Counting Problems on Quantum Graphs

Parameterized and Exact Complexity Classifications

A dissertation submitted towards the degree

Doctor of Natural Sciences (Doctor rer. nat.)

of the Faculty of Mathematics and Computer Science

of Saarland University

Submitted by

Marc Roth

Saarbrücken, May 2019

UNIVERSITÄT
DES
SAARLANDES

DEPARTMENT OF COMPUTER SCIENCE

To Felix

Colloquium Information

Date:	July 15th, 2019 12:00 CEST Saarbrücken, Germany
Dean:	Prof. Dr. Sebastian Hack *Saarland University*
Chairman:	Prof. Dr. Gert Smolka *Saarland University*
Reviewers:	Prof. Dr. Holger Dell *IT University of Copenhagen*
	Prof. Dr. Markus Bläser *Saarland University*
	Prof. Dr. Fedor V. Fomin *University of Bergen*
Scientific Assistant:	Dr. Daniel Stan *Saarland University*

Abstract

Quantum graphs, as defined by Lovász in the late 60s, are formal linear combinations of simple graphs with finite support. They allow for the complexity analysis of the problem of computing finite linear combinations of homomorphism counts, the latter of which constitute the foundation of the structural hardness theory for parameterized counting problems: The framework of parameterized counting complexity was introduced by Flum and Grohe, and McCartin in 2002 and forms a hybrid between the classical field of computational counting as founded by Valiant in the late 70s and the paradigm of parameterized complexity theory due to Downey and Fellows which originated in the early 90s.

The problem of computing homomorphism numbers of quantum graphs subsumes general motif counting problems and the complexity theoretic implications have only turned out recently in a breakthrough regarding the parameterized subgraph counting problem by Curticapean, Dell and Marx in 2017.

We study the problems of counting partially injective and edge-injective homomorphisms, counting induced subgraphs, as well as counting answers to existential first-order queries. We establish novel combinatorial, algebraic and even topological properties of quantum graphs that allow us to provide exhaustive parameterized and exact complexity classifications, including necessary, sufficient and mostly explicit tractability criteria, for all of the previous problems.

Zusammenfassung

Diese Arbeit befasst sich mit der Komplexitätsanalyse von mathematischen Problemen die als Linearkombinationen von Graphhomomorphismenzahlen darstellbar sind. Dazu wird sich sogenannter Quantengraphen bedient, bei denen es sich um formale Linearkombinationen von Graphen handelt und welche von Lovász Ende der 60er eingeführt wurden.

Die Bestimmung der Komplexität solcher Probleme erfolgt unter dem von Flum, Grohe und McCartin im Jahre 2002 vorgestellten Paradigma der parametrisierten Zählkomplexitätstheorie, die als Hybrid der von Valiant Ende der 70er begründeten klassischen Zählkomplexitätstheorie und der von Downey und Fellows Anfang der 90er eingeführten parametrisierten Analyse zu verstehen ist.

Die Berechnung von Homomorphismenzahlen zwischen Quantengraphen und Graphen subsumiert im weitesten Sinne all jene Probleme, die das Zählen von kleinen Mustern in großen Strukturen erfordern. Aufbauend auf dem daraus resultierenden Durchbruch von Curticapean, Dell und Marx, das Subgraphzählproblem betreffend, behandelt diese Arbeit die Analyse der Probleme des Zählens von partiell- und kanteninjektiven Homomorphismen, induzierten Subgraphen, und Treffern von relationalen Datenbankabfragen die sich als existentielle Formeln ausdrücken lassen. Insbesondere werden dabei neue kombinatorische, algebraische und topologische Eigenschaften von Quantengraphen etabliert, die hinreichende, notwendige und meist explizite Kriterien für die Existenz effizienter Algorithmen liefern.

Acknowledgements

I am very grateful to Holger Dell for providing me with most valuable advise and the freedom to perform my own research during my PhD studies. I am especially thankful to Holger for introducing me to the counting complexity community in the course of the program “Counting Complexity and Phase Transitions” at the Simons Institute for the Theory of Computing in 2016. Moreover, I thank Markus Bläser and Radu Curticapean for introducing me to complexity theory, in particular, to parameterized counting problems. Both, Markus and Radu, have been reliable sources for fruitful discussions and helpful advise since my undergraduate studies at Saarland University. In addition to the above, I am very grateful to Holger, Markus, as well as to Fedor V. Fomin, for reviewing this thesis.

Further, I express my gratitude to Johannes Schmitt and Johannes Lengler for providing access to an oracle for advanced topics in algebra and number theory that have been necessary for my research, but exceeded the topics I encountered during my studies in computer science.

I thank Philip Wellnitz, Yannick Forster and Fabian Kunze for proof reading and Felix Rech for proof reading *and* for the design of the title page of this thesis.

I also thank Kathrin Stark for the daily ICoffee breaks and Cornelius Brand for many interesting discussions, often, but not always related to computer science. Moreover, I owe Cornelius for saying “Tutte Polynomial” every once in a while. Furthermore, I am very grateful to all my friends for helping me relax and having a good time during evenings, weekends and holidays. In particular, I am going to miss the weekly meetings at Phil’s place as well as at Al’Bacio, the latter of which provided us with delicious and cheap pizza since 2011.

I wish to thank my roommates Jan and Osman for being patient with me on the days I prioritized information theoretic entropy over the entropy in our living room.

Last but not least, I am very grateful to my family, especially to my parents, and to Felix for always supporting me.

Contents

Chapter 1

Introduction

Complexity theory is a branch of theoretical computer science and concerned with the analysis of the inherent difficulty of computational problems. The approach is rigorous and mathematical in nature and thus the results are essentially independent of current or future hardware. Abstractly speaking, a computational problem requires some kind of input and produces some kind of output (see Figure 1.1). Let us take a look at the following examples which will be the guides of this introduction:

k-SAT: **Input:** A boolean formula φ in k-CNF.

Output: $\begin{cases} 1 & \text{if } \varphi \text{ is satisfiable} \\ 0 & \text{otherwise} \end{cases}$

PerfMatch: **Input:** A simple graph G.

Output: $\begin{cases} 1 & \text{if } G \text{ is contains a perfect matching} \\ 0 & \text{otherwise} \end{cases}$

VC: **Input:** A simple graph G and a positive integer k.

Output: $\begin{cases} 1 & \text{if } G \text{ is contains a vertex cover of size } k \\ 0 & \text{otherwise} \end{cases}$

Clique: **Input:** A simple graph G and a positive integer k.

Output: $\begin{cases} 1 & \text{if } G \text{ is contains a clique of size } k \\ 0 & \text{otherwise} \end{cases}$

DomSet: **Input:** A simple graph G and a positive integer k.

Output: $\begin{cases} 1 & \text{if } G \text{ is contains a } k\text{-dominating set} \\ 0 & \text{otherwise} \end{cases}$

We encourage the reader unfamiliar with the graph-theoretic terminology in the above problems to consult Chapter 2 or e.g. [69, Chapter A.1.1 & A.9.1].

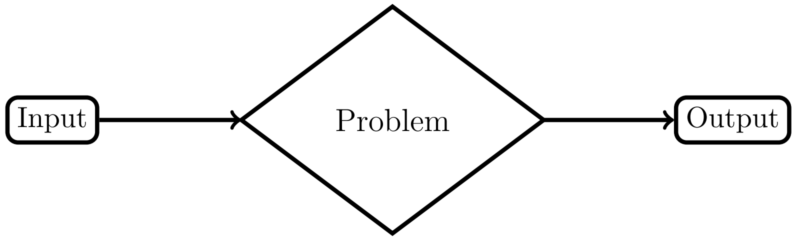


Figure 1.1: A computational problem.

All five of the previous problems have in common that they *decide* the existence of some kind of solution, e.g., a satisfying assignment for φ or a perfect matching in G. Such problems are hence called *decision problems*.

A computational problem is referred to as feasible, if it can be solved by an efficient algorithm. Classically, an algorithm is considered efficient if its worst-case running time is bounded by a polynomial in the input size; we refer the reader to [69, Chapter 1.3] and [71, Chapter 2.1] for a detailed discussion. From all of the five preceding examples, only PERFMATCH is known to admit a polynomial-time algorithm [61]. The other four are merely *conjectured to not allow* for a polynomial-time algorithm.

The latter is due to the fact that disproving the existence of an efficient algorithm for a computational problem seems to be an incredibly difficult task. Examples are rare and include the Time Hierarchy Theorem (cf. [71, Chapter 4]) and, in some sense, unconditional circuit lower bounds such as [135]. For this reason, one usually assumes some popular infeasibility conjecture such as $\mathsf{P} \neq \mathsf{NP}$, stating that the set P of decision problems that admit a polynomial-time algorithm is not equal to the set NP of decision problems whose solutions can be verified in time polynomial in the input size (cf. [71, Chapter 2.1.2]). This conjecture is the most fundamental open problem in complexity theory, perhaps even in computer science in general. We refer the reader to the standard textbook of Garey and Johnson [69] for an exposition of the topic.

Indeed, all of the previous problems except for PERFMATCH have been shown to be among the hardest problems in NP by Karp in his famous list of 21 NP-complete problems [86].[1] Consequently, none of the problems can be solved in polynomial time, unless $\mathsf{P} = \mathsf{NP}$ holds.

In recent years, potentially stronger assumptions than $\mathsf{P} \neq \mathsf{NP}$ arose with the goal of a more fine-grained analysis of the best possible running time for computational problems. In particular, one aims to discover whether existing algorithms that possibly rely on mere brute force, can be improved.

[1]To be precise, DOMSET is not contained in this list but rather its generalized version SETCOVER.

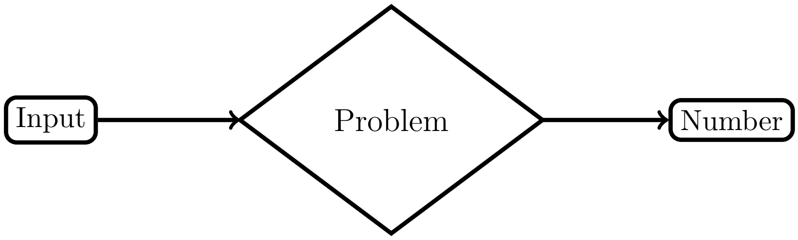


Figure 1.2: A counting problem.

Examples of such conjectures are the Exponential Time Hypothesis (ETH), stating that 3-SAT cannot be solved in subexponential time, as well as the Strong Exponential Time Hypothesis (SETH), intuitively stating that k-SAT cannot be solved faster than brute-force for $k \to \infty$. Those conjectures have been introduced by Calabro, Impagliazzo and Paturi [76, 26] and became popular in the context of fine-grained complexity theory; a formal introduction and exposition of some consequences of ETH and SETH is provided in Chapter 2.3.2. In case of CLIQUE, ETH rules out the existence of an algorithm running in time

$$f(k) \cdot n^{o(k)}$$

for any computable function f, where n is the number of vertices of the input graph G [33, 34]. However, CLIQUE can be solved faster than brute-force, i.e., asymptotically faster than n^k [105], while the same is not true for DOMSET, unless SETH is false [109].

1.1 Counting Problems

In this thesis, we consider so-called *counting problems* which, in contrast to decision problems, require to count solutions (see Figure 1.2). In fact, many decision problems canonically induce a counting problem. An example is given by #k-SAT, which counts the satisfying assignments of a given k-CNF formula. The counting versions #PERFMATCH, #VC, #CLIQUE and #DOMSET of the remaining problems are defined analogously. Now, similar to the complexity of decision problems, we aim to find efficient algorithms or prove that the existence of such algorithms would violate a widely-believed conjecture.

To begin with, let us provide a motivation for the analysis of counting problems. It turns out that, historically, the first researchers that studied such problems were statistical physicists. A prominent example of a counting problem that arises in statistical physics is the partition function of what is called the *dimer model* (cf. [87, 125, 88]).

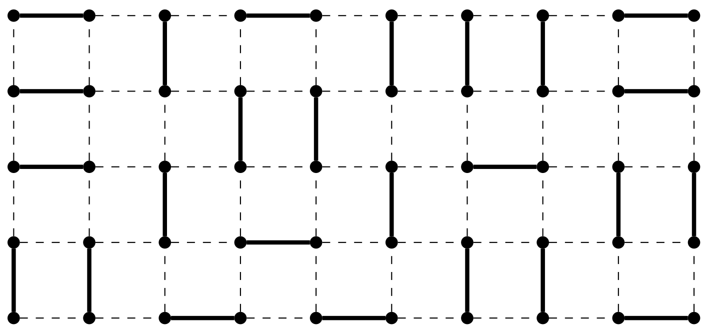


Figure 1.3: Illustration of an instance of the dimer problem. Included is a dimer cover or, equivalently, a perfect matching depicted by solid lines.

Very roughly speaking an instance of the dimer problem is a set of particles that are aligned in a grid-like structure such that every particle can only interact with its vertical and horizontal neighbors. The partition function of such an instance counts the number of *dimer covers* of these particles. A dimer cover is a partition of the set of particles into sets of size two, such that the particles of each pair can interact with each other. An illustration is given in Figure 1.3. The reader might have noticed that the dimer covers are precisely the perfect matchings if the input instance is viewed as a grid graph such that the particles are associated with the vertices, two of which are made adjacent if the corresponding particles can interact with each other. As a consequence, computing the partition function of the dimer model is equivalent to #PerfMatch if restricted to grid graphs as in Figure 1.3. The number of perfect matchings in the latter example is *precisely*

$$185921\,.$$

The fact that this number is efficiently computable, i.e., in polynomial time in the size of the problem instance, is due to the famous FKT-Algorithm, named after the statistical physicists Fisher, Kasteleyn and Temperley [87, 125, 88]. Their algorithm not only works for grid graphs as in Figure 1.3, but more generally for all *planar* graphs, i.e., graphs that can be drawn into the plane without crossing lines. This result was complemented by the seminal paper of Valiant [128], in which it was proved that the unrestricted version of #PerfMatch where arbitrary input graphs are allowed, cannot be solved in polynomial time unless $\mathsf{P} = \mathsf{NP}$.[2]

[2]In fact, Valiant proved #PerfMatch to be hard for the class $\#\mathsf{P}$, which should be seen as a counting equivalent of NP and which is likely to contain problems significantly harder than any problem in NP. Evidence for the latter claim is due to Toda [126] who proved that $\mathsf{PH} \subseteq \mathsf{P}^{\#\mathsf{P}}$, where PH is the Polynomial-Time-Hierarchy (cf. [71, Chapter F.1]).

The impact of Valiant's work was significant as #PerfMatch became the first (non-artificial) hard counting problem known to admit a polynomial-time algorithm for its decision version. From then on, the field of counting complexity evolved into a well-established branch of complexity theory. Unfortunately, it soon turned out that counting problems which are both, interesting and efficiently solvable, are rare. In particular, many efficiently solvable combinatorial decision problems turned out to be hard in their counting versions. Examples include counting of satisfying assignment of monotone 2-CNFs [129], counting independent sets in bipartite graphs [110], counting of s-t-paths [129] and counting of undirected Eulerian circuits [19]. Even worse, it has been shown that brute-force algorithms for hard counting problems often cannot be significantly improved upon, unless ETH fails [40, 47, 16]. However, let us give some credit to the following, quite surprising tractability results:

- The Holant-Framework yields polynomial-time algorithms for counting problems that can be reduced to the FKT-Algorithm by what is called holographic reductions [130, 22, 21, 20, 75, 23, 24, 7].

- Kirchhoff's theorem for counting spanning trees and its generalization to counting bases of regular matroids due to Maurer (cf. [99]).

- The "BEST"-Theorem[3] [131] for counting Eulerian circuits in *directed* graphs.

- The counting versions of Courcelle's theorem [38] and the Frick-Grohe-Theorem [66] for monadic second-order and first-order model counting on graphs with bounded treewidth and local treewidth, respectively (cf. [65, Theorem 14.8]).

Now, what can be done about counting problems that are known to be hard? Especially those that do not allow significant improvements over brute-force algorithms under ETH seem to enforce some kind of relaxation if efficient algorithms are desired. One possible relaxation is to restrict the problem input in some way and it turns out that we already encountered an example where input restriction yields a polynomial-time algorithm: Recall that #PerfMatch, while being hard in the general case, can be solved efficiently by the FKT-Algorithm if the input graphs are planar. In contrast, #PerfMatch stays hard if restricted to bipartite graphs [129]. Further examples include the restriction to 3-regular graphs [140] and more generally to regular or sparse graphs [127].

[3]De **B**ruijn, van Aardenne-**E**hrenfest, **S**mith, **T**utte

Another possible relaxation for hard counting problems is to require only an *approximation of the number of solutions.* Indeed, the field of approximate counting is a very active branch of counting complexity theory. We refer the reader to [83] and [9] for a detailed overview and only present some of the most striking results in what follows. A famous early result in the context of approximate counting is the approximation algorithm for #PerfMatch restricted to bipartite graphs due to Jerrum and Sinclair [82]. Furthermore, the complexity of approximate counting the answers to boolean constraint satisfaction problems has been classified by a trichotomy due to Dyer, Goldberg and Jerrum [59]. Very recently, Barvinok [9] introduced a new method for approximating partition functions by Taylor expansions, which was later refined by Patel and Regts [108] and then used by Guo et al. [72] to obtain an approximation algorithm for all but one non-negative Holant problem on cubic graphs. The missing case of the latter result is the problem of approximating the number of perfect matchings in a graph. While admitting an approximation algorithm if restricted to bipartite graphs, it is not known to admit one in case of cubic graphs (cf. [72]).

A third relaxation is a multivariate analysis of hard counting problems, which brings us to the field of parameterized counting complexity.

1.1.1 Parameterized Counting Complexity Theory

Let us assume we are given a counting problem that is intractable with respect to classical counting complexity. All of the five problems introduced in the beginning are examples of such. Let us take a closer look at #VC, #Clique and #DomSet. All three problems have in common that the input is a pair of a graph G and a positive integer k. Now let us think of k as an additional input parameter and consider the following question:

Is it possible to obtain an efficient algorithm if k is promised to be small?

In other words, we wish to understand the complexity of those problems not only in a single variable (the input size), but instead, we aim for a multivariate analysis, considering the number k as well. This paradigm, which is commonly referred to as *parameterized complexity analysis*, is due to Downey and Fellows and originated in the early 90s [54] (see also [55, 56]). By now, parameterized complexity theory is a well-established subfield of theoretical computer science. Excellent textbooks are [45, 65] and [57]. The first parameterized analysis of counting problems is due to Flum and Grohe [64] and, independently, due to McCartin [100].

Now let us formulate the question above mathematically. To this end, we assume that a counting problem comes with an additional *parameterization function* κ that maps an input x to a parameter $\kappa(x)$. In the previous examples we had that $\kappa(G, k) = k$ and hence it will be convenient to just write k for the parameter.

A parameterized counting problem is called *fixed-parameter tractable*, if it can be solved in time

$$f(k) \cdot \mathsf{poly}(n)\,,$$

where n is the input size[4] and f is some computable function. Note that this notion of tractability is weaker than the classical notion of polynomial-time feasibility in the sense that, while being polynomial in the input size, the running time may be arbitrarily bad in the parameter. Now, arguably, a running time of $\mathsf{a}(k,k)\cdot\mathsf{poly}(n)$, for a being the Ackermann function, should not count as efficient. However, it turns out that many fixed-parameter tractable (counting) problems admit a running time in which f is only single-exponential, often with a basis smaller than 2; we refer the reader to the introduction of Downey's and Fellows' textbook [57] for a detailed discussion. Let us take a look at the problem #VC of counting vertex covers of size k in a graph G with n vertices. A brute-force algorithm would iterate over all k-tuples of vertices in G and then verify whether each tuple constitutes a vertex cover, yielding a running time of $\Theta(n^{k+1})$. In contrast, Flum and Grohe [64] proved that the problem can be solved in time $O(2^k \cdot n^2)$, that is, #VC is fixed-parameter tractable. In particular, $O(2^k \cdot n^2)$ is significantly faster than $\Theta(n^{k+1})$; consult [57, Table 1] to get a feeling on *how much faster* the former is.

For #Clique and #DomSet we can also obtain brute-force algorithms running in time $n^{\Theta(k)}$. However, as we have seen, these problems are not fixed-parameter tractable, unless ETH fails. In fact, #Clique and #DomSet characterize the parameterized complexity classes #W[1] and #W[2]: A parameterized counting problem is contained in #W[1] if it is at most as hard as #Clique and it is contained in #W[2] if it is at most as hard as #DomSet. The hardest problems of #W[1] are not fixed-parameter tractable unless ETH fails. For this reason, the reader should consider a proof of hardness for #W[1] as evidence for (fixed-parameter) intractability of a parameterized problem, at least if they believe ETH to be correct. It is furthermore conjectured that #W[1] is a *proper* subset of #W[2]. However, the evidence for the latter claim is much weaker and discussed in Chapter 2.

Let us summarize the prior discussion and outline the strategy for a parameterized and exact complexity analysis: Given some parameterized counting problem P, we first wish to find out whether P can be solved in polynomial time. If not, we aim to find an algorithm that proves P to be fixed-parameter tractable. In case we are not able to find such an algorithm we try to prove that P is hard for #W[1] (or even #W[2]). Finally, we investigate whether a brute-force algorithm can be (significantly) improved without violating ETH or SETH.

