on planar graphs, the *Fisher-Kasteleyn-Temperley (FKT) method* [90, 65, 66], which we already mentioned earlier. Other than planar graphs, there are further tractable graph classes $\mathcal{C}$: For instance, while planar graphs exclude both $K_{3,3}$ and K_5 as minors, it was shown that excluding *either* of these two minors is sufficient [70, 96, 89]. The FKT method was also extended to an FPT-algorithm with running time $4^\gamma n^{O(1)}$ for graphs that are embedded on a surface of genus γ [53]; together with an FPT-algorithm for finding such embeddings [80], this shows that counting perfect matchings is FPT when parameterized by genus. Furthermore, dynamic programming with subset convolution yields an $O(2^{\mathbf{tw}(G)}n)$ time algorithm when G is given with an optimal tree-decomposition [95]. It is also possible to count perfect matchings in time $O(n^{\mathbf{cw}(G)+1})$ on graphs of cliquewidth $\mathbf{cw}(G)$ given with an optimal parse-tree, but unlike the previous results, this holds only for unweighted graphs [75].

The above list shows that, apart from cliquewidth, all tractability results for counting perfect matchings address graphs that exclude some fixed minor. In fact, the algorithms for $K_{3,3}$-free and K_5-free graphs make use of classical precursors to Robertson and Seymour's graph structure theorem for H-minor free graphs [84]: By Wagner's theorem [43], the $K_{3,3}$-free and K_5-free graphs admit tree-decompositions of adhesion 3 in which every torso is planar or has bounded size. Prior to proving the full graph structure theorem, Robertson and Seymour showed very similar decompositions for all graphs excluding minors H that can be drawn in the plane with at most one crossing [83], such as $H = K_{3,3}$ and $H = K_5$. Such decompositions can be used algorithmically:

▶ **Theorem 5** ([28, 89]). *For any fixed graph H that can be drawn in the plane with at most one crossing, there is an $O(n^4)$ time algorithm for counting perfect matchings in H-minor free graphs G.*

Proof. For single-crossing minors H, a tree-decomposition of H-free graphs G into planar and bounded-treewidth torsos of adhesion 3 can be obtained in $O(f(H) \cdot n^4)$ time [41]. Given such a decomposition, the algorithm now uses a counting version of protrusion replacement [12]: When counting perfect matchings, any 3-boundaried graph S can be simulated by an equivalent planar 3-boundaried graph S' with edge-weights [94], and the graph S' can be computed in FPT-time for planar and bounded-treewidth graphs S. Traversing the decomposition of G bottom-up, we can successively replace torsos S by planar replacement graphs S' until all of G is processed. ◀

How far can we push this result? In particular, is it possible to obtain polynomial-time algorithms for counting perfect matchings for *any* graph class that excludes some fixed graph H? More strongly, is the problem FPT when parameterized by the *Hadwiger number*, which is the maximum size of a clique minor in G? The answer to the second question is negative, as we can show #W[1]-hardness on k-apex graphs. These are the graphs that become planar after deleting at most k vertices; this upper-bounds their Hadwiger number by $k + 5$.

▶ **Theorem 6** ([35]). *Counting perfect matchings in k-apex graphs is #W[1]-hard and an $f(k)n^{o(k/\log k)}$ time algorithm would refute #ETH.*

Thus, counting perfect matchings is also #W[1]-hard when parameterized by the Hadwiger number. However, an XP-algorithm might still be possible, e.g., because there is a simple $O(n^{k+3})$ time algorithm for counting perfect matchings in k-apex graphs: Simply iterate over all n^k possible choices for matching the k apices to vertices in the planar base graph, delete the apices together with their matching partners, and count perfect matchings in the remaining *planar* graph via the FKT method.

To recapitulate, we know that counting perfect matchings is FPT parameterized by genus, in XP parameterized by apex number, and we have seen some techniques for dealing with clique-sums in Theorem 5. By the graph structure theorem [84], these are almost all the building blocks of H-minor free graphs, and thus an XP algorithm for counting perfect matchings on such graphs almost seems within reach.

Alas, we blocked out *vortices*: In general, H-minor free graphs allow for graphs of bounded pathwidth to be inserted at the faces of the bounded-genus parts; these structures are called vortices. It turns out that counting perfect matchings is #P-hard even in planar graphs with one single vortex, and such graphs can be seen to exclude a constant-sized minor.

▶ **Theorem 7** (Unpublished work by the author with Mingji Xia). *There is a fixed graph H such that counting perfect matchings is #P-hard in graphs excluding H.*

Thus, polynomial-time algorithms cannot be obtained for each excluded minor H, but it is still open to find out *which* fixed graphs H can be excluded for polynomial-time algorithms.

3.2 Counting Hamiltonian cycles in low-pathwidth graphs

To conclude, we briefly consider counting Hamiltonian cycles parameterized by pathwidth. The standard DP for this problem gives a running time of $O^*(\mathbf{pw}^{O(\mathbf{pw})})$ for counting and decision, but celebrated improvements [37] enabled $O^*((2+\sqrt{2})^{\mathbf{pw}})$ time for the decision problem and for counting modulo 2, and $O^*(6^{\mathbf{pw}})$ time for counting [11]. The bases $2+\sqrt{2}$ and 6 are optimal under the strong exponential-time hypothesis SETH [37, 32].

Curiously, the concrete numbers $2+\sqrt{2}$ and 6 can be explained through *connection matrices*. These are particularly nice objects in the intersection of linear algebra and combinatorics [72, 73]. Given a graph parameter f such as the number of Hamiltonian cycles, and $k \in \mathbb{N}$, the k-th connection matrix of f is an infinite matrix $C_{f,k}$, indexed by k-boundaried graphs, that describes the behavior of f under graph separations of size k. The entry $C_{f,k}(G,H)$ for k-boundaried graphs G, H is defined to be $f(G \oplus H)$, where $G \oplus H$ is the union of k-boundaried graphs G and H with matching vertices identified. Even though the matrices $C_{f,k}$ are infinite-sized, they have finite rank for many interesting graph parameters f, including all MSOL-definable f [68]. It can be shown furthermore that evaluating graph parameters f with finite-rank connection matrices is nonuniformly FPT when parameterized by treewidth [73]; this gives an alternative proof of Courcelle's theorem.

Let us focus again on the connection matrix $C_{\mathrm{HC},k}$ for the number of Hamiltonian cycles: Lovász [72] upper-bounded the rank of $C_{\mathrm{HC},k}$ over $\mathbb{Q}$ by $k^{O(k)}$. This bound was recently improved to $\Theta(6^k)$ up to polynomial factors, with a matching lower bound [32]. It was also known before that the rank of $C_{\mathrm{HC},k}$ drops to $\Theta((2+\sqrt{2})^k)$ over the ring $\mathbb{Z}_2$ with a matching lower bound [37]. Note that the constants 6 and $2+\sqrt{2}$ appearing in the rank of $C_{\mathrm{HC},k}$ over $\mathbb{Q}$ and $\mathbb{Z}_2$ are the running time bases for counting Hamiltonian cycles over these rings parameterized by pathwidth. This is a consequence of a more general statement:

▶ **Theorem 8** ([32]). *Let $\mathfrak{R}$ be any of the rings $\mathbb{Q}$ or $\mathbb{Z}_p$ for prime p. If the connection matrix $C_{\mathrm{HC},k}$ of the number of Hamiltonian cycles has rank $\Omega(c^k)$ over $\mathfrak{R}$, then an algorithm with running time $O^*((c-\epsilon)^{\mathbf{pw}})$ for counting Hamiltonian cycles over $\mathfrak{R}$ refutes SETH.*

While we do know that the rank of $C_{\mathrm{HC},k}$ is $\Omega(3.97^k)$ over $\mathbb{Z}_p$ when $p \neq 2$, resulting in a corresponding complexity lower bound [32], algorithms faster than $O^*(6^{\mathbf{pw}})$ are not known. It is also interesting to close this gap and to investigate to what extent the proof technique in Theorem 8 can be applied to study problems other than Hamiltonian cycles through their connection matrices.

References

1 Noga Alon, Phuong Dao, Iman Hajirasouliha, Fereydoun Hormozdiari, and S Cenk Sahinalp. Biomolecular network motif counting and discovery by color coding. *Bioinformatics*, 24(13):i241–i249, 2008. doi:10.1093/bioinformatics/btn163.

2 Noga Alon and Shai Gutner. Balanced Hashing, Color Coding and Approximate Counting. In *Parameterized and Exact Computation, 4th International Workshop, IWPEC 2009, Copenhagen, Denmark, September 10-11, 2009, Revised Selected Papers*, pages 1–16, 2009. doi:10.1007/978-3-642-11269-0_1.