[4]For graphs, we can equivalently choose n as the number of vertices.

A further goal in our complexity analysis is, if possible, the classification of entire families of problems. Let us be more precise. Given some set $\mathcal{P}$ of similar parameterized counting problems, we wish to establish a criterion of problems in $\mathcal{P}$ that, if satisfied, induces fixed-parameter tractability and yields a hardness result otherwise. By this, the set $\mathcal{P}$ is partitioned in feasible and (most likely) infeasible problems. Such binary classifications into easy and hard cases are also called "complexity dichotomies". Unfortunately, there are results like Ladner's Theorem [92] which disprove the existence of such dichotomies for all problems in NP. However, it turned out that "Ladner-like" theorems do not apply to *motif counting problems.* Roughly speaking, a motif counting problem is of the following form

Given a large structure and a small pattern called the motif, compute the number of occurrences of the motif in the structure.

In the framework of parameterized counting, the size of the motif seems to be the canonical parameterization. An example of a motif counting problem is #CLIQUE: Here, the large structure is the input graph and the small motif is the complete graph of size k.

A further fundamental parameterized motif counting problem is the problem of counting graph homomorphisms. Formally, the problem

$$\#\textsc{Hom}(\mathcal{H})$$

asks, given a graph $H \in \mathcal{H}$ and an arbitrary graph G, to compute the number of graph homomorphisms from H to G. Intuitively, a graph homomorphism from H to G is an edge-preserving mapping from the vertices of H to the vertices of G; the formal definition is given in Chapter 2. The problem $\#\textsc{Hom}(\mathcal{H})$ is parameterized by the number of vertices of H, which should be viewed as the size of the small motif. In particular, we point out that the class of graphs $\mathcal{H}$ is *not* part of the input, but rather part of the problem definition. Consequently, we can view #HOM as a family of motif counting problems where each problem in the family is given by $\#\textsc{Hom}(\mathcal{H})$ for some class of graphs $\mathcal{H}$. The following complexity classification is due to Dalmau and Jonsson [46]:

Theorem 1.1 (Section 3 in [46]). *Let $\mathcal{H}$ be a class of graphs.*[5]

1. *If $\mathcal{H}$ has bounded treewidth, then $\#\textsc{Hom}(\mathcal{H})$ is fixed-parameter tractable.*

2. *Otherwise, $\#\textsc{Hom}(\mathcal{H})$ is hard for $\#\mathsf{W}[1]$.*

Here, *treewidth* is a structural graph parameter that captures resemblance with a tree; the formal definition is given in Chapter 2.2.1.

[5]To be precise, $\mathcal{H}$ needs to be recursively enumerable to guarantee computability of the reduction.

In other words, assuming ETH holds, Theorem 1.1 establishes low treewidth as a necessary and sufficient criterion for fixed-parameter tractability of graph homomorphism counting and thus constitutes a complete complexity dichotomy.

One of the central questions in parameterized counting complexity is whether and in how far the dichotomy for counting homomorphisms can be used to obtain similar classifications for other parameterized motif counting problems. Curticapean, Dell and Marx gave a surprisingly clean answer in their breakthrough result on the subgraph counting problem [41]:

Theorem 1.1 applies to all parameterized motif counting problems that can be expressed as a linear combination of homomorphism numbers.

Their work made implicit use of objects called quantum graphs which had been originally introduced by Lovász in the 60s (cf. [95]) and should not be confused with the equally named objects in physics [90].

1.1.2 Quantum Graphs

A *quantum graph* Q is a formal linear combination of graphs with finite support, i.e., with only finitely many non-zero coefficients. We write

$$Q = \sum_H \lambda_H \cdot H\,,$$

where the sum is over all simple and finite graphs. Graph parameters are extended linearly to quantum graphs. Let, for example, $\#\mathsf{Hom}(H \to \star)$ be the function mapping a graph G to the number of homomorphisms $\#\mathsf{Hom}(H \to G)$ from H to G. Then

$$\#\mathsf{Hom}(Q \to G) := \sum_H \lambda_H \cdot \#\mathsf{Hom}(H \to G)\,.$$

Given a class $\mathcal{Q}$ of quantum graphs, the problem $\#\textsc{Hom}(\mathcal{Q})$ is then defined analogously to $\#\textsc{Hom}(\mathcal{H})$, that is, given a quantum graph $Q \in \mathcal{Q}$ and an arbitrary graph G, the task is to compute $\#\mathsf{Hom}(Q \to G)$. The parameter is given by the description length of Q.

In their work [41], Curticapean, Dell and Marx established what they called *complexity monotonicity* of counting homomorphisms from quantum graphs, informally stating the following:

Computing $\#\mathsf{Hom}(Q \to \star)$ is ***precisely*** *as hard as computing the numbers $\#\mathsf{Hom}(H \to \star)$ for all H with $\lambda_H \neq 0$.*

In particular, they proved that $\#\textsc{Hom}(\mathcal{Q})$ is fixed-parameter tractable if and only if $\#\textsc{Hom}(\mathcal{H}(\mathcal{Q}))$ is. Here, $\mathcal{H}(\mathcal{Q})$ is the set of all graphs H that occur with a non-zero coefficient in some quantum graph in $\mathcal{Q}$.

As the complexity of $\#\textsc{Hom}(\mathcal{H})$ is fully understood by Theorem 1.1, complexity monotonicity induces the following strategy for the analysis of parameterized motif counting problems.

Given a parameterized motif counting problem P,

(1) find a family $\mathcal{Q}$ of quantum graphs such that P and $\#\textsc{Hom}(\mathcal{Q})$ are identical, and

(2) understand the coefficients of quantum graphs in $\mathcal{Q}$.

Perhaps surprisingly, it turned out that (2) seems to be *much harder* than (1). In their paper [41], Curticapean, Dell and Marx applied the previously outlined strategy to the parameterized subgraph counting problem and by this established both, an improved algorithm for the fixed-parameter tractable cases and an easy hardness proof for the remaining cases, simplifying the previous classification [43] significantly. We provide an in-depth treatment of their work, using quantum graphs explicitly, in Chapter 3.

In this work, we further develop the method of counting homomorphisms from quantum graphs and obtain complete complexity classifications, mostly including explicit criteria for (fixed-parameter) tractability, for various motif counting problems outlined in the next section. In particular, we discover combinatorial, algebraic and even topological interpretations of coefficients of quantum graphs that model our problems. As a consequence, we will be able to rely on deep results from enumerative combinatorics to analyze the complexity of parameterized (motif) counting problems.

1.2 Overview, Contributions and Techniques

Our work brings together questions and techniques from a wide variety of areas, such as parameterized and fine-grained complexity, logics, database theory, matroid theory, lattice theory, graph minor theory, and the theory of transformation groups. The interested reader should not be alarmed, however, as we put considerable effort into making the presentation as self-contained as possible. We present a detailed exposition of all the required background material in Chapter 2. In particular, we include a complete, partially modified proof of Theorem 1.1 in Chapter 2.5. By this, we not only avoid relying on the classification of $\#\textsc{Hom}(\mathcal{H})$ in a black-box manner, but also, our proof illustrates most of the combinatorial and algebraic techniques introduced in the previous sections of Chapter 2.

Chapter 3 provides an introduction to quantum graphs, including the formal statement and a proof of the complexity monotonicity property. In particular, we will first illustrate the latter principle for the concrete example of counting matchings of size 3. More precisely, we show that the motif counting problems of counting 3-matchings and triangles are not only interreducible but essentially equal. As a further, more elaborate example, we present a proof of the complexity classification of the parameterized subgraph counting problem due to Curticapean, Dell and Marx [41], using the framework of quantum graphs *explicitly*.

Furthermore, we modify the original proof in such a way that it relies on Rota's NBC Theorem [118], the latter of which we will encounter as a crucial proof ingredient multiple times in this thesis.

From Chapter 4 on, we present the novel contributions obtained in this thesis. We begin with a parameterized complexity analysis of the problems of counting partially injective and edge-injective homomorphisms. Both notions are natural interpolations between homomorphisms and subgraph embeddings. More precisely, given a graph H, a set of *inequalities* I over the vertices of H, and a graph G, we write $\mathsf{PartInj}(H, I \to G)$ for the set of all homomorphisms h from H to G that satisfy all inequalities in I, i.e., $h(u) \neq h(v)$ whenever "$u \neq v$" $\in I$. Furthermore, we write $\mathsf{EdgeInj}(H \to G)$ for the set of all homomorphisms h from H to G that are injective on the edges of H, i.e., $h(e) \neq h(\hat{e})$ whenever $e \neq \hat{e}$ are edges of H. It is then shown that there are quantum graphs $Q_1[H, I]$ and $Q_2[H]$ such that for every graph G we have

$$\#\mathsf{PartInj}(H, I \to G) = \#\mathsf{Hom}(Q_1[H, I] \to G)\,,$$

and

$$\#\mathsf{EdgeInj}(H \to G) = \#\mathsf{Hom}(Q_2[H] \to G)\,.$$

The construction of $Q_1[H, I]$ is done in Chapter 4.1. Furthermore, we show that the graphs with a non-zero coefficient in $Q_1[H, I]$, called *constituents*, are precisely those graphs that are obtained from H by identifying vertices along inequalities I without creating self-loops. More formally, our proof relies on an application of the Möbius inversion formula over the lattice of flats of the graphic matroid induced by the inequalities I. In particular, we rely on Rota's NBC Theorem [118] to show that Möbius inversion does not induce non-trivial cancellations of constituents of $Q_1[H, I]$.

Having an explicit representation of the constituents of the quantum graph, we then invoke complexity monotonicity and obtain a complete and explicit parameterized complexity dichotomy for the problem of counting partially injective homomorphisms. As an exemplary application, we invoke the latter classification to completely classify the parameterized complexity of counting locally injective homomorphisms.

The construction of $Q_2[H]$ is done in Chapter 4.2. While the construction itself is much easier, we will unfortunately be unable to provide an explicit criterion for the full classification of counting edge-injective homomorphisms. However, using a Ramsey argument, an explicit classification is obtained for hereditary classes of graphs, as well as for the specific cases of counting edge-disjoint cycles and paths, the latter of which are shown to be hard.

We turn to counting *induced* subgraphs in Chapter 5. More precisely, we consider the problem $\#\textsc{IndSub}(\Phi)$ of, given a graph G and a positive integer k, computing the number of induced subgraphs with k vertices in G that satisfy the graph property Φ.

Similar to #HOM($\mathcal{H}$), the property Φ is not part of the input, but part of the problem definition and, when parameterized by k, we wish to understand the parameterized complexity of #INDSUB(Φ) with respect to Φ. By this we build upon the work of Chen, Thurley and Weyer [35], as well as Jerrum and Meeks [79, 80, 101, 81]. For a given pair of a graph property Φ and a positive integer k, we construct a quantum graph $Q[\Phi, k]$ such that the number of induced subgraphs of size k in a graph G that satisfy Φ is precisely

$$\#\mathsf{Hom}(Q[\Phi, k] \to G)\,.$$

Chapter 5.1 establishes a connection between the coefficient of the complete graph of size k in $Q[\Phi, k]$ and Karp's evasiveness conjecture (cf. [102]). In particular, we prove that this coefficient is, up to a factor of $\pm k!$, equal to what is called the reduced Euler characteristic of the simplicial graph complex induced by Φ and k. Inspired by the "topological approach to evasiveness" due to Kahn, Saks and Sturtevant [85], this allows us to rely on fixed-point theorems for the topological interpretation of graph complexes. Thus we obtain an almost exhaustive and explicit complexity classification of #INDSUB(Φ) for properties Φ that are closed under the removal of edges.

We strengthen the previous approach in Chapter 5.2 for properties Φ that are non-trivial on edge-transitive graphs with a prime-power number of edges. Using Sylow's theorems, this allows us to drop the "almost" in the classification of #INDSUB(Φ) if the input graphs are restricted to be bipartite. More precisely, we prove that, assuming ETH, #INDSUB(Φ) is not fixed-parameter tractable whenever Φ is a monotone property and non-trivial on bipartite graphs.

Finally, Chapter 6 generalizes the notions of quantum graphs as well as complexity monotonicity to existential and universal first-order formulas. Building up on the work of Chen, Durand and Mengel [58, 31] we first provide a full classification of the problem of counting answers to conjunctive queries, the latter of which can also be expressed as partial homomorphisms between logical structures. In particular, and in sharp contrast to previous chapters, we will encounter instances that are at least as hard as #DOMSET and, most likely, even harder. This requires to consider the parameterized complexity classes #W[2] and #A[2], which we define via first-order model counting problems (cf. [65, Chapter 14]).

Thereafter, complexity monotonicity is established for counting partial homomorphisms from quantum queries; they are linear combinations of conjunctive queries and constitute the canonical generalization of quantum graphs. This allows us to lift the classification for conjunctive queries to more general classes of queries, namely existential and universal positive queries that may contain inequalities and non-monotone constraints over free variables.[6]

[6]A weaker version of the second step was established by Chen and Mengel [32].

Chapter 2

Preliminaries

2.1 Mathematical Notations

Given a finite set A, we write $\#A$ or $|A|$ for its cardinality and given a natural number $k \in \mathbb{N}$, we write $[k]$ for the set $\{0, \dots, k-1\}$. Adopting this notation, we write $(x_i)_{i \in [k]}$ for the tuple $(x_0, \dots, x_{k-1})$ and $\{x_i\}_{i \in [k]}$ for the set $\{x_0, \dots, x_{k-1}\}$. Given sets A and B and a function $f : A \to B$, we write $\mathsf{supp}(f) = f^{-1}(B \setminus \{0\})$ for the *support* of f. Furthermore, given a subset $S \subseteq A$, we write $f|_S$ for the restriction of f to S. Let M be an $s \times t$ matrix and let N be a $u \times v$ matrix. The *Kronecker product* (or *Tensor product*) of M and N, denoted by $M \otimes N$, is the $su \times tv$ matrix given by

$$M \otimes N := \begin{pmatrix} M_{1,1}N & M_{1,2}N & \dots & M_{1,t}N \\ M_{2,1}N & M_{2,2}N & \dots & M_{2,t}N \\ \vdots & \vdots & \dots & \vdots \\ M_{s,1}N & M_{s,2}N & \dots & M_{s,t}N \end{pmatrix}.$$

If M and N are quadratic matrices then $M \otimes N$ is invertible if and only if M and N are invertible.

2.2 Graphs

Graphs in this work are considered simple, undirected and without self-loops, unless stated otherwise. More precisely, a graph G is a pair of a finite set $V(G)$ of vertices and a symmetric and irreflexive relation $E(G) \subseteq V(G)^2$. We might only write V and E for vertices and edges if the graph is clear from the context. Furthermore we write $\{u, v\}$ to denote edges, emphasizing undirectedness. If u and v are vertices of G and $e = \{u, v\}$ is an edge of G we call u and v *adjacent* and say that u and v are *incident* to e. Given a vertex v of G we write $N_G(v)$ for the subset of vertices of G that are adjacent to v; if G is clear from the context, we just write $N(v)$. Throughout the thesis, we use n for the number of vertices and m for the number of edges.

Clique	Biclique	Path	Cycle	Matching
K_k	$K_{t,t}$	P_k	C_k	M_k

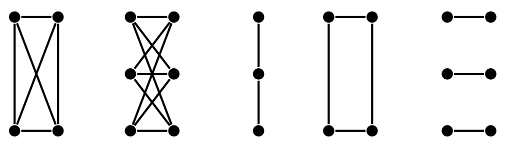


Figure 2.1: Five important classes of graphs we will encounter in this thesis. Depicted are K_4, $K_{3,3}$, P_2, C_4 and M_3. Note that cliques and bicliques are also referred to as "complete graphs" and "complete bipartite graphs".

Given a graph G with vertices $v_1, \ldots, v_n$, its *adjacency matrix* $A(G)$ is of size $n \times n$ and defined as follows

$$A(G)_{i,j} := \begin{cases} 1 & \text{if } \{v_i, v_j\} \in E \\ 0 & \text{if } \{v_i, v_j\} \notin E \end{cases} .$$

We will assume that graphs are encoded by their adjacency matrices.

Given two graphs H and G, the *Tensor product* of H and G, denoted by $H \times G$, is the graph with vertices $V(H) \times V(G)$ satisfying that two vertices (h_1, g_1) and (h_2, g_2) are adjacent if and only if $\{h_1, h_2\} \in E(H)$ and $\{g_1, g_2\} \in E(G)$. Its name stems from the fact that the adjacency matrix of $H \times G$ is given by $A(H) \otimes A(G)$, up to permutation of rows and columns.

If a graph H is obtained from G by deleting a set of edges and a set of vertices of G, including incident edges, then H is called a *subgraph* of G. In case only edges are deleted, we speak of an *edge-subgraph*. Given a subset $\hat{V}$ of $V(G)$ we write $G[\hat{V}]$ for the graph with vertices $\hat{V}$ and edges $E \cap \hat{V}^2$. The resulting graph is called an *induced subgraph* of G. Furthermore, given a subset $\hat{E}$ of $E(G)$ we write $G[\hat{E}]$ for the edge-subgraph $(V(G), \hat{E})$ of G. Note that, in particular, we do not delete isolated vertices in the latter construction.

A *path* of a graph is a sequence of pairwise distinct vertices $v_1, \ldots, v_k$ such that v_i is adjacent to v_{i+1} for all $i = 1, \ldots, k-1$. It is called a *cycle* if additionally $k > 2$ and v_k is adjacent to v_1. Figure 2.1 depicts some further examples of important graph classes.

A graph is *connected* if for every pair of vertices u and v there is a path starting at u and ending at v. If a graph does not contain any cycles we call it *acyclic*; if it is both, acyclic and connected, we call it a *tree*.

2.2.1 Treewidth and Graph Minors

The notion of treewidth is, arguably, one of the most heavily studied graph parameters in the last 30 years with fundamental applications in computational complexity and structural graph theory. While the definition we are about to see is due to Robertson and Seymour [113, 114], equivalent definitions have been introduced before by Halin [73] and Bertelè and Brioschi [10]. We recommend Chapt. 7 of [45] for a comprehensive exposition of treewidth, which we will follow hereinafter.

Intuitively, the treewidth of a graph measures how tree-like it is. In particular, graphs with small treewidth allow a decomposition in small separators. These decompositions are the basis of many efficient dynamic programming algorithms (DPs) for problems that are computationally infeasible on graphs without the promise of having small treewidth. Roughly speaking, those algorithms can be seen as generalizations of known and simple DPs that solve the corresponding problems on trees. Although we will mainly need the concept in a black-box manner, especially in form of consequences for graph minor theory, we decided give the formal definition for reasons of self-containment.

Definition 2.1 (See e.g. Chapt. 7.2 in [45]). Let G be a graph. A *tree decomposition* of G consists of a tree T and a collection of subsets $B_t \subseteq V(G)$ for all $t \in V(T)$ called *bags* such that

(1) $\bigcup_{t\in V(T)} B_t = V(G)$,

(2) for every edge $\{u, v\} \in E(G)$ there exists a vertex t of T such that $\{u, v\} \subseteq B_t$, and

(3) for every vertex $v \in V(G)$ the graph
$$T\big[\{t \in V(T) \mid v \in B_t\}\big]$$
is connected.

The *width* of a tree decomposition is $\max_{t\in V(T)} |B_t| - 1$ and the *treewidth* of a graph G, denoted by $\mathsf{tw}(G)$, is the minimum width a tree decomposition of G can have. We say that a class of graphs has *bounded treewidth* if there exists a constant c such that the treewidth of every graph in the class is at most c.

Example 2.2. Every tree has treewidth 1; a corresponding tree decomposition can be constructed as shown in Figure 2.2. Every cycle has treewidth 2: Just delete an arbitrary vertex, construct a tree decomposition of width one as in case of trees and then add the deleted vertex to every bag.

On the other hand, the clique on k vertices has treewidth $k-1$. It is not hard to show that one cannot do better than taking as tree decomposition a single bag containing all vertices.

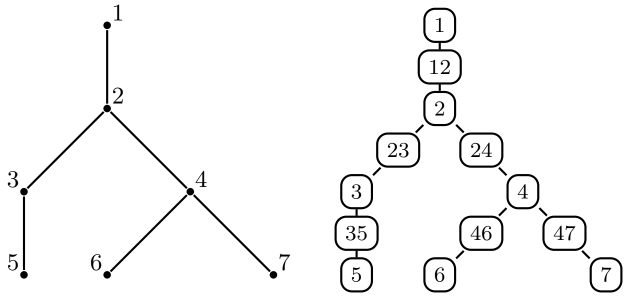


Figure 2.2: A tree decomposition of width 1 of a tree.

As we have seen, small treewidth captures, in some sense, resemblance with a tree. But what can be said about graphs with large treewidth such as the clique? A fundamental theorem of Robertson and Seymour [115] states that such graphs have a grid-like structure. Formalizing this result requires us to introduce the notion of a graph minor.

Definition 2.3 (See e.g. Chapt. 13.2 in [65]). Let G and H be graphs. A *minor mapping* from H to G is a function η that maps vertices v of H to vertex subsets $\eta(v) \subseteq V(G)$ such that the following constraints are satisfied:

(1) For every $v \in V(H)$ the graph $G[\eta(v)]$ is connected and nonempty.