3 Noga Alon and Shai Gutner. Balanced families of perfect hash functions and their applications. *ACM Trans. Algorithms*, 6(3):54:1–54:12, 2010. doi:10.1145/1798596.1798607.

4 Noga Alon, Raphael Yuster, and Uri Zwick. Color-Coding. *J. ACM*, 42(4):844–856, 1995. doi:10.1145/210332.210337.

5 Stefan Arnborg, Jens Lagergren, and Detlef Seese. Easy Problems for Tree-Decomposable Graphs. *J. Algorithms*, 12(2):308–340, 1991. doi:10.1016/0196-6774(91)90006-K.

6 Vikraman Arvind and Venkatesh Raman. Approximation Algorithms for Some Parameterized Counting Problems. In *Algorithms and Computation, 13th International Symposium, ISAAC 2002 Vancouver, BC, Canada, November 21-23, 2002, Proceedings*, pages 453–464, 2002. doi:10.1007/3-540-36136-7_40.

7 Andreas Björklund. Determinant Sums for Undirected Hamiltonicity. In *51th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2010, October 23-26, 2010, Las Vegas, Nevada, USA*, pages 173–182, 2010. doi:10.1109/FOCS.2010.24.

8 Andreas Björklund, Thore Husfeldt, and Mikko Koivisto. Set Partitioning via Inclusion-Exclusion. *SIAM Journal on Computing*, 39(2):546–563, 2009. doi:10.1137/070683933.

9 Andreas Björklund, Petteri Kaski, and Lukasz Kowalik. Counting Thin Subgraphs via Packings Faster than Meet-in-the-Middle Time. *ACM Trans. Algorithms*, 13(4):48:1–48:26, 2017. doi:10.1145/3125500.

10 Markus Bläser and Radu Curticapean. Weighted Counting of k-Matchings Is #W[1]-Hard. In *Parameterized and Exact Computation - 7th International Symposium, IPEC 2012, Ljubljana, Slovenia, September 12-14, 2012. Proceedings*, pages 171–181, 2012. doi:10.1007/978-3-642-33293-7_17.

11 Hans L. Bodlaender, Marek Cygan, Stefan Kratsch, and Jesper Nederlof. Deterministic single exponential time algorithms for connectivity problems parameterized by treewidth. *Inf. Comput.*, 243:86–111, 2015. doi:10.1016/j.ic.2014.12.008.

12 Hans L. Bodlaender, Fedor V. Fomin, Daniel Lokshtanov, Eelko Penninkx, Saket Saurabh, and Dimitrios M. Thilikos. (Meta) Kernelization. *J. ACM*, 63(5):44:1–44:69, 2016. doi:10.1145/2973749.

13 Cornelius Brand, Holger Dell, and Thore Husfeldt. Extensor-coding. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, pages 151–164, 2018. doi:10.1145/3188745.3188902.

14 Cornelius Brand, Holger Dell, and Marc Roth. Fine-Grained Dichotomies for the Tutte Plane and Boolean #CSP. In *11th International Symposium on Parameterized and Exact Computation, IPEC 2016, August 24-26, 2016, Aarhus, Denmark*, pages 9:1–9:14, 2016. doi:10.4230/LIPIcs.IPEC.2016.9.

15 Cornelius Brand and Marc Roth. Parameterized Counting of Trees, Forests and Matroid Bases. In *Computer Science - Theory and Applications - 12th International Computer Science Symposium in Russia, CSR 2017, Kazan, Russia, June 8-12, 2017, Proceedings*, pages 85–98, 2017. doi:10.1007/978-3-319-58747-9_10.

16 Andrei Bulatov and Martin Grohe. The complexity of partition functions. *Theoretical Computer Science*, 348(2):148–186, 2005. doi:10.1016/j.tcs.2005.09.011.

17 Andrei A. Bulatov. The complexity of the counting constraint satisfaction problem. *J. ACM*, 60(5):34, 2013. doi:10.1145/2528400.

18 Andrei A. Bulatov. A Dichotomy Theorem for Nonuniform CSPs. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 319–330, 2017. doi:10.1109/FOCS.2017.37.