(2) For all $u, v \in V(H)$ with $u \neq v$ the sets $\eta(u)$ and $\eta(v)$ are disjoint.

(3) For all edges $\{u, v\} \in E(H)$ there exist $\hat{u} \in \eta(u)$ and $\hat{v} \in \eta(v)$ such that $\{\hat{u}, \hat{v}\} \in E(G)$.

H is called a *minor* of G if there exists a minor mapping from H to G.

An equivalent way of defining minors of a graph is by vertex and edge deletions and edge contractions. Given a graph G and an edge $e = \{u, v\}$ of G, the *contraction* of e yields the graph G/e which is obtained from G by deleting e, u and v, and adding a new vertex uv that is made adjacent to all vertices $w \in V(G) \setminus \{u, v\}$ satisfying that $\{u, w\}$ or $\{v, w\}$ have been an edge of G. The graphs obtained from G by deleting a vertex v or an edge e are denoted by $G - v$ and $G - e$, respectively.

Observation 2.4. *A graph H is a minor of a graph G if and only if H can be obtained from G by successively applying vertex and edge deletions and edge contractions.*

Characterizing large treewidth in terms of minors requires the notion of *grid graphs* which are defined as follows; an illustration is given in Figure 2.3.

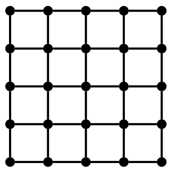


Figure 2.3: Illustration of $\boxplus_5$, that is, the grid graph of size 5×5.

Definition 2.5. The $(k \times k)$*-grid*, denoted by $\boxplus_k$, contains vertices

$$V = \{(i,j) \mid i,j \in [k]\}$$

and an edge between every pair of vertices $(i,j),(i',j')$ that satisfy

$$|i - i'| + |j - j'| = 1\,.$$

We might abuse notation and just write "k-grid" instead of "$(k \times k)$-grid".

Now we have everything we need to state the *Excluded-Grid-Theorem.*

Theorem 2.6 (Robertson and Seymour [115]). *There is a computable function f such that every graph with treewidth at least $f(k)$ contains $\boxplus_k$ as a minor.*

In a line of research, Robertson, Seymour and Thomas [116], Reed [111], Diestel et al. [51], Kawarabayashi and Kobayashi [89], Leaf and Seymour [94], Chekuri and Chuzhoy [29], Chuzhoy [36] and Chuzhoy and Tan [37] improved the upper bound on f from something worse than a tower of exponentials to a polynomial, culminating in the following theorem.

Theorem 2.7 (Chuzhoy and Tan [37]). *There is a computable function $f \in O(k^9 \cdot \mathsf{polylog}(k))$ such that every graph with treewidth at least $f(k)$ contains $\boxplus_k$ as a minor.*

As the treewidth of a clique of size k is precisely $k-1$ and as its largest grid minor is $\boxplus_{\Omega(\sqrt{k})}$, it can easily be concluded that $\Omega(k^2)$ is a lower bound for the function f in the Excluded-Grid-Theorem. The best known lower bound is $\Omega(k^2 \cdot \mathsf{log}(k))$ and due to Robertson, Seymour and Thomas [116].

2.2.2 Homomorphisms, Embeddings and Induced Subgraphs

A *homomorphism* from a graph H to a graph G is a mapping

$$h : V(H) \to V(G)$$

that preserves edges. In other words, for every edge $\{u,v\} \in E(H)$ it holds that $\{h(u),h(v)\} \in E(G)$. If $G = H$ then h is called an *endomorphism.*

An *embedding* is an injective homomorphism and a *strong embedding* from H to G is an embedding h such that for all pairs u and v of vertices of H we have that

$$\{u, v\} \in E(H) \Leftrightarrow \{h(u), h(v)\} \in E(G)\,.$$

A bijective strong embedding is called an *isomorphism* and we say that two graphs H and G are *isomorphic*, denoted by $H \simeq G$, if there exists an isomorphism from H to G.[1] An isomorphism from a graph to itself is called an *automorphism*. The set of automorphisms of a graph, together with functional composition constitutes a group, called the *automorphism group* of a graph. Slightly abusing notation we will write $\mathsf{Aut}(H)$ for both, the set of automorphisms of a graph H as well as for its automorphism group. We will furthermore use the following notations for sets of homomorphisms and its variants.

Definition 2.8. Let H and G be graphs.

- $\mathsf{Hom}(H \to G)$ denotes the set of homomorphisms from H to G.
- $\mathsf{Emb}(H \to G)$ denotes the set of embeddings from H to G.
- $\mathsf{Sub}(H \to G)$ denotes the set of subgraphs of G isomorphic to H.
- $\mathsf{StrEmb}(H \to G)$ denotes the set of strong embeddings from H to G.
- $\mathsf{IndSub}(H \to G)$ denotes the set of induced subgraphs of G isomorphic to H.

It will be very convenient to write $\#\mathsf{Hom}(H \to \star)$ for the function that maps a graph G to the number $\#\mathsf{Hom}(H \to G)$. The functions

$$\#\mathsf{Emb}(H \to \star), \#\mathsf{Sub}(H \to \star), \#\mathsf{StrEmb}(H \to \star) \text{ and } \#\mathsf{IndSub}(H \to \star)$$

are defined likewise.

Observe that the sets $\mathsf{Emb}(H \to G)$ and $\mathsf{StrEmb}(H \to G)$ can both be partitioned by the images of their elements. Furthermore, all classes of the induced equivalence relations have size $\#\mathsf{Aut}(H)$. As the images of the embeddings from H to G are precisely the subgraphs of G isomorphic to H and, similarly, the images of strong embeddings correspond to induced subgraphs, we obtain the following.

Fact 2.9. *Let H be a graph. We have that*

$$\#\mathsf{Emb}(H \to \star) = \#\mathsf{Aut}(H) \cdot \#\mathsf{Sub}(H \to \star)\,,$$

and

$$\#\mathsf{StrEmb}(H \to \star) = \#\mathsf{Aut}(H) \cdot \#\mathsf{IndSub}(H \to \star)\,.$$

[1]The notion of isomorphic graphs has already been used implicitly when we spoke about *the* complete graph on k vertices or *the* cycle of length three. Since we do not care about the elements of the vertex set, but only about the structure of the graph as given by its adjacency matrix, we will continue to distinguish graphs only by their isomorphism classes.

2.3 Computational Counting

We assume familiarity with the basics of computational complexity theory as provided for example in the first two chapters of the standard textbook of Goldreich [71]. A *counting problem* is a function $P : \Sigma^* \to \mathbb{N}$. Here Σ denotes a fixed finite alphabet which the reader can assume to be $\{0, 1\}$ for all purposes in this work.

2.3.1 Parameterized Counting Complexity

The field of parameterized counting was introduced by Flum and Grohe [64] and by McCartin [100] with the goal of adapting the relaxation of parameterization for decision problems to the counting realm. A *parameterized counting problem* is a pair (P, κ) of a counting problem and a computable parameterization $\kappa : \Sigma^* \to \mathbb{N}$. Let us take a look at some examples.

#Clique **Input:** A graph G and a positive integer k.
Parameter: $\kappa(G, k) := k$.
Output: The number of cliques of size k in G.

#VC **Input:** A graph G and a positive integer k.
Parameter: $\kappa(G, k) := k$.
Output: The number of vertex covers of size k in G, i.e., the number of subsets $S \subseteq V(G)$ with $|S| = k$ such that every edge of G is incident to a vertex in S.

#DomSet **Input:** A graph G and a positive integer k.
Parameter: $\kappa(G, k) := k$.
Output: The number of dominating sets of size k in G, i.e., the number of subsets $D \subseteq V(G)$ with $|D| = k$ such that every vertex in $V(G) \setminus D$ is adjacent to a vertex in D.

#Hom($\mathcal{H}$) **Input:** A graph G and a graph $H \in \mathcal{H}$.
Parameter: $\kappa(G, H) := |V(H)|$.
Output: $\#\mathsf{Hom}(H \to G)$, i.e., the number of homomorphisms from H to G.

Note that in the last example, $\mathcal{H}$ is a fixed, possibly infinite set of graphs that is *not* part of the input. If we choose $\mathcal{H}$ as the set of all complete graphs, then #Hom($\mathcal{H}$) and #Clique are equivalent up to a factor of $k!$, because the image of a homomorphism from a clique of size k is itself a clique of size k. Let us furthermore remark that we will omit defining κ explicitly from now on as it will be clear from the context.

While, classically, a problem is considered to be feasible if it can be solved by a polynomial-time algorithm, parameterization allows to relax the condition of tractability as follows.

Definition 2.10. A parameterized counting problem (P, κ) is called *fixed-parameter tractable (FPT)* if there exists a deterministic algorithm $\mathbb{A}$ that, on input x, computes $P(x)$ in time $f(\kappa(x)) \cdot |x|^{O(1)}$ for some computable function f independent of x. In particular, $\mathbb{A}$ is called an *FPT algorithm.*

Consider for example the problem #VC, which is NP-hard[2] [86] and therefore deemed infeasible in the classical sense. However, it is known that it can be solved in time $O(2^k \cdot n)$ [64], which is not a polynomial but should still be considered feasible if k is assumed to be much smaller than n. On the other hand, the problems #Clique and #DomSet are most likely not fixed-parameter tractable as shown in Section 2.3.2. First of all, we start by introducing a notion of reducibility and two of the most important complexity classes for parameterized counting problems.

Definition 2.11. Let (P, κ) and $(\hat{P}, \hat{\kappa})$ be parameterized counting problems. A *parameterized Turing reduction* from (P, κ) to $(\hat{P}, \hat{\kappa})$ is an FPT algorithm $\mathbb{A}$ that solves (P, κ) and is given oracle access to $(\hat{P}, \hat{\kappa})$. Furthermore, every oracle query y that is posed by $\mathbb{A}$ on input x must satisfy

$$\hat{\kappa}(y) \leq g(\kappa(x)) \tag{2.1}$$

for some computable function g independent of x.
We write $(P, \kappa) \leq^{\mathrm{T}}_{\mathrm{fpt}} (\hat{P}, \hat{\kappa})$ if a parameterized Turing reduction from (P, κ) to $(\hat{P}, \hat{\kappa})$ exists.

While the first part of the definition of a parameterized Turing reduction is most similar to the notion of a classical Turing or Cook reduction (see e.g. Definition 2.9 in [71]), one might wonder about condition (2.1) regarding the parameter of the oracle queries. However, this additional restriction is crucial for the reduction to guarantee that the problem that is reduced from is at least as hard as the problem that is reduced to with respect to fixed-parameter tractability. This is formalized in the following lemma, the proof of which is similar to the decision version as found in e.g. [65] as Exercise 2.9. We include it for completeness.

Lemma 2.12. *Let (P, κ) and $(\hat{P}, \hat{\kappa})$ be parameterized counting problems such that $(\hat{P}, \hat{\kappa})$ is fixed-parameter tractable and $(P, \kappa) \leq^{\mathrm{T}}_{\mathrm{fpt}} (\hat{P}, \hat{\kappa})$. Then (P, κ) is fixed-parameter tractable as well.*

Proof. Let $\mathbb{A}$ be the parameterized Turing reduction from (P, κ) to $(\hat{P}, \hat{\kappa})$ and let $\hat{\mathbb{A}}$ be an FPT algorithm that solves $(\hat{P}, \hat{\kappa})$. An FPT algorithm for (P, κ) can be constructed by executing $\mathbb{A}$ and simulating every oracle query y by $\hat{\mathbb{A}}$. Now (2.1) guarantees that, on input x, the parameter of y is

[2]In fact, Karp [86] proved the decision version of #VC to be NP-hard. However, if the number of solutions is known, it is certainly known whether a solution exists.

bounded by $g(\kappa(x))$. As $\mathbb{A}$ and $\hat{\mathbb{A}}$ are FPT algorithms, we obtain an overall running time bound of

$$f(\kappa(x)) \cdot |x|^c \cdot \hat{f}(g(\kappa(x))) \cdot |y|^c$$

for some constant c independent of x. In particular, the size of y is bounded by the running time of $\mathbb{A}$, i.e., by $f(\kappa(x)) \cdot |x|^c$. Hence the total running time is bounded by

$$f(\kappa(x))^{c+1} \hat{f}(g(\kappa(x))) \cdot |x|^{c+c^2} ,$$

which proves fixed-parameter tractability. ■

Having obtained a notion of parameterized reductions, we proceed with the introduction of the parameterized complexity class #W[1], which should be seen as a parameterized counting equivalent of NP. For historical reasons, containment in #W[1] was defined using what is called weighted "weft one" satisfiability — see for example [64, 100] and Chapt. 14 in [65]. In particular, the definition requires the notion of parameterized *parsimonious* reductions, which can informally be described as parameterized Turing reductions that allow only a single oracle call at the very end of the computation. As it will be crucial for most of our purposes to pose multiple oracle queries, we decided to define #W[1] in terms of parameterized Turing reductions and the problem #CLIQUE, which is known to be #W[1]-complete [64].

Definition 2.13. A parameterized counting problem (P, κ) is called

- #W[1]-*hard* if #CLIQUE $\leq^{\mathrm{T}}_{\mathrm{fpt}}$ (P, κ),
- #W[1]-*easy* if $(P, \kappa) \leq^{\mathrm{T}}_{\mathrm{fpt}}$ #CLIQUE, and
- #W[1]-*equivalent* if it is both, #W[1]-hard and #W[1]-easy.

We emphasize the following two facts about #W[1], the former of which is immediate and the latter of which is given by Lemma 2.12.

Fact 2.14. *Every fixed-parameter tractable counting problem is #W[1]-easy.*

Fact 2.15. *If a parameterized counting problem is both, #W[1]-hard and fixed-parameter tractable, then so are all #W[1]-easy problems, including #CLIQUE.*

Now let us revisit the parameterized counting problems we have seen so far. #CLIQUE is #W[1]-equivalent by definition. #VC is fixed-parameter tractable and therefore not #W[1]-hard, unless #CLIQUE is fixed-parameter tractable as well. Furthermore, the problem #HOM($\mathcal{H}$) is shown to be #W[1]-easy for all (recursively enumerable) classes $\mathcal{H}$ in Section 2.5.

In particular, all of those problems allow a parameterized Turing reduction to #CLIQUE. This seems to be different for the problem #DOMSET: While it is known that #CLIQUE $\leq^{\mathrm{T}}_{\mathrm{fpt}}$ #DOMSET (implicitly in [64]), the backward direction would imply that #W[1] coincides with the class #W[2]. Let us remark that the original definition of #W[2], as well as the proof of #W[2]-completeness of #DOMSET was given by Flum and Grohe [64]. A more detailed exposition of the classes #W[1] and #W[2] is out of the scope of this thesis. However, we recommend the interested reader to make themself familiar with what is called the W-hierarchy. Although this hierarchy consists of classes of parameterized decision problems, most of the structural insights transfer immediately to the counting versions. Excellent overviews are given by Downey and Fellows [57, Part V], Cygan et al. [45, Chapt. 13] and Flum and Grohe [65, Chapt. 7]. In particular, Flum and Grohe provide a comprehensive treatment of the #W-hierarchy in [65, Chapt. 14], the first two levels of which are #W[1] and #W[2]. For the purpose of this work, it will again suffice to define #W[2] via the complete problem #DOMSET.

Definition 2.16. A parameterized counting problem (P, κ) is called

- #W[2]-*hard* if #DOMSET $\leq^{\mathrm{T}}_{\mathrm{fpt}}$ (P, κ),
- #W[2]-*easy* if (P, κ) $\leq^{\mathrm{T}}_{\mathrm{fpt}}$ #DOMSET, and
- #W[2]-*equivalent* if it is both, #W[2]-hard and #W[2]-easy.

2.3.2 Fine-Grained Complexity Theory

In the previous section we defined the parameterized complexity classes #W[1] and #W[2] by complete problems, that is, #CLIQUE and #DOMSET. However, we did not provide any evidence for fixed-parameter intractability of problems that are hard for #W[1] and #W[2]. This will be done in the current section by introducing two popular conjectures from fine-grained complexity theory: The *Exponential Time Hypothesis* (ETH) and the *Strong Exponential Time Hypothesis* (SETH). Both conjectures arose from the fact that the classical assumption P $\neq$ NP usually does not yield tight lower bounds on the complexity of (NP-)hard problems, but instead only rules out polynomial time algorithms. Roughly speaking, ETH and SETH are stronger assumptions in the sense that they are used to disprove the existence of subexponential time algorithms [76, 77, 25, 47, 16]. Moreover, SETH rules out any non-trivial savings concerning the running time of known algorithms for some well studied problems [26, 109, 44], even for problems in P [134, 1, 2]. We will be particularly interested in the consequences of ETH and SETH in parameterized (counting) complexity. A detailed overview is given by Cygan et al. in Chapt. 14 of [45]. We proceed with formally defining ETH and SETH.

Definition 2.17 (Calabro, Impagliazzo and Paturi [76, 26]).

- The *Exponential Time Hypothesis* (ETH) asserts that 3-SAT cannot be solved in time $\mathsf{exp}(o(m))$, where m is the number of clauses of the input formula.
- The *Strong Exponential Time Hypothesis* (SETH) asserts that for every $\delta > 0$ there exists a positive integer k such that k-SAT cannot be solved in time $O(2^{(1-\delta)n})$, where n is the number of variables of the input formula.

Chen et al. established the following conditional lower bound for (the decision version of) #Clique under ETH. As promised, this result gives evidence for the fixed-parameter intractability of #W[1]-hard problems.

Theorem 2.18 ([33, 34]). *#Clique cannot be solved in time $f(k) \cdot n^{o(k)}$ for any function f unless ETH fails.*

Corollary 2.19. *Any #W[1]-hard parameterized counting problem is not fixed-parameter tractable unless ETH fails.*

Since ETH is widely believed by the community and a refutation would yield a significant speed-up for the satisfiability problem which generations of theoretical computer scientists have tried to come up with, we argue that Corollary 2.19 legitimizes the notion of #W[1]-hardness as strong evidence for fixed-parameter intractability.

Now let us take a closer look at Theorem 2.18. While any algorithm for #Clique running in time $f(k) \cdot n^{o(k)}$ is impossible under ETH, nothing is said about an algorithm running in time say $f(k) \cdot n^{0.99k}$. Indeed, fast matrix multiplication can be used to improve upon the brute force approach for counting cliques of size k.

Theorem 2.20 (Nešetřil and Poljak [105]). *Let ω be the matrix multiplication exponent. Then #Clique can be solved in time $f(k) \cdot n^{\omega k/3+O(1)}$ for some computable function f.*

Since $\omega \leq 2.3728639$ [67], we obtain an algorithm for #Clique that beats brute-force. However, Patrascu and Williams have shown that a similar improvement for #DomSet seems to be unlikely.

Theorem 2.21 ([109]). *#DomSet cannot be solved in time $O(f(k) \cdot n^{k-\varepsilon})$ for any $\varepsilon > 0$ and function f unless SETH fails.*

Indeed it seems that, empirically, #W[1]-easy problems do not allow strong lower bounds under SETH while problems hard for #W[2] do. Chapt 6 will provide an infinite amount of examples for the latter. We consider this phenomenon as weak evidence for the claim that #W[2]-hard problems are not #W[1]-easy. Further evidence is given by Downey and Fellows in an in-depth treatment of the W-hierarchy in Chapt. 23 of [57].

2.4 Combinatorial and Algebraic Methods

Throughout this thesis, we will encounter a large variety of tools, techniques and methods from discrete mathematics such as group theory and combinatorics, the most important of which are presented in the current section. We start by reviewing the notions of posets and lattices.

A *partially ordered set (poset)* is a pair $(L, \leq)$ of a finite universe L and a reflexive, transitive and anti-symmetric relation $\leq \subseteq L^2$. We write $\sigma \leq \rho$ if (σ, ρ) is contained in $\leq$ and we write $\sigma \geq \rho$ if $\rho \leq \sigma$. A pair $\sigma, \rho \in L$ has a *least upper bound* $\sigma \vee \rho$ if

- $\sigma \vee \rho \geq \sigma$ and $\sigma \vee \rho \geq \rho$, and
- for all $\delta \in L$ with $\delta \geq \sigma$ and $\delta \geq \rho$ it holds that $\delta \geq \sigma \vee \rho$.

A pair $\sigma, \rho \in L$ has a *greatest lower bound* $\sigma \wedge \rho$ if

- $\sigma \wedge \rho \leq \sigma$ and $\sigma \wedge \rho \leq \rho$, and
- for all $\delta \in L$ with $\delta \leq \sigma$ and $\delta \leq \rho$ it holds that $\delta \leq \sigma \vee \rho$.

A poset $(L, \leq)$ is called a *lattice* if every pair $\sigma, \rho \in L$ has a least upper bound and a greatest lower bound.

2.4.1 Matroids

We will follow the notation of the first chapter of the standard textbook of Oxley [107].

Definition 2.22. A *matroid* M is a pair $(E, \mathcal{I})$ of a finite set E and a non-empty subset $\mathcal{I} \subseteq \mathcal{P}(E)$ such that for every pair $A, B \subseteq E$ we have that

(1) if $A \in \mathcal{I}$ and $B \subseteq A$ then $B \in \mathcal{I}$, and

(2) if $A, B \in \mathcal{I}$ and $|B| < |A|$ then there exists an element $a \in A \setminus B$ such that $B \cup \{a\} \in \mathcal{I}$.

We call E the *ground set* of M. Furthermore, elements of $\mathcal{I}$ are referred to as *independent sets* of M. In particular, a subset-maximal element of $\mathcal{I}$ is called a *basis* of M and the *rank* of M, denoted by $\mathsf{rk}(M)$, is defined to be the size of a basis.

We point out that $\mathsf{rk}(M)$ is well-defined as every basis has the same cardinality due to (2). Now given a subset $X \subseteq E$ we write $\mathsf{rk}(X)$ for the rank of X which is defined to be the size of the largest independent set $A \subseteq X$. Furthermore, we define the *closure* of X to be

$$\mathsf{cl}(X) := \{e \in E \mid \mathsf{rk}(X \cup \{e\}) = \mathsf{rk}(X)\}.$$

By definition we have that $\mathsf{rk}(X) = \mathsf{rk}(\mathsf{cl}(X))$. Furthermore, we call X a *flat* if $\mathsf{cl}(X) = X$. Now given a matroid $M = (E, \mathcal{I})$ we write $L(M)$ for the set of all flats of M and observe that subset inclusion over $L(M)$ constitutes a lattice, called the *lattice of flats* of M. The least upper bound of two flats X and Y is given by $\mathsf{cl}(X \cup Y)$ and the greatest lower bound is given by $X \cap Y$. We point out that the lattices of flats are precisely the geometric lattices and refer the interested reader to e.g. Chapt. 3 of [133] and Chapt. 1.7 of [107] for a treatment of the latter.

We will be particularly interested in graphic matroids and their lattices of flats in this work.

Definition 2.23. Let $G = (V, E)$ be a graph. The *graphic matroid* $M(G)$ has as ground set the edges E of G and a set of edges $A \subseteq E$ is independent if the graph $G[A]$ does not contain a cycle.

If G is connected then the bases of $M(G)$ are precisely the spanning trees of G. If G consists of several connected components then every basis of $M(G)$ induces a spanning tree for each of the components and vice versa. Now given a subset of edges $X \subseteq E$, it can easily be verified that

$$\mathsf{rk}(X) = |V| - \mathsf{comp}(G[X]),$$

where comp denotes the number of connected components. Note that, in particular, every isolated vertex of $G[X]$ is a connected component.

2.4.2 Möbius Inversion and Inclusion-Exclusion

The Möbius inversion theorem, along with the special case of the (weighted) inclusion-exclusion principle yield powerful combinatorial “sieve methods” that will be crucial ingredients of many reductions in this work. An excellent and comprehensive treatment of this topic can be found in the standard textbook of Stanley [124, Chapt. 1-3].

Definition 2.24. Let $(L, \leq)$ be a poset and let $f : L \to \mathbb{C}$ be a function. Then the *zeta transformation* $\zeta f : L \to \mathbb{C}$ is given by

$$\zeta f(\sigma) := \sum_{\rho \geq \sigma} f(\rho).$$

We provide the following example of a zeta transformation. Fix a graph $H = (V, E)$ and let $(L, \leq)$ be the partition lattice of V, that is, every element $\delta \in L$ is a set of pairwise disjoint non-empty subsets of V such that

$$\bigcup_{S \in \delta} S = V.$$

Furthermore, given two partitions ρ and σ of V, we have that $\rho \geq \sigma$ if σ can be obtained from ρ by further partitioning ρ. More formally, $\rho \geq \sigma$ holds

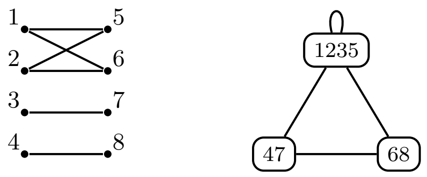


Figure 2.4: *Left:* A graph H with vertex set $V(H) = \{1, 2, 3, 4, 5, 6\}$. *Right:* The quotient graph H/δ for the partition $\delta = \{\{1, 2, 3, 5\}, \{47\}, \{68\}\}$.

whenever every set in σ is a subset of a set in ρ. We will see in Chapt. 4.1 that $(L, \leq)$ is not only a poset, but indeed the lattice of flats of the graphic matroid $M(V, V^2)$. Given $\delta \in L$, we define the *quotient graph* H/δ to be the graph that is obtained from H by contracting the vertices that are contained in the same set of δ and deleting multiple edges, but *keeping* self-loops. An illustration is given in Figure 2.4. Now fix a further graph G and define

$$f(\rho) := \#\mathsf{Emb}(H/\rho \to G)\,.$$

It is due to Lovász [95, Chapt. 5.2.3] that the zeta transformation ζf of f computes, given an element $\sigma \in L$, the number of homomorphisms from H/σ to G, i.e.,

$$\#\mathsf{Hom}(H/\sigma \to G) = \sum_{\rho \geq \sigma} \#\mathsf{Emb}(H/\rho \to G)\,. \tag{2.2}$$

In other words, zeta transformation allows us to express the number of homomorphisms as a sum of numbers of embeddings. The principle of Möbius inversion will allow us to invert this equation.

Theorem 2.25 (Möbius Inversion, cf. Proposition 3.7.2 in [124]). *Let $(L, \leq)$ be a poset. There exists a computable function $\mu_L : L \times L \to \mathbb{Z}$ such that for all $f : L \to \mathbb{C}$ and $\sigma \in L$ we have that*

$$f(\sigma) = \sum_{\rho \geq \sigma} \mu_L(\sigma, \rho) \cdot \zeta f(\rho)\,.$$

μ_L is called the Möbius function *of L.*

If the poset is clear from the context we will drop the L and only write μ. Now the application of Möbius inversion to the preceding example yields

$$\#\mathsf{Emb}(H/\sigma \to G) = \sum_{\rho \geq \sigma} \mu(\sigma, \rho) \cdot \#\mathsf{Hom}(H/\rho \to G)\,. \tag{2.3}$$

In particular, we can choose for σ the discrete partition $\perp$ with $|V(H)|$ many singletons and obtain

$$\#\mathsf{Emb}(H \to G) = \sum_{\rho \in L} \mu(\perp, \rho) \cdot \#\mathsf{Hom}(H/\rho \to G)\,. \tag{2.4}$$

We will depend on the following corollary of Rota's NBC Theorem which is concerned with the sign of the Möbius function over a lattice of flats.

Theorem 2.26 (Theorem 4 in [118]). *Let $(L, \leq)$ be the lattice of flats of some matroid and let $X \in L$ be a flat. Then we have that*

$$\mathsf{sign}(\mu_L(\emptyset, X)) = (-1)^{\mathsf{rk}(X)} .$$

Möbius inversion over the poset of subset inclusion yields the weighted inclusion-exclusion principle, see Chapt. 3.7 in [124] for a discussion. As we will only need the unweighted principle of inclusion and exclusion in a quite restricted setting of sieving, we provide an explicit statement.

Theorem 2.27 (Inclusion-exclusion, cf. Chapter 2.1 in [124]). *Let $A_0, \dots, A_{k-1}$ be subsets of a finite universe $\mathcal{U}$. Then we have that*

$$\#\left(\bigcap_{i=0}^{k-1} \overline{A_i}\right) = \sum_{J \subseteq [k]} (-1)^{|J|} \cdot |A_J| ,$$

where $A_J := \bigcap_{j \in J} A_j$ for $J \neq \emptyset$ and $A_\emptyset := \mathcal{U}$.

2.4.3 Multivariate Polynomial Interpolation

The method of polynomial interpolation yields a further, more implicit method of sieving which will be crucial for some reductions in the later chapters of this thesis. A concise statement of the principle for algorithmic purposes is given by Curticapean [39]:

Theorem 2.28 (Grid interpolation, cf. Theorem 1.38 in [39]). *Let $p \in \mathbb{Z}[x_0, \dots, x_{k-1}]$ be a multivariate polynomial such that for all $i \in [k]$ the degree of x_i is d_i. Furthermore, let*

$$\Upsilon = \Upsilon_0 \times \dots \times \Upsilon_{k-1} \subseteq \mathbb{Q}^k$$

such that $|\Upsilon_i| = d_i + 1$ for all $i \in [k]$. Then the coefficients of p can be computed with $O(|\Upsilon|^3)$ arithmetic operations when given as input the set

$$\mathcal{L} := \{(\upsilon, p(\upsilon)) \mid \upsilon \in \Upsilon\} .$$

Let us expose the most important ideas of the proof. To this end, we define $D := [d_0] \times \dots \times [d_{k-1}]$ and, given some $\mathbf{d} \in D$, we write $\upsilon^{\mathbf{d}}$ for

$$\prod_{i \in [k]} \upsilon^{\mathbf{d}_i}$$

Now observe that every element $(\upsilon, p(\upsilon))$ of $\mathcal{L}$ yields a linear equation

$$\sum_{\mathbf{d} \in D} c_{\mathbf{d}} \cdot \upsilon^{\mathbf{d}} = p(\upsilon) ,$$

where $c_{\mathbf{d}}$ is the coefficient of the monomial $x^{\mathbf{d}}$. In particular, the matrix $\mathcal{M}$ corresponding to the system of linear equations given by $\mathcal{L}$ can be written as a Kronecker product

$$\mathcal{M} = \bigotimes_{i \in [k]} \mathcal{V}_i \,,$$

where $\mathcal{V}_i$ is the *Vandermonde matrix* of $\Upsilon_i = \{\upsilon_{i,0}, \dots, \upsilon_{i,d_i}\}$, that is,

$$\mathcal{V}_i = \begin{pmatrix} 1 & \upsilon_{i,0} & \upsilon_{i,0}^2 & \dots & \upsilon_{i,0}^{d_i} \\ 1 & \upsilon_{i,1} & \upsilon_{i,1}^2 & \dots & \upsilon_{i,1}^{d_i} \\ 1 & \upsilon_{i,2} & \upsilon_{i,2}^2 & \dots & \upsilon_{i,2}^{d_i} \\ \vdots & \vdots & \vdots & \dots & \vdots \\ 1 & \upsilon_{i,d_i} & \upsilon_{i,d_i}^2 & \dots & \upsilon_{i,d_i}^{d_i} \end{pmatrix}.$$

It is well known that $\mathcal{V}_i$ is non-singular if $\upsilon_{i,0}, \dots, \upsilon_{i,d_i}$ are pairwise different, which is the case as $|\Upsilon_i| = d_i + 1$. Consequently, the Kronecker product $\mathcal{M}$ is non-singular as well and hence the coefficients $c_{\mathbf{d}}$ are uniquely determined an can be computed using Gaussian elimination.

2.4.4 Transformation Groups and Simplicial Complexes

Recall that a *group* is a pair of a set Γ and a function $\circ : \Gamma \times \Gamma \to \Gamma$ satisfying that

(1) for all $\alpha, \beta, \gamma \in \Gamma$ we have that $\alpha \circ (\beta \circ \gamma) = (\alpha \circ \beta) \circ \gamma$,

(2) there exists $e \in \Gamma$ such that $e \circ \gamma = \gamma \circ e = \gamma$ for all $\gamma \in \Gamma$, and

(3) for all $\gamma \in \Gamma$ there exists $\gamma^{-1} \in \Gamma$ such that $\gamma \circ \gamma^{-1} = \gamma^{-1} \circ \gamma = e$.

Examples of groups we have seen so far include $\mathbb{Z}$ and $\mathbb{Q}$ with addition, as well as the automorphism group $\mathsf{Aut}(H)$ of a graph H. Furthermore, we write $\mathbb{Z}_k$ for the group with elements $[k]$ and addition modulo k as operation.

Given a group $(\Gamma, \circ)$ and a set Ω, we call a function $\triangleright : \Gamma \times \Omega \to \Omega$ a *group action* if it satisfies the following two constraints:

(1) For all $s \in \Omega$ we have that $e \triangleright s = s$.

(2) For all $s \in \Omega$ and $\alpha, \beta \in \Gamma$ we have that $\alpha \triangleright (\beta \triangleright s) = (\alpha \circ \beta) \triangleright s$.

Consider for example a graph $H = (V, E)$. Then $\mathsf{Aut}(H)$ acts on both, V and E as given by $a \triangleright v := a(v)$ for all $v \in V$ and $a \triangleright \{u, v\} := \{a(u), a(v)\}$ for all $\{u, v\} \in E$.

Given a group $(\Gamma, \circ)$ acting on a set Ω and an element $s \in \Omega$, the *orbit* of s is defined as

$$\mathcal{O}(\Gamma, s) := \{\gamma \triangleright s \mid \gamma \in \Gamma\}.$$

Furthermore, the *stabilizer* of s is defined as

$$\mathcal{S}(\Gamma, s) := \{\gamma \in \Gamma \mid \gamma \triangleright s = s\}.$$

If the group is clear from the context, we might only wright $\mathcal{O}(s)$ and $\mathcal{S}(s)$. Note that Ω is partitioned by its orbits, that is, there exist $s_1, \dots, s_k$ such that

$$\Omega = \mathcal{O}(s_1) \mathbin{\dot\cup} \dots \mathbin{\dot\cup} \mathcal{O}(s_k).$$

A group action is called *transitive* if $k = 1$, i.e., if there exists only a single orbit. Adopting this notion, we call a graph $H = (V, E)$ *vertex-transitive* if $\mathsf{Aut}(H)$ acts transitively on V and we call it *edge-transitive* if $\mathsf{Aut}(H)$ acts transitively on E.

The following is a well-known result relating the cardinalities of Ω, $\mathcal{O}(s)$ and $\mathcal{S}(s)$.

Theorem 2.29 (Orbit-Stabilizer-Theorem, cf. Chapt. 1.5 in [93]). *Let $(\Gamma, \circ)$ be a finite group acting on a finite set Ω. Then the following holds for all $s \in \Omega$:*

$$|\Gamma| = |\mathcal{O}(s)| \cdot |\mathcal{S}(s)|$$

We will be particularly interested in orbits of group actions on what is called simplicial complexes.

Definition 2.30. A *simplicial complex* Δ is a pair $(\Omega, \mathcal{I})$ of a finite ground set Ω and a subset $\mathcal{I} \subseteq \mathcal{P}(\Omega)$ such that

(1) $\{s\} \in \mathcal{I}$ for all $s \in \Omega$,

(2) $\emptyset \notin \mathcal{I}$, and

(3) if $A \in \mathcal{I}$ and $\emptyset \subsetneq B \subseteq A$ then $B \in \mathcal{I}$.

An element $A \in \mathcal{I}$ is called a *simplex*.

In the remainder of this thesis we abuse notation and write $A \in \Delta$ for a simplex $A \in \mathcal{I}$.

The following observation points out that simplicial complexes generalize matroids.

Observation 2.31. *Let $M = (E, \mathcal{I})$ be a matroid such that $\{e\} \in \mathcal{I}$ for all elements of the ground set $e \in E$. Then $(E, \mathcal{I} \setminus \{\emptyset\})$ is a simplicial complex.*

Given a simplex A of Δ we define the *dimension* of A, denoted as $\mathsf{dim}(A)$, to be $\#A - 1$. The *Euler characteristic* χ of a simplicial complex Δ is defined to be

$$\chi(\Delta) := \sum_{i \geq 0} (-1)^i \cdot \#\{A \in \Delta \mid \mathsf{dim}(A) = i\}.$$

The *reduced Euler characteristic* of Δ is defined to be

$$\hat{\chi}(\Delta) := 1 - \chi(\Delta)\,.$$

It will be very convenient to rewrite the reduced Euler characteristic as follows.

Fact 2.32. $\hat{\chi}(\Delta) = \sum_{i\geq 0}(-1)^i \cdot \#\{A \in \Delta \cup \{\emptyset\} \mid \#A = i\}$.

Given a simplicial complex Δ and a finite group Γ that acts on the ground set Ω of Δ, we say that Δ is a Γ-*simplicial complex* if the induced action of Γ on subsets of Ω preserves Δ. More precisely, if $A \in \Delta$ and $g \in \Gamma$ then the set $g \triangleright A := \{g \triangleright a \mid a \in A\}$ is contained in Δ as well. If this is the case we can define the *fixed-point complex* Δ^Γ as follows. Let $\mathcal{O}_1, \dots, \mathcal{O}_k$ be the orbits of Ω with respect to the action of Γ. Then

$$\Delta^\Gamma := \left\{ S \subseteq \{1,\dots,k\} \;\middle|\; S \neq \emptyset \wedge \bigcup_{i\in S} \mathcal{O}_i \in \Delta \right\}$$

The following topological fixed-point theorem[3] is due to Smith [123] (cf. [106] and Chapt. 3 in [18]) and will be of crucial importance in Chapter 5.1.

Theorem 2.33. *Let Γ a group of order p^s for some prime p and natural number s and let Δ be a Γ-simplicial complex. Then $\chi(\Delta) \equiv \chi(\Delta^\Gamma) \mod p$ and hence $\hat{\chi}(\Delta) \equiv \hat{\chi}(\Delta^\Gamma) \mod p$.*

2.5 A Classification for Counting Homomorphisms

The complexity dichotomy result of the homomorphism counting problem $\#\textsc{Hom}(\mathcal{H})$ is a fundamental theorem in parameterized counting complexity theory that will be the basis of most of the contributions in this thesis. The original classification of $\#\textsc{Hom}(\mathcal{H})$ is due to Dalmau and Jonsson [46], and was established for the even more general case of counting homomorphisms between logical structures. We decided to give a self-contained proof of the classification in the current section, the reasons for which are twofold. First, the proof will invoke most of the machinery we have introduced in the preceding sections — from treewidth and grid minors to parameterized reductions relying on the inclusion-exclusion principle and polynomial interpolation — allowing us to provide a coherent set of examples. Second, we will introduce color-prescribed graph homomorphisms which constitute an important technical tool used in Chapter 5.2 and, in a more general form, in Chapter 6.

[3]Indeed, simplicial complexes have a topological interpretation. A concise introduction to the topic, including an explicit statement and proof of Theorem 2.33 can be found in Edmond's lecture notes on transformation groups [60] (cited February 2019).

Before we begin, let us be very clear that the current section does not contain new contributions but rather provides an alternative presentation of the proof of the following complexity classification.

Theorem 2.34 ([46]). *Let $\mathcal{H}$ be a recursively enumerable class of graphs.*

1. If $\mathcal{H}$ has bounded treewidth then $\#\textsc{Hom}(\mathcal{H})$ is polynomial-time solvable.

2. Otherwise $\#\textsc{Hom}(\mathcal{H})$ is $\#\mathsf{W}[1]$-equivalent.

We start with an algorithm based on dynamic programming over tree decompositions that allows us to solve $\#\textsc{Hom}(\mathcal{H})$ in polynomial time, given that the treewidth of $\mathcal{H}$ is bounded. In particular, we prove the following theorem which is slightly weaker than the result of Díaz et al. [50].

Theorem 2.35. *There is a deterministic algorithm that, given graphs H and G and a tree decomposition $(T, \{B_t\}_{t \in V(T)})$ of width ℓ of H, computes $\#\mathsf{Hom}(H \to G)$ in time*

$$\mathsf{poly}(s, k, \ell) \cdot n^{\ell + O(1)}\,,$$

where $s = |V(T)|$, $k = |V(H)|$ and $n = |V(G)|$.

Recall that the notion of treewidth was motivated by the existence of small separators. This is made formal in the following lemma.

Lemma 2.36 (Lemma 7.3 in [45]). *Let $(T, \{B_t\}_{t \in V(T)})$ be a tree decomposition of a graph H and let $\{a, b\}$ be an edge of T. Furthermore, let T_a and T_b be the two trees obtained from T by deleting $\{a, b\}$ such that $a \in V(T_a)$ and $b \in V(T_b)$. Define*

$$A := \bigcup_{t \in V(T_a)} B_t \quad \text{and} \quad B := \bigcup_{t \in V(T_b)} B_t\,.$$

*Then there exists **no** edge $\{u, v\}$ of H such that $u \in A \setminus B$ and $v \in B \setminus A$.*

Furthermore, it will be very convenient to assume a nice tree decomposition (see e.g. Chapt. 7.2 in [45]) in the proof of Theorem 2.35. To this end, we say that a tree decomposition $(T, \{B_t\}_{t \in V(T)})$ of a graph H is *nice* if T is a rooted binary tree such that $B_r = \emptyset$ for the root r and $B_\ell = \emptyset$ for every leaf l and all other vertices t of T are of one of the following types:

(1) *Introduce*: t has precisely one child t' and $B_t = B_{t'} \cup \{v\}$ for some vertex v of H with $v \notin B_{t'}$.

(2) *Forget*: t has precisely one child t' and $B_{t'} = B_t \cup \{v\}$ for some vertex v of H with $v \notin B_t$.

(3) *Join*: t has precisely two children t_1 and t_2 such that $B_t = B_{t_1} = B_{t_2}$.

The next lemma guarantees that one can always assume that a given tree decomposition is nice.