19 Jin-Yi Cai and Xi Chen. Complexity of Counting CSP with Complex Weights. *J. ACM*, 64(3):19:1–19:39, 2017. doi:10.1145/2822891.

20 Jin-yi Cai and Pinyan Lu. Holographic algorithms: From art to science. *J. Comput. Syst. Sci.*, 77(1):41–61, 2011. doi:10.1016/j.jcss.2010.06.005.

21 Jin-yi Cai, Pinyan Lu, and Mingji Xia. Computational Complexity of Holant Problems. *SIAM J. Comput.*, 40(4):1101–1132, 2011. doi:10.1137/100814585.

22 Hubie Chen and Stefan Mengel. Counting Answers to Existential Positive Queries: A Complexity Classification. In *Proceedings of the 35th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS 2016, San Francisco, CA, USA, June 26 - July 01, 2016*, pages 315–326, 2016. doi:10.1145/2902251.2902279.

23 Jianer Chen, Benny Chor, Mike Fellows, Xiuzhen Huang, David W. Juedes, Iyad A. Kanj, and Ge Xia. Tight lower bounds for certain parameterized NP-hard problems. *Information and Computation*, 201(2):216–231, 2005. doi:10.1016/j.ic.2005.05.001.

24 Xi Chen, Martin E. Dyer, Leslie Ann Goldberg, Mark Jerrum, Pinyan Lu, Colin McQuillan, and David Richerby. The complexity of approximating conservative counting CSPs. *J. Comput. Syst. Sci.*, 81(1):311–329, 2015. doi:10.1016/j.jcss.2014.06.006.

25 Yijia Chen, Marc Thurley, and Mark Weyer. Understanding the Complexity of Induced Subgraph Isomorphisms. In *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*, pages 587–596, 2008. doi:10.1007/978-3-540-70575-8_48.

26 Bruno Courcelle, Johann A. Makowsky, and Udi Rotics. On the fixed parameter complexity of graph enumeration problems definable in monadic second-order logic. *Discrete Applied Mathematics*, 108(1-2):23–52, 2001. doi:10.1016/S0166-218X(00)00221-3.

27 Radu Curticapean. Counting Matchings of Size k Is #W[1]-Hard. In *Automata, Languages, and Programming - 40th International Colloquium, ICALP 2013, Riga, Latvia, July 8-12, 2013, Proceedings, Part I*, pages 352–363, 2013. doi:10.1007/978-3-642-39206-1_30.

28 Radu Curticapean. Counting perfect matchings in graphs that exclude a single-crossing minor. *CoRR*, abs/1406.4056, 2014. URL: http://arxiv.org/abs/1406.4056, arXiv:1406.4056.

29 Radu Curticapean. Block interpolation: A framework for tight exponential-time counting complexity. *Inf. Comput.*, 261(Part):265–280, 2018. doi:10.1016/j.ic.2018.02.008.

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

31 Radu Curticapean, Holger Dell, and Marc Roth. Counting Edge-Injective Homomorphisms and Matchings on Restricted Graph Classes. In *34th Symposium on Theoretical Aspects of Computer Science, STACS 2017, March 8-11, 2017, Hannover, Germany*, pages 25:1–25:15, 2017. doi:10.4230/LIPIcs.STACS.2017.25.

32 Radu Curticapean, Nathan Lindzey, and Jesper Nederlof. A Tight Lower Bound for Counting Hamiltonian Cycles via Matrix Rank. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018, New Orleans, LA, USA, January 7-10, 2018*, pages 1080–1099, 2018. doi:10.1137/1.9781611975031.70.

33 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, 2014. doi:10.1109/FOCS.2014.22.

34 Radu Curticapean and Dániel Marx. Tight conditional lower bounds for counting perfect matchings on graphs of bounded treewidth, cliquewidth, and genus. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages 1650–1669, 2016. doi:10.1137/1.9781611974331.ch113.

35 Radu Curticapean and Mingji Xia. Parameterizing the Permanent: Genus, Apices, Minors, Evaluation Mod 2k. In *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*, pages 994–1009, 2015. doi:10.1109/FOCS.2015.65.

36 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.

37 Marek Cygan, Stefan Kratsch, and Jesper Nederlof. Fast Hamiltonicity Checking Via Bases of Perfect Matchings. *J. ACM*, 65(3):12:1–12:46, 2018. doi:10.1145/3148227.

38 Paul Dagum and Michael Luby. Approximating the Permanent of Graphs with Large Factors. *Theoretical Computer Science*, 102(2):283–305, 1992. doi:10.1016/0304-3975(92)90234-7.

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

40 Holger Dell, Thore Husfeldt, Dániel Marx, Nina Taslaman, and Martin Wahlen. Exponential Time Complexity of the Permanent and the Tutte Polynomial. *ACM Transactions on Algorithms*, 10(4):21, 2014. doi:10.1145/2635812.

41 E. Demaine, M. Hajiaghayi, and D. Thilikos. Exponential Speedup of Fixed-Parameter Algorithms for Classes of Graphs Excluding Single-Crossing Graphs as Minors. *Algorithmica*, 41(4):245–267, 2005. doi:10.1007/s00453-004-1125-y.

42 Erik D. Demaine and MohammadTaghi Hajiaghayi. The Bidimensionality Theory and Its Algorithmic Applications. *Comput. J.*, 51(3):292–302, 2008. doi:10.1093/comjnl/bxm033.

43 Reinhard Diestel. *Graph Theory, 4th Edition*, volume 173 of *Graduate texts in mathematics*. Springer, 2012.

44 Rodney G. Downey and Michael R. Fellows. *Parameterized Complexity*. Monographs in Computer Science. Springer, 1999. doi:10.1007/978-1-4612-0515-9.

45 Martin Dyer, Leslie Ann Goldberg, and Mike Paterson. On counting homomorphisms to directed acyclic graphs. *J. ACM*, 54, December 2007. doi:10.1145/1314690.1314691.

46 Martin E. Dyer and Catherine S. Greenhill. The complexity of counting graph homomorphisms. *Random Struct. Algorithms*, 17(3-4):260–289, 2000. doi:10.1002/1098-2418(200010/12)17:3/4\%3C260::AID-RSA5\%3E3.0.CO;2-W.

47 Martin E. Dyer and David Richerby. An Effective Dichotomy for the Counting Constraint Satisfaction Problem. *SIAM J. Comput.*, 42(3):1245–1274, 2013. doi:10.1137/100811258.

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

49 Jörg Flum and Martin Grohe. *Parameterized Complexity Theory*. Springer, 2006.

50 Fedor V. Fomin, Erik D. Demaine, and Mohammad Taghi Hajiaghayi. Bidimensionality. In *Encyclopedia of Algorithms*. Springer, 2015. doi:10.1007/978-3-642-27848-8_47-2.

51 Fedor V. Fomin and Dieter Kratsch. *Subset Convolution*, pages 125–139. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010. doi:10.1007/978-3-642-16533-7_7.

52 Markus Frick. Generalized Model-Checking over Locally Tree-Decomposable Classes. *Theory Comput. Syst.*, 37(1):157–191, 2004. doi:10.1007/s00224-003-1111-9.

53 Anna Galluccio and Martin Loebl. On the Theory of Pfaffian Orientations. I. Perfect Matchings and Permanents. *Electronic Journal of Combinatorics*, 6, 1998.

54 Leslie Ann Goldberg and Mark Jerrum. Approximating the Tutte polynomial of a binary matroid and other related combinatorial polynomials. *J. Comput. Syst. Sci.*, 79(1):68–78, 2013. doi:10.1016/j.jcss.2012.04.005.

55 Leslie Ann Goldberg and Mark Jerrum. The Complexity of Approximately Counting Tree Homomorphisms. *TOCT*, 6(2):8, 2014. doi:10.1145/2600917.

56 Mark S. Granovetter. The Strength of Weak Ties. *American Journal of Sociology*, 78(6):1360–1380, 1973.

57 Thore Husfeldt. Invitation to Algorithmic Uses of Inclusion-Exclusion. In *Automata, Languages and Programming - 38th International Colloquium, ICALP 2011, Zurich, Switzerland, July 4-8, 2011, Proceedings, Part II*, pages 42–59, 2011. doi:10.1007/978-3-642-22012-8_3.

58 Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which problems have strongly exponential complexity? *J. Comput. System Sci.*, 63(4):512–530, 2001.

59 A. Isihara. *Statistical physics.* Academic Press, 1971.

60 M. Jerrum, A. Sinclair, and E. Vigoda. A polynomial-time approximation algorithm for the permanent of a matrix with nonnegative entries. *J. ACM*, 51(4):671–697, 2004. doi:10.1145/1008731.1008738.