Lemma 2.37 (Lemma 7.3 in [45]). *There is a deterministic algorithm that, given a tree decomposition $(T, \{B_t\}_{t\in V(T)})$ of width ℓ of a graph H, computes in time $O(\ell^2 \cdot \mathsf{max}\{|V(T)|, |V(H)|\})$ a nice tree decomposition $(\hat{T}, \{B_t\}_{t\in V(\hat{T})})$ of H of width at most ℓ such that $|V(\hat{T})| \in O(\ell|V(H)|)$.*

Now we have everything we need to proceed with the proof of Theorem 2.35.

Proof (of Theorem 2.35). Given graphs H and G and a tree decomposition $(T, \{B_t\}_{t\in V(T)})$ of width ℓ of H we first invoke Lemma 2.37 to obtain a nice tree decomposition $(\hat{T}, \{B_t\}_{t\in V(\hat{T})})$ of H of width at most ℓ. Next we introduce some further notation which is required for the algorithm: Given $t \in V(\hat{T})$, we write $\hat{T}_t$ for the subtree of $\hat{T}$ rooted at t. Furthermore, we define

$$H_t := H\left[\bigcup_{t'\in V(\hat{T}_t)} B_{t'}\right],$$

that is, H_t is the subgraph of H induced by the vertices contained in the bags of t and its descendants. Now let DP be the function that, given $t \in V(\hat{T})$ and a function $f : B_t \to V(G)$, computes

$$\mathsf{DP}[t, f] := \#\{h \in \mathsf{Hom}(H_t \to G) \mid h|_{B_t} = f\}\,.$$

We will compute DP for all t and f using dynamic programming. Note that this requires us to initialize a table of size at most

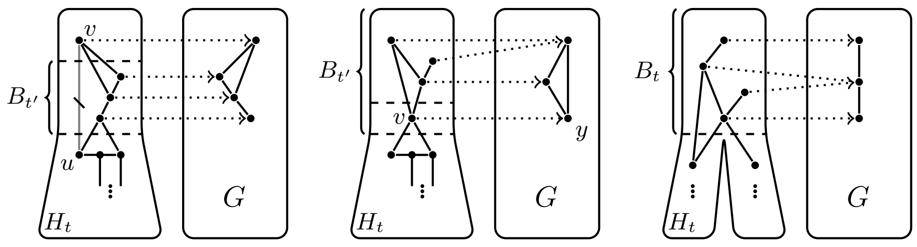
$$|V(\hat{T})| \cdot |V(G)|^{\ell+1} \overset{\text{Lemma 2.37}}{\leq} O(\ell|V(H)|) \cdot |V(G)|^{\ell+1}\,. \tag{2.5}$$

We initialize the table by setting $\mathsf{DP}[l, \emptyset] = 1$ for every leaf l. In what follows, it is shown how to fill the table from bottom up. More precisely, we provide rules for each of the three types of vertices of $\hat{T}$; consult Figure 2.5 for an illustration of each rule.

(1) Let $t \in V(\hat{T})$ be a vertex of type *introduce*. Then t has precisely one child t' and $B_t = B_{t'} \,\dot\cup\, \{v\}$. We establish the following claims.

Claim 2.38. $N_{H_t}(v) \subseteq B_{t'}$.

Proof. Assume for contradiction that there exists $u \in V(H_t)\backslash B_{t'}$ which is adjacent to v. Now invoke Lemma 2.36 with $a = t'$ and $b = t$ and set A and B accordingly. Then $u \in A$ and $v \in B$. By the definition of a tree decomposition, we have that $u \notin B$, because otherwise the subgraph $\hat{T}[\{t \mid u \in B_t\}]$ of $\hat{T}$ would not be connected as $u \notin B_{t'}$ by assumption. Similarly, $v \notin A$ as $v \notin B_{t'}$ as well. Hence, $u \in A \setminus B$ and $v \in B \setminus A$ which yields the desired contradiction. □



Introduce: $B_t = B_{t'} \,\dot\cup\, \{v\}$ **Forget:** $B_{t'} = B_t \,\dot\cup\, \{v\}$ **Join:** $B_t = B_{t_1} = B_{t_2}$

Figure 2.5: Illustration of the three recursive steps of the dynamic programming algorithm used in the proof of Theorem 2.35. The function f is depicted as dotted arrows. Additionally, Claim 2.38 is highlighted in the first graphic with a crossed out gray line from v to u.

Claim 2.39. *We have that:*

$$\mathsf{DP}[t, f] = \begin{cases} 0 & f(N_{H_t}(v)) \not\subseteq N_G(f(v)) \\ \mathsf{DP}[t', f|_{B_{t'}}] & \textit{otherwise} \end{cases}$$

Proof. Assume first that $f(N_{H_t}(v)) \not\subseteq N_G(f(v))$. Then there exists a vertex w of H_t that is adjacent to v, but $f(w)$ is not adjacent to $f(v)$. In this case, f cannot be extended to a homomorphism h from H_t to G and hence $\mathsf{DP}[t, f] = 0$.

Otherwise, homomorphisms $h \in \mathsf{Hom}(H_t \to G)$ s.t. $h|_{B_t} = f$ are in one-to-one correspondence with homomorphisms $h' \in \mathsf{Hom}(H_{t'} \to G)$ s.t. $h'|_{B_{t'}} = f|_{B_{t'}}$, which is given by the bijection

$$h \mapsto h' := h|_{V(H_{t'})} .$$

Note that this bijection is well-defined as by Claim 2.38 there are no vertices u in $V(H_t) \setminus B_{t'}$ that are adjacent to v (see Figure 2.5). □

(2) Let $t \in V(\hat{T})$ be a vertex of type *forget*. Then t has precisely one child t' and $B_{t'} = B_t \,\dot\cup\, \{v\}$. In particular, $H_t = H_{t'}$. The rule for recursion is given by the following claim.

Claim 2.40. *We have that:*

$$\mathsf{DP}[t, f] = \sum_{y \in V(G)} \mathsf{DP}[t', f_y] ,$$

where $f_y(v) := y$ and $f_y(x) := f(x)$ for $x \neq v$.

Proof. Partition the homomorphisms $h \in \mathsf{Hom}(H_t \to G)$ s.t. $h|_{B_t} = f$ by the image of h on input v. This induces an equivalence relation for which each equivalence class is uniquely identified by a vertex y of G. The size of a class identified by y is hence precisely $\mathsf{DP}[t', f_y]$. □

(3) Let $t \in V(\hat{T})$ be a vertex of type *join*. Then t has precisely two children t_1 and t_2 such that $B_t = B_{t_1} = B_{t_2}$. The rule for recursion is given by the following claim.

Claim 2.41. *We have that:*

$$\mathsf{DP}[t, f] = \mathsf{DP}[t_1, f] \cdot \mathsf{DP}[t_2, f]\,.$$

Proof. We construct a bijection from pairs of homomorphisms (h_1, h_2), where $h_i \in \mathsf{Hom}(H_{t_i} \to G)$ such that $h_i|_{B_{t_i}} = f$ for $i \in \{1, 2\}$ to homomorphisms $h \in \mathsf{Hom}(H_t \to G)$ such that $h|_{B_t} = f$. The bijection is given by

$$b(h_1, h_2)(v) := \begin{cases} f(v) & v \in B_t \\ h_1(v) & v \in V(H_{t_1}) \setminus B_t \\ h_2(v) & v \in V(H_{t_2}) \setminus B_t \end{cases}$$

Note that b is well-defined as $B_t = B_{t_1} = B_{t_2}$ and furthermore there is no edge $\{u, w\} \in E(H_t)$ such that $u \in V(H_{t_1})\setminus B_t$ and $w \in V(H_{t_2})\setminus B_t$, the latter of which is proved similar to Claim 2.38. □

The algorithm for computing the table DP is given by Claim 2.39, Claim 2.40 and Claim 2.41. The final output is

$$\mathsf{DP}[r, \emptyset] = \#\{h \in \mathsf{Hom}(H_r \to G) \mid h|_\emptyset = \emptyset\} = \#\mathsf{Hom}(H \to G)\,.$$

For the running time, we observe that every update of the table DP takes time $\mathsf{poly}(\ell, |V(G)|)$. Together with the bound on the size of the table given by Equation (2.5), we obtain the following bound on the overall running time; recall that $s = |V(T)|$, $k = |V(H)|$ and $n = |V(G)|$.

$$O(\ell^2 \cdot \mathsf{max}\{s, k\}) + O(\ell \cdot k) \cdot n^{\ell+1} \cdot \mathsf{poly}(\ell, n) = \mathsf{poly}(s, k, \ell) \cdot n^{\ell+O(1)} \quad ■$$

Remark 2.42 ([50], see also the full version of [41]). Refining the algorithm and its analysis in the above proof yields an enhanced running time of $\mathsf{poly}(s, k, \ell) \cdot n^{\ell+1}$.

Corollary 2.43. *Let $\mathcal{H}$ be class of graphs of bounded treewidth. Then the problem $\#\textsc{Hom}(\mathcal{H})$ is solvable in polynomial time.*

Proof. The only catch in applying Theorem 2.35 is the fact that computing a tree decomposition of minimum width is NP-hard [5]. However, Bodlaender's algorithm [14] allows us, given a graph H and $c \in \mathbb{N}$, to compute a tree decomposition of H of width at most c if $\mathsf{tw}(H) \leq c$ or to correctly decide that $\mathsf{tw}(H) > c$. Furthermore, the algorithm runs in time

$$c^{O(c^3)} \cdot |V(H)|\,.$$

Now let c be an upper bound on the treewidth of graphs in $\mathcal{H}$. Then #HOM($\mathcal{H}$) can be solved as follows. On input H and G, we first apply Bodlaender's algorithm to H and c. As $\mathsf{tw}(H) \leq c$ by assumption, we obtain in time $O(|V(H)|)$ — note that c is a constant — a tree decomposition $(T, \{B_t\}_{t \in V(T)})$ of width at most c of H. In particular, $|V(T)| \in O(|V(H)|)$ because otherwise there would be not enough time to output T. By Theorem 2.35 we can then perform dynamic programming over a nice tree decomposition computed from $(T, \{B_t\}_{t \in V(T)})$ and obtain a total running time of

$$O(|V(H)|) + \mathsf{poly}(|V(H)|, c) \cdot |V(G)|^{c+O(1)},$$

which clearly is a polynomial in the input size. ■

It remains to prove #W[1]-equivalence of #HOM($\mathcal{H}$) in case $\mathcal{H}$ is of unbounded treewidth. To this end, we will introduce the following color-prescribed variant of graph homomorphisms (see also [95, Chapter 5.4.2]).

Definition 2.44. A graph G is *H-colored* if it comes with a homomorphism c from G to H. The value $c(v)$ for a vertex $v \in V(G)$ is called the *color* of v and given a vertex $u \in V(H)$ we write $c^{-1}(u)$ for the set of all vertices in G that are mapped to u.

A homomorphism $h \in \mathsf{Hom}(H \to G)$ is called *color-prescribed* if every vertex $u \in V(H)$ maps to a vertex $h(u)$ whose color is u. We write $\mathsf{cp\text{-}Hom}(H \to G)$ for the set of all color-prescribed homomorphisms. Similarly, we write #CP-HOM($\mathcal{H}$) for the problem of, given a graph $H \in \mathcal{H}$ and an H-colored graph G, computing $\#\mathsf{cp\text{-}Hom}(H \to G)$; the problem is parameterized by $|V(H)|$.

In what follows, we write $\boxplus$ for the set of all grids $\boxplus_k$ for $k \in \mathbb{N}$. The next lemma will be the basis of our sequence of reductions.

Lemma 2.45. #CLIQUE $\leq^{\mathrm{T}}_{\mathrm{fpt}}$ #CP-HOM($\boxplus$).

Proof. First of all, we point out that #CP-HOM($\boxplus$) is essentially equivalent to the counting version of the *grid tiling* problem (see e.g. [45, Chapt. 14.4.1] and [39, Problem 1.12]). For this reason, the following reduction is similar to existing hardness proofs for variants of the grid tiling problem. We include it for completeness.

Given an instance (G, k) of #CLIQUE, we construct a $\boxplus_k$-colored graph $\hat{G}$ from G as follows. The vertex set of $\hat{G}$ is partitioned by

$$V(\hat{G}) := \bigcup_{i,j \in [k]} V_{i,j},$$

where

$$V_{i,j} := \begin{cases} \{(v,v) \mid v \in V(G)\} & i = j \\ E(G) & i \neq j \end{cases}.$$

There are only two kinds of edges of $\hat{G}$.

- *Horizontal edges* connect vertices in $V_{i,j}$ and $V_{i,j+1}$ for $i \in [k]$ and $j \in [k-1]$ as follows: Let $u = (a,b) \in V_{i,j}$ and $v = (a',b') \in V_{i,j+1}$. Then $\{u,v\} \in E(\hat{G})$ if and only if $a = a'$.

- *Vertical edges* connect vertices in $V_{i,j}$ and $V_{i+1,j}$ for $i \in [k-1]$ and $j \in [k]$ as follows: Let $u = (a,b) \in V_{i,j}$ and $v = (a',b') \in V_{i+1,j}$. Then $\{u,v\} \in E(\hat{G})$ if and only if $b = b'$.

The $\boxplus_k$-coloring $c \in \mathsf{Hom}(G \to \boxplus_k)$ of $\hat{G}$ is given by $c(v) = (i,j)$ if and only if $v \in V_{i,j}$.

Intuitively, the above construction yields a graph $\hat{G}$ whose vertices are partitioned in k^2 blocks $V_{i,j}$ for $i,j \in [k]$ where each block $V_{i,j}$ corresponds to the vertex (i,j) of the k-grid $\boxplus_k$. The blocks on the diagonal contain as vertices pairs (v,v) for $v \in V(G)$ and the remaining blocks contain as vertices pairs (u,v) where $\{u,v\} \in E(G)$ — recall that $\{u,v\} \in E(G)$ is only notation for $(u,v) \in E(G)$ and $(v,u) \in E(G)$.

Claim 2.46. *The number of cliques of size k in G equals*

$$(k!)^{-1} \cdot \#\mathsf{cp\text{-}Hom}(\boxplus_k \to \hat{G})\,.$$

Proof. The factor of $(k!)^{-1}$ is due to the fact that there are $k!$ possibilities to order the vertices of a clique of size k. For this reason, we will consider k-cliques to be ordered tuples instead of sets in this proof and construct a bijection to $\mathsf{cp\text{-}Hom}(\boxplus_k \to \hat{G})$ in what follows.

Let $(v_i)_{i\in[k]}$ be a clique of size k in G. We define a color-prescribed homomorphism $h \in \mathsf{cp\text{-}Hom}(\boxplus_k \to \hat{G})$ by $h(i,j) := (v_i, v_j) \in V_{i,j}$. While color-prescribedness follows immediately from the definition, it is a priori not clear that h is well-defined in the sense that there exists a vertex (v_i, v_j) in $V_{i,j}$ for every $i,j \in [k]$. We hence consider two cases. If $i = j$ then (v_i, v_j) is included in $V_{i,j}$ by definition of $\hat{G}$. Otherwise, we observe that $\{v_i, v_j\} \in E(G)$ as $(v_i)_{i\in[k]}$ is a clique. Therefore, by construction, the vertex (v_i, v_j) is contained in $V_{i,j}$. Now let $\{(i,j),(i',j')\} \in E(\boxplus_k)$. By the definition of the grid we have that either $i' = i$ and $j' = j+1$ for $i \in [k]$ and $j \in [k-1]$, or $i' = i+1$ and $j' = j$ for $i \in [k-1]$ and $j \in [k]$. As $h(i,j) = (v_i, v_j) \in V_{i,j}$ and $h(i',j') = (v_{i'}, v_{j'}) \in V_{i',j'}$ we conclude that in the first case the adjacency is preserved by a *horizontal edge* and in the second case the adjacency if preserved by a *vertical edge.*

For the other direction let $h \in \mathsf{cp\text{-}Hom}(\boxplus_k \to \hat{G})$. As h is color-prescribed we have that for every $i,j \in [k]$ the vertex $(i,j) \in V(\boxplus_k)$ is mapped to a vertex in $V_{i,j}$ by h. For every $i \in [k]$ let (v_i, v_i) be the vertex of $\hat{G}$ which is the image of $h(i,i)$. We claim that $(v_i)_{i\in[k]}$ is a clique in G. To this end we show that for each pair $i < j$ the edge (v_i, v_j) is contained in $E(G)$.

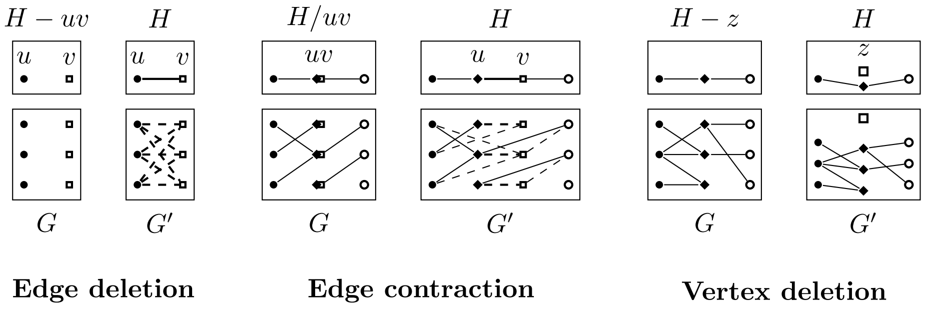


Figure 2.6: Illustration of the reduction for each operation as demonstrated in Lemma 2.47. Edges that are added in the reduction are dashed.

As h is a homomorphism, the following edges must be contained in $\hat{G}$:

$$\begin{tikzcd}
h(i,i) \arrow[r, dash] & h(i,i+1) \arrow[r, dash] & \dots & h(i,j) \\
 & & & \vdots \arrow[d, dash] \\
 & & & h(j-1,j) \arrow[d, dash] \\
 & & & h(j,j)
\end{tikzcd}$$

Now recall that $h(i,i) = (v_i, v_i)$ and $h(j,j) = (v_j, v_j)$ and let $h(i,j) = (a,b)$. Inductively applying the definition of *horizontal edges* yields that $a = v_i$. Similarly, using *vertical edges*, we obtain that $b = v_j$. Hence $(v_i, v_j) \in V_{i,j}$ and consequently $\{v_i, v_j\} \in E(G)$. This concludes the proof. $\square$

Using Claim 2.46, the algorithm of the reduction hence first computes $\hat{G}$, including its $\boxplus_k$-coloring, and then queries the oracle for #cp-Hom($\boxplus$) to obtain the number $\#\textsf{cp-Hom}(\boxplus_k \to G)$, which is then divided by $k!$ and returned as output. $\blacksquare$

Next we show that the complexity of #cp-Hom($\mathcal{H}$) cannot increase by taking minors.

Lemma 2.47. *Let H be a graph and let $\hat{H}$ be a minor of H. Given an $\hat{H}$-colored graph G, we can in polynomial time compute an H-colored graph G' with $|V(G')| \leq |V(H)| \cdot |V(G)|$ that satisfies*

$$\#\textsf{cp-Hom}(\hat{H} \to G) = \#\textsf{cp-Hom}(H \to G')\,.$$

Proof. The claim is trivial if $\hat{H}$ and H are equal. We prove the claim in case $\hat{H}$ is obtained from H by a single deletion or contraction operation, the full result then follows by induction. Figure 2.6 illustrates the proof for each of the three operations. In what follows, we abuse notation and write only uv for an edge $\{u,v\}$.

Edge deletions. Let $e \in E(H)$ be an edge with $e = uv$ and suppose that $\hat{H} = H - e$. Let G be a $\hat{H}$-colored graph given as input, together with the coloring $c\colon V(G) \to V(\hat{H})$. To construct G', we start from G and simply add all possible edges between the color classes $c^{-1}(u)$ and $c^{-1}(v)$; clearly this construction takes polynomial time, G' has the same number of vertices as G, and c is a homomorphism from G' to H. To verify the correctness, we show that $\mathsf{cp\text{-}Hom}(\hat{H} \to G) = \mathsf{cp\text{-}Hom}(H \to G')$ holds. Indeed, let $h : V(H) \to V(G)$ be a color-prescribed mapping. Since $h(e) \in E(G')$ holds by construction and e is the only edge where H and $\hat{H}$ differ, the addition of the edge e does not matter. Hence h is an element of $\mathsf{cp\text{-}Hom}(\hat{H} \to G)$ if and only if it is an element of $\mathsf{cp\text{-}Hom}(H \to G')$.

Vertex deletions. Let $z \in V(H)$ be a vertex and suppose $\hat{H} = H - z$. We can w.l.o.g. assume that z is isolated, because otherwise we could have performed edge deletions as above before. Let G be a $\hat{H}$-colored graph given as input, together with the coloring $c\colon V(G) \to V(\hat{H})$. To construct G', we start from G and simply add an isolated vertex z' to it, whose color $c(z')$ we define as z; clearly c is now a homomorphism from G' to H. To verify the correctness, observe that $\#\mathsf{cp\text{-}Hom}(\hat{H} \to G) = \#\mathsf{cp\text{-}Hom}(H \to G')$ holds: Any color-prescribed homomorphism h from H to G' remains a color-prescribed homomorphism from $\hat{H}$ to G by restricting h to $V(\hat{H})$. Conversely, any h from $\hat{H}$ to G can be extended in exactly one color-prescribed way by setting $h(z) = z'$. Thus the number of color-prescribed homomorphisms stays the same.