61 Mark Jerrum and Kitty Meeks. Some hard families of parameterized counting problems. *ACM Transactions on Computation Theory*, 7(3):11, 2015. doi:10.1145/2786017.

62 Mark Jerrum and Kitty Meeks. The parameterised complexity of counting connected subgraphs and graph motifs. *J. Comput. Syst. Sci.*, 81(4):702–716, 2015. doi:10.1016/j.jcss.2014.11.015.

63 Mark Jerrum and Kitty Meeks. The parameterised complexity of counting even and odd induced subgraphs. *Combinatorica*, pages 1–26, 2016. doi:10.1007/s00493-016-3338-5.

64 Mark Jerrum and Alistair Sinclair. Polynomial-time approximation algorithms for the Ising model. *SIAM Journal on Computing*, 22(5):1087–1116, 1993. doi:10.1137/0222066.

65 Pieter W. Kasteleyn. The statistics of dimers on a lattice: I. The number of dimer arrangements on a quadratic lattice. *Physica*, 27(12):1209–1225, 1961. doi:10.1016/0031-8914(61)90063-5.

66 Pieter W. Kasteleyn. Graph Theory and Crystal Physics. In *Graph Theory and Theoretical Physics*, pages 43–110. Academic Press, 1967.

67 Gustav Kirchhoff. Über die Auflösung der Gleichungen, auf welche man bei der Untersuchung der linearen Vertheilung galvanischer Ströme geführt wird. *Annalen der Physik und Chemie*, LXXIL(12), 1847.

68 Tomer Kotek and Johann A. Makowsky. Connection Matrices and the Definability of Graph Parameters. *Logical Methods in Computer Science*, 10(4), 2014. doi:10.2168/LMCS-10(4:1)2014.

69 Miroslaw Kowaluk, Andrzej Lingas, and Eva-Marta Lundell. Counting and Detecting Small Subgraphs via Equations. *SIAM J. Discrete Math.*, 27(2):892–909, 2013. doi:10.1137/110859798.

70 Charles Little. An Extension of Kasteleyn's method of enumerating the 1-factors of planar graphs. In *Combinatorial Mathematics*, LNCS, pages 63–72. Springer, 1974. doi:10.1007/BFb0057377.

71 László Lovász. Operations with structures. *Acta Mathematica Hungarica*, 18(3-4):321–328, 1967.

72 László Lovász. The rank of connection matrices and the dimension of graph algebras. *Eur. J. Comb.*, 27(6):962–970, 2006. doi:10.1016/j.ejc.2005.04.012.

73 László Lovász. *Large networks and graph limits*, volume 60. American Mathematical Society Providence, 2012.

74 Johann A. Makowsky. Algorithmic uses of the Feferman-Vaught Theorem. *Annals of Pure and Applied Logic*, 126(1-3):159–213, 2004. Provinces of logic determined. Essays in the memory of Alfred Tarski. Parts I, II and III. doi:10.1016/j.apal.2003.11.002.

75 Johann A. Makowsky, Udi Rotics, Ilya Averbouch, and Benny Godlin. Computing Graph Polynomials on Graphs of Bounded Clique-Width. In *Graph-Theoretic Concepts in Computer Science, 32nd International Workshop, WG 2006, Bergen, Norway, June 22-24, 2006, Revised Papers*, pages 191–204, 2006. doi:10.1007/11917496_18.

76 Dániel Marx. Can You Beat Treewidth? *Theory of Computing*, 6(1):85–112, 2010. doi:10.4086/toc.2010.v006a005.

77 Catherine McCartin. Parameterized counting problems. *Ann. Pure Appl. Logic*, 138(1-3):147–182, 2006. doi:10.1016/j.apal.2005.06.010.

78 Kitty Meeks. The challenges of unbounded treewidth in parameterised subgraph counting problems. *Discrete Applied Mathematics*, 198:170–194, 2016. doi:10.1016/j.dam.2015.06.019.

79 Ron Milo, Shai Shen-Orr, Shalev Itzkovitz, Nadav Kashtan, Dmitri Chklovskii, and Uri Alon. Network motifs: Simple building blocks of complex networks. *Science*, 298(5594):824–827, 2002. doi:10.1126/science.298.5594.824.

80 Bojan Mohar. A Linear Time Algorithm for Embedding Graphs in an Arbitrary Surface. *SIAM J. Discrete Math.*, 12(1):6–26, 1999. URL: http://epubs.siam.org/sam-bin/dbq/article/29248.

81 Jaroslav Nešetřil and Svatopluk Poljak. On the complexity of the subgraph problem. *Commentationes Mathematicae Universitatis Carolinae*, 26(2):415–419, 1985.

82 Rolf Niedermeier. *Invitation to fixed-parameter algorithms*, volume 31 of *Oxford Lecture Series in Mathematics and its Applications*. Oxford University Press, Oxford, 2006.

83 Neil Robertson and Paul D. Seymour. Excluding a graph with one crossing. In *Proceedings of the AMS-IMS-SIAM Joint Summer Research Conference on Graph Minors, Held June 22 to July 5, 1991*, pages 669–675, 1993.

84 Neil Robertson and Paul D. Seymour. Graph Minors. XVI. Excluding a non-planar graph. *J. Comb. Theory, Ser. B*, 89(1):43–76, 2003. doi:10.1016/S0095-8956(03)00042-X.

85 Marc Roth. Counting Restricted Homomorphisms via Möbius Inversion over Matroid Lattices. In *25th Annual European Symposium on Algorithms, ESA 2017, September 4-6, 2017, Vienna, Austria*, pages 63:1–63:14, 2017. doi:10.4230/LIPIcs.ESA.2017.63.

86 Marc Roth and Johannes Schmitt. Counting Induced Subgraphs: A Topological Approach to #W[1]-hardness. *CoRR*, abs/1807.01920, 2018. arXiv:1807.01920.

87 Alexander D. Scott and Gregory B. Sorkin. Linear-programming design and analysis of fast algorithms for Max 2-CSP. *Discrete Optimization*, 4(3-4):260–287, 2007. doi:10.1016/j.disopt.2007.08.001.

88 Richard P. Stanley. *Enumerative Combinatorics*, volume 1. Cambridge University Press, second edition, 2011.

89 Simon Straub, Thomas Thierauf, and Fabian Wagner. Counting the Number of Perfect Matchings in K_5-Free Graphs. In *IEEE 29th Conference on Computational Complexity, CCC 2014, Vancouver, BC, Canada, June 11-13, 2014*, pages 66–77, 2014. doi:10.1109/CCC.2014.15.

90 H. N. V. Temperley and Michael E. Fisher. Dimer problem in statistical mechanics - an exact result. *Philosophical Magazine*, 6(68):1478–6435, 1961.

91 Marc Thurley. Kernelizations for Parameterized Counting Problems. In *Theory and Applications of Models of Computation, 4th International Conference, TAMC 2007, Shanghai, China, May 22-25, 2007, Proceedings*, pages 703–714, 2007. doi:10.1007/978-3-540-72504-6_64.

92 Salil P. Vadhan. The Complexity of Counting in Sparse, Regular, and Planar Graphs. *SIAM J. Comput.*, 31(2):398–427, 2001. doi:10.1137/S0097539797321602.

93 Leslie G. Valiant. The complexity of computing the permanent. *Theoret. Comput. Sci.*, 8(2):189–201, 1979.

94 Leslie G. Valiant. Holographic Algorithms. *SIAM J. Comput.*, 37(5):1565–1594, 2008. doi:10.1137/070682575.

95 Johan M. M. van Rooij, Hans L. Bodlaender, and Peter Rossmanith. Dynamic Programming on Tree Decompositions Using Generalised Fast Subset Convolution. In *Algorithms - ESA 2009, 17th Annual European Symposium, Copenhagen, Denmark, September 7-9, 2009. Proceedings*, pages 566–577, 2009. doi:10.1007/978-3-642-04128-0_51.

96 Vijay V. Vazirani. NC algorithms for computing the number of perfect matchings in $K_{3,3}$-free graphs and related problems. *Inf. Comput.*, 80(2):152–164, 1989.

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

98 Mingji Xia, Peng Zhang, and Wenbo Zhao. Computational complexity of counting problems on 3-regular planar graphs. *Theoretical Computer Science*, 384(1):111–125, 2007. Theory and Applications of Models of Computation. doi:10.1016/j.tcs.2007.05.023.

99 Dmitriy Zhuk. A Proof of CSP Dichotomy Conjecture. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 331–342, 2017. doi:10.1109/FOCS.2017.38.