Edge contractions. Let $e \in E(H)$ be an edge with $e = uv$, and suppose $\hat{H} = H/e$. Contracting the edge e in H identifies the vertices u and v; let us call the new vertex $w \in V(\hat{H})$. Let G be a $\hat{H}$-colored graph given as input, together with the coloring $c\colon V(G) \to V(\hat{H})$. We want to use G' to ensure that any color-prescribed homomorphism h from H to G' assigns u and v to the same value, that is, satisfies the equality constraint $h(u) = h(v)$. To do this, we simply put an induced perfect matching in G' between the color class of u and the color class of v. More formally, we start from G and split every vertex $x \in c^{-1}(w)$ into an edge $x_u x_v$ in G', but we leave their neighborhoods intact, that is, we have $N_{G'}(x_u) \cap V(G) = N_{G'}(x_v) \cap V(G) = N_G(x)$. Clearly G' is now H-colored, and it has exactly $|c^{-1}(w)|$ vertices more than G. To verify correctness, again observe that $\#\mathsf{cp\text{-}Hom}(\hat{H} \to G) = \#\mathsf{cp\text{-}Hom}(H \to G')$ holds: Our construction forces any color-prescribed homomorphism h from H to G' to satisfy $h(u) = h(v)$ and thus gives rise to a color-prescribed homomorphism $\hat{h}$ from $\hat{H}$ to G by setting $\hat{h}(w) = h(u)$; this mapping $h \mapsto \hat{h}$ is a bijection. ■

Corollary 2.48. *Let $\hat{\mathcal{H}}$ and $\mathcal{H}$ be recursively enumerable classes of graphs such that for every $\hat{H} \in \hat{\mathcal{H}}$ there exists a graph $H \in \mathcal{H}$ such that $\hat{H}$ is a minor of H. Then $\#\textsc{cp-Hom}(\hat{\mathcal{H}}) \leq^{\mathrm{T}}_{\mathrm{fpt}} \#\textsc{cp-Hom}(\mathcal{H})$.*

Proof. Let $(\hat{H}, G)$ be an instance of $\#\textsc{cp-Hom}(\hat{\mathcal{H}})$, that is, $\hat{H} \in \hat{\mathcal{H}}$ and G is an $\hat{H}$-colored graph for which we want to compute $\#\mathsf{cp\text{-}Hom}(\hat{H} \to G)$. As $\mathcal{H}$ is recursively enumerable, there exists a computable function f independent of the graph $\hat{H}$ such that a graph $H \in \mathcal{H}$ which contains $\hat{H}$ as a minor can be found in time $f(|V(\hat{H})|)$. We can then invoke the algorithm of Lemma 2.47 to conclude the reduction — note that this yields indeed a parameterized Turing reduction as the size of the graph H and hence the parameter of the oracle query is bounded by $f(|V(\hat{H})|)$ as well. ∎

It remains to show that $\#\textsc{Hom}(\mathcal{H})$ and $\#\textsc{cp-Hom}(\mathcal{H})$ are interreducible. We start with the slightly more involved direction.

Lemma 2.49. *$\#\textsc{cp-Hom}(\mathcal{H}) \leq^{\mathrm{T}}_{\mathrm{fpt}} \#\textsc{Hom}(\mathcal{H})$.*

The proof of the above lemma will be presented in two steps via the following intermediate problem.

Definition 2.50. Given an H-colored graph G with coloring c, we say that a homomorphism $h \in \mathsf{Hom}(H \to G)$ is *colorful* if $c(h(V(H))) = V(H)$, that is, the image of h must contain vertices of all colors of G. We write $\mathsf{cf\text{-}Hom}(H \to G)$ for the set of all colorful homomorphisms from H to G. Furthermore, we define $\#\textsc{cf-Hom}(\mathcal{H})$ as the problem of, given a graph H and an H-colored graph G, computing $\#\mathsf{cf\text{-}Hom}(H \to G)$; the problem is parameterized by $|V(H)|$.

Lemma 2.51. *$\#\textsc{cp-Hom}(\mathcal{H}) \leq^{\mathrm{T}}_{\mathrm{fpt}} \#\textsc{cf-Hom}(\mathcal{H})$.*

Proof. Let (H, G) be an instance of $\#\textsc{cp-Hom}(\mathcal{H})$, that is, $H \in \mathcal{H}$ and G is an H-colored graph. Let furthermore c be the H-coloring of G. We claim the following, which immediately induces the reduction.

$$\#\mathsf{cf\text{-}Hom}(H \to G) = \#\mathsf{Aut}(H) \cdot \#\mathsf{cp\text{-}Hom}(H \to G)\,.$$

Intuitively, this holds as the set $\mathsf{cf\text{-}Hom}(H \to G)$ can be partitioned by the images of the colorful homomorphisms and the induced equivalence classes are each of size $\#\mathsf{Aut}(H)$ and represented by a color-prescribed homomorphism. What follows is a formal proof of this statement using a rather simple application of the Orbit-Stabilizer-Theorem.

If the coloring c is not surjective, then there are neither color-prescribed nor colorful homomorphisms from H to G, so both sides of the equation are zero. Otherwise we define the following group action of $\mathsf{Aut}(H)$ on the set $\mathsf{cf\text{-}Hom}(H \to G)$:

$$a \triangleright h := a \circ h$$

Now consider the function b which maps *color-prescribed* homomorphisms to orbits of the above group action given by

$$b(h) := \mathcal{O}(h)\Big(= \{a \triangleright h \mid a \in \mathsf{Aut}(H)\}\Big)\,.$$

It is easy to see that b is injective, as two different color-prescribed homomorphisms must have different images, and so they cannot be in the same orbit since composing an automorphism with a homomorphism does not change the image.

For surjectivity, let $\mathcal{O} = \mathcal{O}(h')$ be the orbit of some colorful homomorphism $h' \in \mathsf{cf\text{-}Hom}(H \to G)$. Now observe that $h' \circ c$ is a surjective homomorphism from H to H as h' is colorful and c is surjective. Furthermore, every surjective homomorphism from H to H is an automorphism. Since $\mathsf{Aut}(H)$ is a group, there exists hence an inverse $a \in \mathsf{Aut}(H)$ of $(h' \circ c)$ such that

$$a \circ (h' \circ c) = \mathsf{id}\,,$$

where id is the neutral element of $\mathsf{Aut}(H)$, that is, the identity. By associativity of functional composition we obtain that

$$(a \circ h') \circ c = \mathsf{id}\,.$$

In other words, $(a \circ h')$ is a color-prescribed homomorphism from H to G and furthermore $b(a \circ h') = \mathcal{O}(h')$. Finally, we observe that for every $h \in \mathsf{cf\text{-}Hom}(H \to G)$ the stabilizer $\mathcal{S}(h)$ is trivial in the sense that it only contains id. By the Orbit-Stabilizer-Theorem (Theorem 2.29) we hence have that $|\mathcal{O}(h)| = \#\mathsf{Aut}(H)$. As $\mathsf{cf\text{-}Hom}(H \to G)$ is partitioned by its orbits we conclude that

$$\#\mathsf{cf\text{-}Hom}(H \to G) = \sum_{h \in \mathsf{cp\text{-}Hom}(H \to G)} |\mathcal{O}(h)| = \#\mathsf{Aut}(H) \cdot \#\mathsf{cp\text{-}Hom}(H \to G)\,.$$

■

Lemma 2.52. $\#\textsc{cf-Hom}(\mathcal{H}) \leq^{\mathrm{T}}_{\mathrm{fpt}} \#\textsc{Hom}(\mathcal{H})$.

We decided to give two proofs of the above lemma. One uses the inclusion-exclusion principle, and the other relies on multivariate polynomial interpolation.

Proof (by inclusion-exclusion). Let (H, G) be an instance of $\#\textsc{cf-Hom}(\mathcal{H})$, that is, $H \in \mathcal{H}$ and G is H-colored by a coloring c. In what follows, given a subset $J \subseteq V(H)$ we set

$$A_J := \{h \in \mathsf{Hom}(H \to G) \mid \forall j \in J : j \notin c(h(V(H)))\}\,,$$

that is, A_J is the set of all homomorphisms from H to G whose images do not contain vertices colored with some $j \in J$. Slightly abusing notation, we write A_j instead of $A_{\{j\}}$ for single colors $j \in V(H)$. Now observe that

$$|A_J| = \#\mathsf{Hom}(H \to G - J)\,,$$

where $G - J$ is the graph obtained from G by deleting all vertices that are colored with some $j \in J$. Furthermore, it holds that for every pair $I, J \subseteq V(H)$

$$A_I \cap A_J = A_{I \cup J}\,.$$

Finally, we apply the principle of inclusion and exclusion (Theorem 2.27) and obtain the following; note that the complement $\overline{A_j}$ is taken over the universe $\mathsf{Hom}(H \to G)$.

$$\begin{aligned}\#\mathsf{cf\text{-}Hom}(H \to G) &= \#\left(\bigcap_{j\in V(H)} \overline{A_j}\right)\\ &= \sum_{J\subseteq V(H)} (-1)^{|J|}\cdot |A_J|\\ &= \sum_{J\subseteq V(H)} (-1)^{|J|}\cdot \#\mathsf{Hom}(H \to G - J)\,.\end{aligned}$$

In particular, the latter sum can be computed in time $2^k \cdot |V(G)|$, provided oracle access to $\#\textsc{Hom}(\mathcal{H})$. ■

Proof (by polynomial interpolation). Let (H, G) and c as in the previous proof and let $V(H) = [k]$. Now, given some $\mathbf{x} \in [k]^k$, we define $G[\mathbf{x}]$ to be the graph obtained from G by cloning each vertex of color $i \in [k]$ precisely $\mathbf{x}_i$ times. Here, "cloning" a vertex v is the operation of adding a new vertex v' and edges $\{v', u\}$ for every vertex u that is adjacent to the primal vertex v. In particular, we extend the coloring c by setting $c(v') = c(v)$ for each clone v' of v. Note that $G[\mathbf{0}] = G$. Now given a (not necessarily colorful) homomorphism $h \in \mathsf{Hom}(H \to G[\mathbf{x}])$ for some $\mathbf{x} \in [k]^k$, we define its *color vector* $d(h) \in [k]^k$ by

$$d(h)_i := \#\{v \in V(H) \mid c(h(v)) = i\}\,,$$

that is, $d(h)_i$ is the number of vertices of H that are mapped to a (clone of some) vertex of color i in G. Next we partition the set $\mathsf{Hom}(H \to G[\mathbf{x}])$ by its color vectors and obtain

$$\#\mathsf{Hom}(H \to G[\mathbf{x}]) = \sum_{\mathbf{d}\in[k]^k} \#\{h \in \mathsf{Hom}(H \to G[\mathbf{x}]) \mid d(h) = \mathbf{d}\}\,.$$

Now observe that

$$\#\{h \in \mathsf{Hom}(H \to G[\mathbf{x}]) \mid d(h) = \mathbf{d}\} = \mathbf{x}^{\mathbf{d}}\cdot\#\{h \in \mathsf{Hom}(H \to G) \mid d(h) = \mathbf{d}\}\,,$$

where $\mathbf{x}^{\mathbf{d}}$ is the product $\prod_{i=1}^{k} \mathbf{x}_i^{\mathbf{d}_i}$. Hence $\#\mathsf{Hom}(H \to G[\mathbf{x}])$ is a polynomial given by

$$\#\mathsf{Hom}(H \to G[\mathbf{x}]) = \sum_{\mathbf{d}\in[k]^k} \#\{h \in \mathsf{Hom}(H \to G) \mid d(h) = \mathbf{d}\} \cdot \mathbf{x}^{\mathbf{d}}$$

and can be interpolated in time $k^{O(k)} \cdot |V(G)|$ using Grid-Interpolation (Theorem 2.28), provided oracle access to $\#\textsc{Hom}(\mathcal{H})$. Finally, we point out that the coefficient of $\mathbf{x}^{\mathbf{1}}$ satisfies

$$\#\{h \in \mathsf{Hom}(H \to G) \mid d(h) = \mathbf{1}\} = \#\mathsf{cf\text{-}Hom}(H \to G)\,.$$

■

Proof (of Lemma 2.49). Follows by Lemma 2.51 and Lemma 2.52. ∎

We proceed with the backward direction of the interreducibility result.

Lemma 2.53. $\#\textsc{Hom}(\mathcal{H}) \leq^{\mathrm{T}}_{\mathrm{fpt}} \#\textsc{cp-Hom}(\mathcal{H})$.

Proof. Let (H, G) be an instance of $\#\textsc{Hom}(\mathcal{H})$, that is $H \in \mathcal{H}$ and G is an arbitrary graph. It will be convenient to assume that $V(H) = [k]$. Given G, we construct a graph $\hat{G}$ as follows. The vertex set of $\hat{G}$ is defined to be

$$V(\hat{G}) = \bigcup_{i=0}^{k-1} V_i\,,$$

where $V_i = \{v_i \mid v \in V(G)\}$ is a copy of $V(G)$ identified by vertex $i \in V(H)$. We add an edge $\{u_i, v_j\}$ to $\hat{G}$ if and only if $\{i, j\} \in E(H)$ and $\{u, v\} \in E(G)$. Now it can easily be verified that the function $c : V(\hat{G}) \to V(H)$ given by $c(v_i) := i$ is an H-coloring of $\hat{G}$. Furthermore it is easy to see that

$$\#\mathsf{cp\text{-}Hom}(H \to \hat{G}) = \#\mathsf{Hom}(H \to G)\,,$$

which concludes the reduction. ∎

We are finally able to prove the second case of the classification for counting homomorphisms.

Theorem 2.54. *Let $\mathcal{H}$ be a recursively enumerable class of graphs. If the treewidth of $\mathcal{H}$ is unbounded, then $\#\textsc{Hom}(\mathcal{H})$ is $\#\mathsf{W}[1]$-equivalent.*

Proof. Hardness for $\#\mathsf{W}[1]$ is given by the following sequence of reductions.

$$\#\textsc{Clique} \overset{\text{Lemma 2.45}}{\leq^{\mathrm{T}}_{\mathrm{fpt}}} \#\textsc{cp-Hom}(\boxplus) \overset{\text{Corollary 2.48}}{\leq^{\mathrm{T}}_{\mathrm{fpt}}} \#\textsc{cp-Hom}(\mathcal{H}) \overset{\text{Lemma 2.49}}{\leq^{\mathrm{T}}_{\mathrm{fpt}}} \#\textsc{Hom}(\mathcal{H})$$

Note that the precondition of Corollary 2.48 states that every grid $\boxplus_k$ is a minor of some graph $H \in \mathcal{H}$. However, this follows by the Excluded-Grid-Theorem (see Theorem 2.6 and Theorem 2.7) and the fact that $\boxplus_{k'}$ is a minor of $\boxplus_k$ whenever $k' \leq k$.
For $\#\mathsf{W}[1]$-equivalence it remains to show that $\#\textsc{Hom}(\mathcal{H}) \leq^{\mathrm{T}}_{\mathrm{fpt}} \#\textsc{Clique}$. To this end, we let $\mathcal{K}$ be the set of all cliques and observe that $\#\textsc{Clique}$ and $\#\textsc{Hom}(\mathcal{K})$ are equivalent as the number of k-cliques in a graph G (without self-loops) is equal to $(k!)^{-1} \cdot \#\mathsf{Hom}(K_k \to G)$. Hence we have that

$$\#\textsc{Hom}(\mathcal{H}) \overset{\text{Lemma 2.53}}{\leq^{\mathrm{T}}_{\mathrm{fpt}}} \#\textsc{cp-Hom}(\mathcal{H}) \overset{\text{Corollary 2.48}}{\leq^{\mathrm{T}}_{\mathrm{fpt}}} \#\textsc{cp-Hom}(\mathcal{K}) \overset{\text{Lemma 2.49}}{\leq^{\mathrm{T}}_{\mathrm{fpt}}} \#\textsc{Hom}(\mathcal{K})\,,$$

where Corollary 2.48 is applicable as every graph H is the minor of some complete graph. ∎

We conclude this section by combining the $\#\mathsf{W}[1]$-equivalence result above with the dynamic programming algorithm for the case of bounded treewidth.

Proof (of Theorem 2.34). Holds by Theorem 2.54 and Corollary 2.43. ∎

2.6 First-Order Logic

In the last section of this chapter we will introduce logical generalizations of graphs and their related notions. To this end, we will consider logical structures and first-order formulas which constitute the mathematical foundations of relational databases and queries. This will allow us to lift existing and novel complexity classifications obtained in this thesis from graphs to the more general setting of (relational) model counting problems. In particular, building upon the work of Chen, Durand and Mengel [58, 31, 32] we will provide an exhaustive classification for the problem of counting answers to conjunctive queries and linear combinations thereof in Chapt. 6. We will mainly follow the notation of [65, Chapt. 4.2] in the subsequent definitions.

Definition 2.55. A *signature* τ is a finite tuple of *relation symbols* $(E_i)_{i\in[\ell]}$ with arities $(\mathsf{a}_i)_{i\in[\ell]}$. We set $\mathsf{a}(\tau) = \max\{\mathsf{a}_i \mid i \in [\ell]\}$ to be the arity of τ.

A *structure* H with signature τ consists of a finite set of vertices $V(\mathrm{H})$ and sets of (hyper-)edges $E_i(\mathrm{H}) \subseteq V(\mathrm{H})^{\mathsf{a}_i}$ for every $i \in [\ell]$.

The *complementary structure* $\overline{\mathrm{H}}$ of H has vertices $V(\mathrm{H})$ and for every $i \in [\ell]$ and every $\vec{e} \in V(\mathrm{H})^{\mathsf{a}_i}$ it holds that $\vec{e} \in E_i(\overline{\mathrm{H}})$ if and only $\vec{e} \notin E_i(\mathrm{H})$. Given structures H and F over the same signature τ, we say that F is a *substructure* of H if $V(\mathrm{F}) \subseteq V(\mathrm{H})$ and $E_i(\mathrm{F}) \subseteq E_i(\mathrm{H})$ for every $i \in [\ell]$.

Given two structures H and F with signature τ, a *homomorphism* from H to F is a function $h : V(\mathrm{H}) \to V(\mathrm{F})$ such that the following holds

$$\forall i \in [\ell] : \forall \vec{e} \in E_i(\mathrm{H}) : h(\vec{e}) \in E_i(\mathrm{F})\,,$$

where $h(\vec{e}) = (h(\vec{e}_i))_{i\in[\ell]}$. We denote $\mathsf{Hom}(\mathrm{H} \to \mathrm{F})$ as the set of all homomorphisms from H to F. Now the notions of isomorphisms, endomorphisms and automorphisms, as well as color-prescribed and colorful homomorphisms (see Definitions 2.44 and 2.50) are defined similarly as in case of graphs.

Example 2.56. Let $\tau = (E)$ such that $\mathsf{a}(E) = 2$. Then the set of structures with signature τ is precisely the set of directed graphs. If we consider the subset of structures H such that additionally $E(\mathrm{H})$ is symmetric and irreflexive, then this set is precisely the set of undirected graphs without self-loops. In this case the notions of homomorphisms coincide.

We will be particularly interested in connectivity measures of what is called the Gaifman graph of a structure, which is defined as follows.

Definition 2.57 (Gaifman graph). Given a structure H over some signature $\tau = (E_i)_{i\in[\ell]}$, the *Gaifman graph* $\mathbb{G}(\mathrm{H})$ of H has vertices $V(\mathrm{H})$ and contains an edge $\{u, v\}$ if and only if $u \neq v$ and u and v contained in a common edge of H, i.e., there exists $i \in [\ell]$ and $\vec{e} \in E_i(\mathrm{H})$ such that $u = \vec{e}_j$ and $v = \vec{e}_k$ for some $j, k \in [\mathsf{a}_i]$.

$$\frac{\alpha = E_i(z_j)_{j\in[\mathsf{a}_i]} \qquad (\beta(z_j))_{j\in[\mathsf{a}_i]} \in E_i(\mathrm{H})}{\mathrm{H} \models_\beta \alpha} \qquad \frac{\mathrm{H} \not\models_\beta \varphi}{\mathrm{H} \models_\beta \neg\varphi}$$

$$\frac{\mathrm{H} \models_{\beta|_{\mathsf{free}(\varphi)}} \varphi \qquad \mathrm{H} \models_{\beta|_{\mathsf{free}(\psi)}} \psi}{\mathrm{H} \models_\beta \varphi \wedge \psi} \qquad \frac{\mathrm{H} \models_{\beta|_{\mathsf{free}(\varphi)}} \varphi}{\mathrm{H} \models_\beta \varphi \vee \psi} \qquad \frac{\mathrm{H} \models_{\beta|_{\mathsf{free}(\psi)}} \psi}{\mathrm{H} \models_\beta \varphi \vee \psi}$$

$$\frac{\exists v \in V(H) : \mathrm{H} \models_{\beta\cup\{z\mapsto v\}} \varphi}{\mathrm{H} \models_\beta \exists z : \varphi} \qquad \frac{\forall v \in V(H) : \mathrm{H} \models_{\beta\cup\{z\mapsto v\}} \varphi}{\mathrm{H} \models_\beta \forall z : \varphi}$$

Figure 2.7: Inductive definition of the semantics of first-order formulas. Note that the term $\mathrm{H} \not\models_\beta \varphi$ in the top right rule is syntactic sugar for "$\mathrm{H} \models_\beta \varphi$ is false".

Now let $\tau = (E_i)_{i\in[\ell]}$ be a fixed signature and let $\mathcal{V}$ be a countably infinite set of variables. An *atom* of τ is of the form

$$E_i(z_j)_{j\in[\mathsf{a}_i]},$$

where $i \in [\ell]$ and $z_j \in \mathcal{V}$ for all $j \in [\mathsf{a}_i]$. The set of *first-order formulas* over τ is defined inductively as follows

$$\varphi, \psi ::= \alpha \mid \neg\varphi \mid \varphi \wedge \psi \mid \varphi \vee \psi \mid \exists z : \varphi \mid \forall z : \varphi\,,$$

where α is an atom and $z \in \mathcal{V}$. We may assume w.l.o.g. that a variable z is used at most once in a quantification $\exists z : \varphi$ or $\forall z : \varphi$ — otherwise we rename consistently. Now let φ be a first-order formula and let z be a variable contained in an atom of φ. We say that z is *free* if it was not introduced by a quantification $\exists z$ or $\forall z$ before. We write $\mathsf{free}(\varphi)$ for the set of all free variables of φ. Otherwise z is called *quantified.* Furthermore, we call φ *closed* if it contains no free variables and we call it *quantifier-free* if it contains no quantifications.

Now let H be a structure and let φ be a first-order formula, both over the same signature. Furthermore, let β be a function mapping free variables of φ to vertices in $V(\mathrm{H})$. Then the predicate $\mathrm{H} \models_\beta \varphi$ is defined inductively as given by the inference rules in Figure 2.7. Given a formula φ we define

$$\varphi(\mathrm{H}) := \{\beta : \mathsf{free}(\varphi) \to V(\mathrm{H}) \mid H \models_\beta \varphi\}\,.$$

For notational convenience, we will only write the last colon if multiple quantifications occur successively, that is, we write

$$\exists z_1 \exists z_2 \dots \exists z_\ell : \varphi$$

instead of

$$\exists z_1 : \exists z_2 : \dots \exists z_\ell : \varphi\,.$$

A formula is in *negation normal form* if every occurrence of $\neg$ is in front of an atom. Furthermore, a first-oder formula φ is in *prenex normal form* if

$$\varphi = Q_1 y_1 Q_2 y_2 \dots Q_\ell y_\ell : \psi \,,$$

where $Q_i \in \{\exists, \forall\}$ and ψ is quantifier-free. It will be very convenient to emphasize the set $\mathsf{free}(\varphi)$ by writing

$$\varphi = x_1 x_2 \dots x_k Q_1 y_1 Q_2 y_2 \dots Q_\ell y_\ell : \psi \,,$$

where the x_i are the free variables of φ. From now on we will assume w.l.o.g. that all first-order formulas are both, in prenex and negation normal form, the former assumption of which allows us to define the following hierarchy.

Definition 2.58. The set of all quantifier-free first-order formulas is denoted by both, Π_0 and Σ_0. A formula φ is contained in Σ_{t+1} if

$$\varphi = x_1 x_2 \dots x_k \exists y_1 \exists y_2 \dots \exists y_\ell : \psi \quad \text{, where } \psi \in \Pi_t \,,$$

and a formula φ is contained in Π_{t+1} if

$$\varphi = x_1 x_2 \dots x_k \forall y_1 \forall y_2 \dots \forall y_\ell : \psi \quad \text{, where } \psi \in \Sigma_t \,.$$

First-order formulas contained in Π_1 and Σ_1 are called universal and existential formulas, respectively. A formula is called *positive* if it contains no negations. We will be particularly interested in *conjunctive queries*, also known as primitive positive formulas, which are existential formulas of the form

$$\varphi = x_1 x_2 \dots x_k \exists y_1 \exists y_2 \dots \exists y_\ell : \alpha_1 \wedge \alpha_2 \wedge \dots \wedge \alpha_m \,,$$

where the α_i are atoms.

Now let Φ be a set of first-order formulas. Following the notation of [65], we define #p-MC(Φ) as the problem of, given $\varphi \in \Phi$ and a structure H over the same signature, computing $\#\varphi(\mathrm{H})$; the problem is parameterized by the length of φ. What follows is an alternative criterion for #W[1]-equivalence that will be very useful in Chapt. 6.

Theorem 2.59 ([64]). *#p-MC(Π_0) is #W[1]-equivalent.*

In particular, a stronger version of the above theorem states that the class #W[1] is actually equal to a class called #A[1] (see Theorem 14.17 in [65]). The latter is the first level of the #A-Hierarchy, which should be considered as a parameterized counting equivalent of the polynomial-time hierarchy PH. We will encounter, and define, the #A-classes in Chapter 6 when we discuss the complexity of counting answers to existential first-order formulas.

Chapter 3

Quantum Graphs

Unarguably, the most important objects of study in this thesis are so-called *quantum graphs.* Despite the name though, those objects have no immediate connections to quantum computing, at least to the best of the author's knowledge. In particular, we avoid confusion and point out that the term "quantum graph" refers to both, physical networks equipped with what is called a pseudo-differential operator [90] as well as to formal linear combinations of ordinary graphs as coined by Lovász [95, Chapter 6.1]. In this work, **we consider the latter** and provide a formal introduction in the subsequent sections.

3.1 Basic Definitions

We follow the notation of the textbook of Lovász — see Chapt. 6 in [95].

Definition 3.1 (Quantum graphs). A *quantum graph* is a formal linear combination of simple graphs without self-loops with finite support and rational coefficients. We write

$$Q = \sum \lambda_H \cdot H \,,$$

where the sum is over all (isomorphism types of) graphs and $\lambda_H \in \mathbb{Q}$ is zero for all but finitely many H. We write $\mathsf{supp}(Q)$ for the support of Q, that is, the set containing all graphs with a non-zero coefficient in Q. Furthermore, we call graphs in $\mathsf{supp}(Q)$ *constituents* of Q.

The functions we have seen in the preceding section extend to quantum graphs linearly. We make it explicit for the case of homomorphisms:

$$\#\mathsf{Hom}(Q \to G) := \sum \lambda_H \cdot \#\mathsf{Hom}(H \to G) \,.$$

The study of homomorphism numbers of quantum graphs has far-reaching consequences for the complexity classification of counting problems that require to count small structures in large graphs. They were first used in this context by Curticapean, Dell and Marx in a breakthrough result regarding the exact complexity of the subgraph counting problem [41]. Roughly speaking, they showed how to compute a quantum graph $Q[H]$, given a graph H, such that

$$\#\mathsf{Hom}(Q[H] \to \star) = \#\mathsf{Sub}(H \to \star)\,.$$

After that, they established that computing the number of homomorphisms from a quantum graph is *precisely as hard* as counting homomorphisms from the hardest constituent. This property, which seems to be unique for homomorphisms numbers from quantum graphs, is referred to as *complexity monotonicity* and allows to obtain exhaustive complexity classifications for counting problems that can be expressed as homomorphisms numbers from quantum graphs by ultimately relying on the fact that the problem of counting homomorphisms is well understood (see Theorem 2.34).

The remainder of this chapter is dedicated to a careful exposition of the strategy outlined above. We give an explicit example of the problem of counting matchings of size 3, which is shown to be precisely as hard as counting triangles of size 3 using the framework of quantum graphs. After that, we will provide a formal statement and a proof of the complexity monotonicity property, which is then invoked to present a self-contained proof of the result in [41]. We emphasize that, except for the example of counting matchings, the current chapter does not present new contributions but rather illustrates the most important techniques necessary for our treatment of quantum graphs in the subsequent chapters. From Chapt. 4 on, we then invoke and significantly refine those principles to obtain exhaustive complexity classification for counting problems that considerably generalize counting of subgraphs and induced subgraphs.

3.2 Linear Combinations of Homomorphisms

Recall that we wish to express the number of subgraphs as a linear combination of homomorphism numbers, that is, as a quantum graph. To this end, given a graph H, we define the quantum graph $Q[H] = \sum \lambda_{\hat{H}} \hat{H}$ by

$$\lambda_{\hat{H}} := \#\mathsf{Aut}(H)^{-1} \cdot \sum_{\substack{\rho \in L \\ H/\rho \,\simeq\, \hat{H}}} \mu_L(\bot, \rho)\,, \tag{3.1}$$

where L is the lattice of partitions of $V(H)$. Indeed, we already introduced all machinery we need to prove the following lemma.

Lemma 3.2. $\#\mathsf{Hom}(Q[H] \to \star) = \#\mathsf{Sub}(H \to \star)$.

Proof. Recall Equation (2.4) which was shown by Möbius inversion. For every graph G we have that

$$\begin{aligned}\#\mathsf{Sub}(H \to G) &\overset{\text{Fact 2.9}}{=} \#\mathsf{Aut}(H)^{-1} \cdot \#\mathsf{Emb}(H \to G)\\ &\overset{(2.4)}{=} \#\mathsf{Aut}(H)^{-1} \cdot \textstyle\sum_{\rho \in L} \mu_L(\bot, \rho) \cdot \#\mathsf{Hom}(H/\rho \to G)\,.\end{aligned}$$

The claim then follows by collecting for isomorphic quotient graphs and deleting terms $\#\mathsf{Hom}(H/\rho \to G)$ for which H/ρ contains a self-loop. Note that the latter is valid as G does not have self-loops and there are no homomorphisms from a graph with self-loops to a graph without self-loops. ∎

The next step is to decide which graphs $\hat{H}$ occur as constituents in $Q[H]$, that is, for which graphs $\hat{H}$ the coefficient $\lambda_{\hat{H}}$ is non-zero. It will turn out that this step is usually the much more involved one. In the current case however, one can rely on the explicit formula of the Möbius function over the partition lattice as done by Curticapean, Dell and Marx [41, Section 3.1]. We will give an alternative proof using Rota's NBC Theorem (Theorem 2.26) and matroid lattices, which is motivated by the fact that our proof can be generalized to partially injective homomorphisms (see Chapt. 4.1).

Lemma 3.3. *$\lambda_{\hat{H}} \neq 0$ whenever $\hat{H} \simeq H/\rho$ for some partition ρ of $V(H)$.*

Proof. We rely on the fact that the partition lattice of $V(H)$ is equivalent to the lattice of flats of the graphic matroid of the complete graph with vertices $V(H)$. Recall that a flat of a graphic matroid of a graph F is a subset of edges A such that for every edge $e \in E(F) \setminus A$ we have that

$$\mathsf{rk}(A \cup \{e\}) = \mathsf{rk}(A) + 1\,.$$

In other words, as $\mathsf{rk}(A) = |V(F)| - \mathsf{comp}(F[\rho])$ (see Chapt. 2.4.1), adding e to A must decrease the number of connected components by one; recall that every isolated vertex constitutes a single connected component. For the complete graph with vertices $V(H)$ we hence have that the flats correspond one-to-one to partitions of the graph in vertex-disjoint cliques; again, isolated vertices constitute cliques of size one. In particular those partitions in cliques are precisely the partitions of $V(H)$. This induces a natural bijection b between flats and partitions of $V(H)$. In particular, the bijection satisfies that

$$b(A) \leq b(B) \Leftrightarrow A \subseteq B\,,$$

where "$\leq$" is the partial ordering of the partition lattice and "$\subseteq$" is subset inclusion, that is, the partial ordering of the lattice of flats. Therefore, the two lattices are indeed the same up to renaming the elements. If this is the case, then the Möbius functions coincide and we have that

$$\lambda_{\hat{H}} = \#\mathsf{Aut}(H)^{-1} \cdot \sum_{\substack{A \in L' \\ H/b(A)\, \simeq\, \hat{H}}} \mu_{L'}(\emptyset, A)\,, \tag{3.2}$$

where L' is the lattice of flats of the complete graph with vertices $V(H)$. Now observe that $H/b(A) \simeq H/b(B)$ implies that $\mathsf{rk}(A) = \mathsf{rk}(B)$, as otherwise the number of vertices would not coincide. Finally, we can apply Theorem 2.26 and obtain that every term $\mu_{L'}(\emptyset, A)$ in (3.2) has the same sign. Hence $\lambda_{\hat{H}}$ is non-zero if there is at least one partition of ρ of $V(H)$ for which the quotient graph H/ρ becomes isomorphic to $\hat{H}$. ∎

We remark that, in the above proof, we were only required to formally rely on the bijection b as $\lambda_{\hat{H}}$ was defined as a sum over Möbius functions over the partition lattice. It is of course possible to define the quantum graph $Q[H]$ and its coefficients immediately over the lattice of flats. However, this would require us to prove Lemma 3.2 by Möbius inversion over the lattice of flats as well. We will generalize Lemma 3.2 to the more general case of partially injective homomorphisms in Chapt. 4.1 and in proving the generalization we will indeed rely on the lattice of flats.

Expressing a counting problem as $\#\mathsf{Hom}(Q \to \star)$ for some quantum graph Q immediately yields a naive algorithm: On input G, just compute $\#\mathsf{Hom}(H \to G)$ using Theorem 2.35 for every constituent H of Q and then compute the linear combination as given by Q. Let us provide an example in case of subgraph counting.

Theorem 3.4 (Corollary 1.2 in [41]). *Given a graph H with k edges and a graph G with n vertices, we can compute $\#\mathsf{Sub}(H \to G)$ in time*

$$f(|V(H)|) \cdot n^{0.174 \cdot k + o(k)},$$

where f is a computable function independent of G.

Proof. We compute by brute-force the quantum graph $Q[H]$ including the coefficients $\lambda_{\hat{H}}$ (3.1) for all $\hat{H}$ that are isomorphic to H/ρ for some partition ρ of $V(H)$. Now observe that every constituent $\hat{H}$ of $Q[H]$ has at most k edges. Scott and Sorkin [121, Corollary 21] established that the treewidth of a graph with k edges is bounded by $0.174 \cdot k + o(k)$. Hence we can apply Bodlaender's algorithm [14] to obtain a tree decomposition of width at most $0.174 \cdot k + o(k)$ in time $k^{O(k^3)} \cdot |V(H)|$. In particular, the number of nodes of this tree decomposition is bounded by $k^{O(k^3)} \cdot |V(H)|$ as well. Next we invoke Theorem 2.35 to obtain $\#\mathsf{Hom}(\hat{H} \to G)$ for every constituent $\hat{H}$ in time

$$\mathsf{poly}(|V(H)|, k) \cdot n^{0.174 \cdot k + o(k) + O(1)} = \mathsf{poly}(|V(H)|) \cdot n^{0.174 \cdot k + o(k)}.$$

Finally, given those numbers, we compute the linear combination induced by $Q[H]$ which, by Lemma 3.2, equals $\#\mathsf{Sub}(H \to G)$. Clearly, the overall running time is bounded by

$$f(|V(H)|) \cdot n^{0.174 \cdot k + o(k)}.$$

∎

Furthermore, we emphasize a fixed-parameter tractability result for the parameterized subgraph counting problem. To this end, given a class of graphs $\mathcal{H}$, we define $\#\textsc{Sub}(\mathcal{H})$ as the problem of, given a graph $H \in \mathcal{H}$ and an arbitrary graph G, computing $\#\mathsf{Sub}(H \to G)$, that is, the number of subgraphs of G that are isomorphic to H. The problem is parameterized by $|V(H)|$. Moreover, recall that a vertex cover of a graph H is a set $S \subseteq V(H)$ such that every edge of H is incident to at least one vertex in S. The *vertex cover number* of H is the size of the smallest vertex cover of H and we say that a class of graphs $\mathcal{H}$ has *bounded vertex cover number* if there is a constant d such that the vertex cover number of every graph in $\mathcal{H}$ is at most d.

Lemma 3.5. *Let $\mathcal{H}$ be a class of graphs of bounded vertex cover number. Then $\#\textsc{Sub}(\mathcal{H})$ is fixed-parameter tractable.*

Proof. First of all, note that the treewidth of a graph H with vertex cover number d is at most d: Let S be a vertex cover of size d of H. We make S a bag of the tree decomposition and further add bags $S \cup \{v\}$ for all vertices $v \in V(H) \setminus S$. Finally we connect the vertices of the tree corresponding to S and $S \cup \{v\}$ for every $v \in V(H) \setminus S$. It can easily be verified that this construction yields a tree decomposition of width $|S|$; recall the "-1" in the definition of treewidth.

Next we observe that the vertex cover number of H/ρ cannot be larger than the vertex cover number of H for every partition ρ of H. Hence, given an instance (H, G) of $\#\textsc{Sub}(\mathcal{H})$ we have that the treewidth of all graphs H/ρ is bounded by the constant upper bound d on the vertex cover numbers of graphs in $\mathcal{H}$. Similarly to the proof of Theorem 3.4 we apply dynamic programming over the tree decompositions of the constituents of $Q[H]$ using Theorem 2.35 which allows us to compute the linear combination of homomorphisms and hence the number $\#\mathsf{Sub}(H \to G)$ in time

$$f(|V(H)|) \cdot |V(G)|^{d+O(1)} . \qquad \blacksquare$$

Remark 3.6 ([91, 138, 43]). $\#\textsc{Sub}(\mathcal{H})$ is even solvable in polynomial time if the vertex cover number of $\mathcal{H}$ is bounded.

We have seen that the parameterized problem of counting subgraphs is fixed-parameter tractable if there exists a constant upper bound on the vertex cover number of the subgraphs we which to count. We obtained this result by a quite naive algorithm using the framework of quantum graphs: Just rely on the known treewidth-based DP algorithm to compute the homomorphism numbers of all constituents. The obvious next question is whether the constant bound on the vertex cover number is also *necessary* for the problem to be fixed-parameter tractable. It turns out that this holds true and, in particular, the proof shows the full power of the framework of quantum graphs. More precisely, we will show that the complexity of computing $\#\mathsf{Hom}(Q \to \star)$ for some quantum graph Q is not only upper bounded *but also lower bounded* by the complexity of counting homomorphisms from its hardest constituent; the property which was previously advertised as *complexity monotonicity.* The principle is first illustrated in case of the 3-matching.

3.2.1 Matchings and Triangles

In the current section, we write $\equiv$ for the matching of size 3 and $\lhd$ for the clique of size 3. We will establish that counting subgraphs isomorphic to $\equiv$ is precisely as hard as counting triangles.

Theorem 3.7. *Let $c \in \mathbb{R}$ be a fixed real. Then the following statements are equivalent:*

(1) $\#\mathsf{Sub}(\equiv \to \star)$ can be computed in time $O(n^c)$.

(2) $\#\mathsf{Hom}(\lhd \to \star)$ can be computed in time $O(n^c)$.

Here n denotes the number of vertices of the input graph. In particular, the equivalence holds true for combinatorial algorithms.[1]

Let us first discuss some consequences of the above result. As the decision version of $\#\mathsf{Hom}(\lhd \to \star)$ is equivalent to the problem of triangle-detection in a graph,[2] we obtain that any algorithm computing $\#\mathsf{Sub}(\equiv\to \star)$ in time $O(n^c)$ can be used to find a triangle in time $O(n^c)$. In particular, if $\#\mathsf{Sub}(\equiv\to \star)$ can be computed by a combinatorial algorithm in time $O(n^{3-\varepsilon})$ for some $\varepsilon > 0$, then there exists a $\delta > 0$ and a combinatorial algorithm that can perform boolean matrix multiplication of $n \times n$ matrices in time

$$O(n^{3-\delta} \cdot \mathsf{poly}(\mathsf{log}(M)))\,,$$

where M is the absolute value of the largest entry [139, Theorem 1.3].

Furthermore, we can use fast matrix multiplication to obtain a subcubic (non-combinatorial) algorithm for counting matchings of size 3.

Corollary 3.8. $\#\mathsf{Sub}(\equiv \to \star)$ *can be computed in time $O(n^\omega)$, where ω is the matrix multiplication exponent.*

Proof. Follows by Theorem 3.7 and the fact that $\#\mathsf{Hom}(\lhd \to \star)$ can be computed in time $O(n^\omega)$ (see e.g. [105]). ■

[1]A formal definition of combinatorial algorithms seems to be elusive. Roughly speaking, those algorithms do not rely on fast matrix multiplication and are hence more relevant for practical applications (see [8, Section 1] for a discussion).

[2]Every homomorphism from $\lhd$ to a graph without self-loops must be injective.

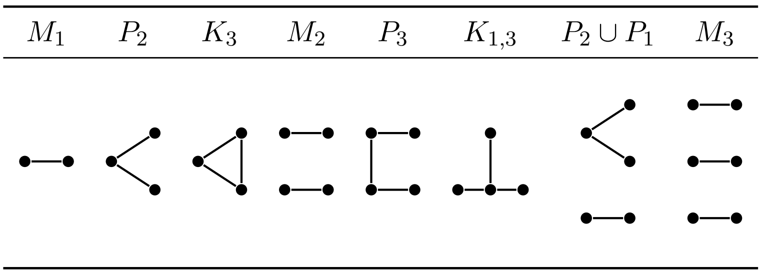


Figure 3.1: The constituents of the quantum graph $Q[\equiv]$.

Both directions of the proof of Theorem 3.7 will use the quantum graph $Q[\equiv]$ as defined in (3.1). By Lemma 3.2 we have that

$$\#\mathsf{Sub}(\equiv \to \star) = \#\mathsf{Hom}(Q[\equiv] \to \star)\,.$$

Now using Lemma 3.3, we can give the support of $Q[\equiv]$ explicitly as depicted in Figure 3.1; recall that quotient graphs with self-loops can be deleted. Note that all coefficients of $Q[\equiv]$ are known constants given by (3.1) for $H = \equiv$.

Observation 3.9. *Every constituent in* $\mathsf{supp}(Q[\equiv])$ *has treewidth* 1, *except for* $\triangleleft$, *which is of treewidth* 2.

Lemma 3.10. *Let $c \geq 2$ be a fixed real such that $\#\mathsf{Hom}(\triangleleft \to \star)$ can be computed in time $O(n^c)$. Then $\#\mathsf{Sub}(\equiv \to \star)$ can be computed in time $O(n^c)$ as well. Here n denotes the number of vertices of the input graph. In particular, this holds true for combinatorial algorithms.*

Proof. Given a graph G with n vertices, we compute $\#\mathsf{Hom}(Q[\equiv] \to G)$ by applying the treewidth-based DP for all constituents of treewidth 1 and use the assumed algorithm for $\#\mathsf{Hom}(\triangleleft \to \star)$. By Theorem 2.35 and Remark 2.42 the former can be done in time $O(n^2)$. Furthermore, the latter takes time $O(n^c)$. After that, we compute the linear combination, which takes time $O(\mathsf{log}(n))$. The total running time is hence bounded by $O(n^2 + n^c + \mathsf{log}(n)) = O(n^c)$. In particular, the algorithm is combinatorial if and only if the algorithm for computing $\#\mathsf{Hom}(\triangleleft \to G)$ is. ∎

Having established the easy direction, it remains to show that $\#\mathsf{Sub}(\equiv \to \star)$ is at least as hard as $\#\mathsf{Hom}(\triangleleft \to \star)$. To this end we will, at last, use a concrete application of complexity monotonicity. Recall that $G \times F$ denotes the tensor product of G and F, that is, the graph whose adjacency matrix is given by the Kronecker product of the adjacency matrices of G and F. It turns out that the function $\#\mathsf{Hom}(H \to \star)$ is linear on the operation $\times$.

Fact 3.11 (Equation (5.30) in [95]). *Let H, F and G be graphs. Then*

$$\#\mathsf{Hom}(H \to G \times F) = \#\mathsf{Hom}(H \to G) \cdot \#\mathsf{Hom}(H \to F)\,.$$

$$\begin{array}{c|cccccccc} & - & < & \lhd & = & \sqsubset & \perp & \leq & \equiv \\ \hline - & 2 & 4 & 6 & 4 & 6 & 6 & 6 & 6 \\ < & 2 & 6 & 12 & 4 & 10 & 12 & 8 & 6 \\ \lhd & 0 & 0 & 6 & 0 & 0 & 0 & 0 & 0 \\ = & 4 & 16 & 36 & 16 & 36 & 36 & 36 & 36 \\ \sqsubset & 2 & 8 & 24 & 4 & 16 & 18 & 10 & 6 \\ \perp & 2 & 10 & 24 & 4 & 18 & 30 & 12 & 6 \\ \leq & 4 & 24 & 72 & 16 & 60 & 72 & 48 & 36 \\ \equiv & 8 & 64 & 216 & 64 & 216 & 216 & 216 & 216 \end{array}$$

Figure 3.2: The matrix $M_{\#\mathsf{Hom}}$ for the constituents of $Q[\equiv]$. In particular, the entry corresponding to row F and column F' is precisely $\#\mathsf{Hom}(F \to F')$.

Lemma 3.12. *Let $c \geq 2$ be a fixed real such that $\#\mathsf{Sub}(\equiv \to \star)$ can be computed in time $O(n^c)$. Then $\#\mathsf{Hom}(\lhd \to \star)$ can be computed in time $O(n^c)$ as well. Here n denotes the number of vertices of the input graph. In particular, this holds true for combinatorial algorithms.*

Proof. Given a graph G with n vertices for which we want to compute $\#\mathsf{Hom}(\lhd \to G)$, the idea is to query $\#\mathsf{Sub}(\equiv \to \star) = \#\mathsf{Hom}(Q[\equiv] \to \star)$ multiple times on graphs $G \times F$. By Fact 3.11 we have that

$$\begin{aligned} \#\mathsf{Hom}(Q[\equiv] \to G \times F) \; = \; & \lambda_{-} \cdot \#\mathsf{Hom}(- \to G) \cdot \#\mathsf{Hom}(- \to F) \\ + \; & \lambda_{<} \cdot \#\mathsf{Hom}(< \to G) \cdot \#\mathsf{Hom}(< \to F) \\ + \; & \lambda_{\lhd} \cdot \#\mathsf{Hom}(\lhd \to G) \cdot \#\mathsf{Hom}(\lhd \to F) \\ + \; & \lambda_{=} \cdot \#\mathsf{Hom}(= \to G) \cdot \#\mathsf{Hom}(= \to F) \\ + \; & \lambda_{\sqsubset} \cdot \#\mathsf{Hom}(\sqsubset \to G) \cdot \#\mathsf{Hom}(\sqsubset \to F) \\ + \; & \lambda_{\perp} \cdot \#\mathsf{Hom}(\perp \to G) \cdot \#\mathsf{Hom}(\perp \to F) \\ + \; & \lambda_{\leq} \cdot \#\mathsf{Hom}(\leq \to G) \cdot \#\mathsf{Hom}(\leq \to F) \\ + \; & \lambda_{\equiv} \cdot \#\mathsf{Hom}(\equiv \to G) \cdot \#\mathsf{Hom}(\equiv \to F)\,, \end{aligned}$$

which yields a system of linear equations. In particular, we choose for F the 8 graphs in $\mathsf{supp}(Q[\equiv])$. The induced matrix is of size 8×8 and given by

$$M_{\#\mathsf{Hom}}(F, F') := \#\mathsf{Hom}(F \to F')\,,$$

where $F, F' \in \mathsf{supp}(Q[\equiv])$; consult Figure 3.2 for an explicit representation. Since $M_{\#\mathsf{Hom}}$ is non-singular, we can solve the system of linear equations in time $O(\log n)$. In particular, we obtain the coefficient of $\#\mathsf{Hom}(\lhd \to F)$ which, divided by $\lambda_{\lhd}$, yields $\#\mathsf{Hom}(\lhd \to G)$. As every graph $G \times F$ for $F \in \mathsf{supp}(Q[\equiv])$ has $O(n)$ vertices, the overall running time is bounded by

$$8 \cdot O(n^c) + O(\log n) = O(n^c)\,. \qquad \blacksquare$$

Proof (of Theorem 3.7). Follows by Lemma 3.10 and Lemma 3.12 and the fact that neither of $\#\mathsf{Sub}(\equiv \to \star)$ and $\#\mathsf{Hom}(\lhd \to \star)$ can be solved in time $O(n^c)$ for $c < 2$ as this would be sublinear in the input size — note that a graph on n vertices might have $\Omega(n^2)$ edges. $\blacksquare$

3.2.2 Complexity Monotonicity

We will now provide a general version of complexity monotonicity. To this end, the proof of Lemma 3.12 is lifted to arbitrary quantum graphs. The statement is presented in a form that is suitable for both, parameterized and fine-grained complexity lower bounds. We point out that the following theorem holds for arbitrary quantum graphs, not only for those defined in Equation (3.1).

Theorem 3.13 (Complexity monotonicity, cf. Lemma 3.6 in [41]). *Let $Q = \sum_{i\in[k]} \lambda_i \cdot H_i$ be a quantum graph with constituents $\{H_i\}_{i\in[k]}$. There exists an algorithm $\mathbb{A}$ that is given a graph G with $n > 0$ vertices as input and has oracle access to*

$$\#\mathsf{Hom}(Q \to \star)\,,$$

and computes $\#\mathsf{Hom}(H_i \to G)$ for all $i \in [k]$ in time $f(Q)\cdot n$. Additionally, every graph $\hat{G}$ that is posed as oracle query has at most $f(Q)\cdot n$ vertices. Here f is a computable function independent of G.

Proof. We will query the oracle for k tensor products $G \times F_j$ where $j \in [k]$. Using Fact 3.11 we have that

$$\#\mathsf{Hom}(Q \to G \times F_j) = \sum_{i\in[k]} \underbrace{\lambda_i \cdot \#\mathsf{Hom}(H_i \to G)}_{=:c_i} \cdot \#\mathsf{Hom}(H_i \to F_j)\,.$$

Similar to the proof of Lemma 3.12, this induces a system of linear equations. Unfortunately, the corresponding matrix might be singular for $F_j = H_j$ in the general case. However, it is known that we can find graphs for which non-singularity can be guaranteed.

Fact 3.14 (cf. Proposition 5.44 (b) in [95]). *For every positive $k \in \mathbb{N}$ and pairwise non-isomorphic graphs $\{H_i\}_{i\in[k]}$ there exist graphs $\{F_j\}_{j\in[k]}$ such that the $k \times k$-matrix given by*

$$M_{\#\mathsf{Hom}}(H_i, F_j) := \#\mathsf{Hom}(H_i \to F_j)$$

is non-singular.

In particular, the proof yields an algorithm,[3] which allows us to compute F_j and hence $G \times F_j$ for every $j \in [k]$ in time $f(Q)\cdot n$. Observe further, that the number of vertices of $G \times F_j$ is bounded by $f(Q)\cdot n$ as well. The coefficients c_i can therefore be computed using Gaussian elimination and we may output c_i/λ_i for every $i \in [k]$. ∎

[3]Roughly speaking, the graph F_j can be obtained from H_j by cloning its vertices in an appropriate way. Details can be found in [95, Chapt. 5.5].

Complexity monotonicity is a powerful tool for proving lower bounds on the complexity of (parameterized) counting problems. More precisely, whenever a counting problem can be expressed as counting homomorphisms from a quantum graph, the problem is at least as hard as counting homomorphisms from its hardest constituent. We provide a formal application in the proof of the following exhaustive classification of the subgraph counting problem, which is originally due to Curticapean and Marx.

Theorem 3.15 ([43]). *Let $\mathcal{H}$ be a recursively enumerable class of graphs.*

(1) If $\mathcal{H}$ has bounded vertex cover number then $\#\textsc{Sub}(\mathcal{H})$ is solvable in polynomial time.

(2) Otherwise $\#\textsc{Sub}(\mathcal{H})$ is $\#\mathsf{W}[1]$-equivalent.

Proof. The first item is Remark 3.6. Hence assume that $\mathcal{H}$ has unbounded vertex cover number. We define

$$\mathcal{Q} := \{F \in \mathsf{supp}(Q[H]) \mid H \in \mathcal{H}\}\,,$$

where $Q[H]$ is the quantum graph given by Equation (3.1). We claim that $\#\textsc{Sub}(\mathcal{H})$ and $\#\textsc{Hom}(\mathcal{Q})$ are interreducible with respect to parameterized Turing reductions.

Claim 3.16. $\#\textsc{Sub}(\mathcal{H}) \leq^{\mathrm{T}}_{\mathrm{fpt}} \#\textsc{Hom}(\mathcal{Q})$.

Proof. Given $H \in \mathcal{H}$ and a graph G, we construct $Q[H]$ and use the oracle for $\#\textsc{Hom}(\mathcal{Q})$ to compute

$$\sum_{F \in \mathsf{supp}(Q[H])} \lambda_F \cdot \#\mathsf{Hom}(F \to G) = \#\mathsf{Hom}(Q[H] \to G)\,,$$

which is equal to $\#\mathsf{Sub}(H \to G)$ by Lemma 3.2.. □

Claim 3.17. $\#\textsc{Hom}(\mathcal{Q}) \leq^{\mathrm{T}}_{\mathrm{fpt}} \#\textsc{Sub}(\mathcal{H})$.

Proof. Given $F \in \mathcal{Q}$ and a graph G, we search for a graph $H \in \mathcal{H}$ for which F is a constituent of $Q[H]$. This can be done by brute-force and is guaranteed to terminate as $\mathcal{H}$ is recursively enumerable. In particular, the running time and the number of vertices of H are bounded by $f(|V(F)|)$ for some computable function f independent of G. By Lemma 3.2 we have that

$$\#\mathsf{Sub}(H \to \star) = \#\mathsf{Hom}(Q[H] \to \star)\,.$$

Hence we can use the oracle access to $\#\mathsf{Sub}(H \to \star)$ to invoke Theorem 3.13 and obtain $\#\mathsf{Hom}(F \to G)$. □

It remains to show that $\mathcal{Q}$ has unbounded treewidth. If this is the case, we can invoke the classification for counting homomorphisms to obtain #W[1]-equivalence (see Theorem 2.34). By Lemma 3.3 we have that $\mathcal{Q}$ contains all loopless graphs H/ρ where $H \in \mathcal{H}$ and ρ is a partition of $V(H)$. Now observe that for every $k > 0$ there exists a graph in $\mathcal{H}$ that contains the matching M_k of size k as a subgraph, because otherwise the vertex cover number of $\mathcal{H}$ would be bounded. Furthermore, given a graph H containing a matching of size k it is easy to see that every graph with at most k edges (and no isolated vertices) is a minor of a loopless quotient graph of H: Endpoints of two edges of the matching are identified using the contraction operation if they are adjacent and they are identified using the quotient construction if they are not adjacent, which does hence not create a self-loop. A detailed construction is given by Curticapean, Dell and Marx [41, Fact 3.4.3].

Consequently, the set of minors of $\mathcal{Q}$ contains all complete graphs. As the treewidth cannot increase by taking a minor we conclude that the treewidth of $\mathcal{Q}$ is unbounded. ∎

Chapter 4

Constrained Homomorphisms

As we have seen, the complexity of counting homomorphisms and subgraphs is fully resolved, at least from a parameterized perspective (see Theorem 2.34 and Theorem 3.15). Subgraph counting is equivalent to counting injective homomorphisms. More precisely, we recall that for every pair of graphs H and G the following holds true.

$$\#\mathsf{Emb}(H \to G) = \#\mathsf{Aut}(H) \cdot \#\mathsf{Sub}(H \to G)\,.$$

In other words, the task of counting subgraphs adds injectivity as a constraint to the problem of counting homomorphisms. In the current chapter, we will interpolate between the problems of counting homomorphisms and counting injective homomorphisms.

One possible relaxation is partial injectivity. Roughly speaking, we will be interested in counting homomorphisms from a graph H to a graph G that additionally satisfy a set of partial injectivity constraints, also called "inequalities", that are part of the input. Those partial injective homomorphisms do not only generalize subgraph embeddings, but also locally injective homomorphisms. We will give a formal introduction in Chapter 4.1 and furthermore provide an explicit classification that allows to pinpoint the parameterized complexity for every possible class of allowed inequalities and graphs.

A further relaxation of full injectivity is the restriction to edge-injectivity only. More precisely, we will consider the problem of, given a graph H and a graph G, computing the number of homomorphisms from H to G that are injective on the edges of H. By this, we generalize problems like counting of edge-disjoint cycles and paths as well as matchings in line graphs. Chapter 4.2 provides a complete classification for this problem, depending on the class of allowed graphs for H. Unfortunately, we do not obtain an explicit criterion as in case of counting partially injective homomorphisms.

The reason for this is the fact that the support of the quantum graphs we use to express the number of edge-injective homomorphisms cannot be given explicitly. However, we are able to provide an explicit criterion in case the class of allowed graphs for H is closed under taking induced subgraphs.

The results of Chapter 4.1 have been published in [119]. The results of Chapter 4.2 have been obtained in collaboration with Radu Curticapean and Holger Dell at the Simons Institute for the Theory of Computing. They have been published in [42].

4.1 Partially Injective Homomorphisms

A *graph with inequalities* is a pair (H, I) where H is a graph and I is an irreflexive and symmetric relation $I \subseteq V(H)^2$. We say that I is a set of inequalities. Intuitively, given a graph G and a graph with inequalities (H, I), a homomorphism $h \in \mathsf{Hom}(H \to G)$ satisfies I if for every inequality $\{u, v\} \in I$, it holds that $h(u) \neq h(v)$. Formally, we define the set of homomorphisms that satisfy I in terms of partially injective homomorphisms

$$\mathsf{PartInj}(H, I \to G) := \{h \in \mathsf{Hom}(H \to G) \mid \forall \{u, v\} \in I : h(u) \neq h(v)\}\,.$$

Given a class $\mathcal{H}$ of graphs with inequalities, we write $\#\textsc{PartInj}(\mathcal{H})$ for the problem of, given $(H, I) \in \mathcal{H}$ and a graph G, computing $\#\mathsf{PartInj}(H, I \to G)$; the parameter is $|V(H)|$.

Now let (H, I) be graph with inequalities. It will be very convenient to assume that I does not contain an edge of H and, indeed, we can make this assumption without loosing generality. To see this, observe that whenever $\{u, v\}$ is an edge of H, no homomorphism in $\mathsf{Hom}(H \to G)$ can map u and v to the same vertex in G, because otherwise the image of u and v would have a self-loop. As we do not allow input graphs G with self-loops, every homomorphism h satisfies $h(u) \neq h(v)$, regardless of whether $\{u, v\} \in I$.

Our classification theorem requires us to generalize quotient graphs to sets of inequalities as follows. Let (H, I) be a graph with inequalities and let ρ be a subset of I. Now consider the graph with vertices $V(H)$ and edges ρ. The connected components of this graph induce a partition $\mathsf{Part}(\rho)$ of $V(H)$. In particular, every isolated vertex constitutes a singleton in the partition. We abuse notation and write H/ρ for the quotient graph $H/\mathsf{Part}(\rho)$ in what follows; consult Figure 4.1 for an illustration.

Graphs H/ρ for $\rho \subseteq I$ will turn out to be constituents of the quantum graph for (H, I) if they do not contain self-loops. For this reason, we adopt the notation of [41] and call a graph F a *spasm* of (H, I) if F does not contain self-loops and is isomorphic to a graph H/ρ for $\rho \subseteq I$. In other words, F is a spasm of (H, I) if it can be obtained from H by successively identifying vertices of H along inequalities in I without creating self-loops.

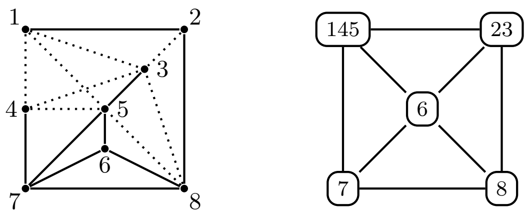


Figure 4.1: *Left:* A graph H (solid lines) with inequalities I (dotted lines). *Right:* The graph H/ρ for the subset $\rho = \{\{1,4\},\{1,5\},\{2,3\}\}$ of I. Note that the connected components of the graph with vertices $V(H)$ and edges ρ are precisely $\{1,4,5\}$, $\{2,3\}$, $\{6\}$, $\{7\}$ and $\{8\}$, which constitutes a partition of $V(H)$. As H/ρ does not contain a self-loop, it is a spasm of (H,I) and hence a constituent of the quantum graph $Q[H,I]$ by Theorem 4.1.

We write $\mathsf{Spasm}(H,I)$ for the set of all spasms of (H,I) and given a class $\mathcal{H}$ of graphs with inequalities, we define $\mathsf{Spasm}(\mathcal{H})$ to be the set containing every graph F that is the spasm of some $(H,I) \in \mathcal{H}$.

The following is the main result of the current section.

Theorem 4.1. *Let (H,I) be a graph with inequalities. Then there exists a quantum graph $Q[H,I]$ which satisfies*

$$\#\mathsf{PartInj}(H,I \to \star) = \#\mathsf{Hom}(Q[H,I] \to \star)\,.$$

In particular, the mapping $(H,I) \mapsto Q[H,I]$ is computable and

$$\mathsf{supp}(Q[H,I]) = \mathsf{Spasm}(H,I)\,.$$

Let us first emphasize the following consequence of Theorem 4.1 for the parameterized complexity of $\#\textsc{PartInj}(\mathcal{H})$.

Corollary 4.2. *Let $\mathcal{H}$ be a recursively enumerable class of graphs with inequalities.*

(1) If the treewidth of $\mathsf{Spasm}(\mathcal{H})$ is bounded, then $\#\textsc{PartInj}(\mathcal{H})$ is fixed-parameter tractable.

(2) Otherwise $\#\textsc{PartInj}(\mathcal{H})$ is $\#\mathsf{W}[1]$-equivalent.

Proof. We show that the problems $\#\textsc{PartInj}(\mathcal{H})$ and $\#\textsc{Hom}(\mathsf{Spasm}(\mathcal{H}))$ are interreducible with respect to parameterized Turing reductions.

Claim 4.3. $\#\textsc{PartInj}(\mathcal{H}) \leq^{\mathrm{T}}_{\mathrm{fpt}} \#\textsc{Hom}(\mathsf{Spasm}(\mathcal{H}))$.

Proof. We observe that by Theorem 4.1

$$\#\mathsf{PartInj}(H,I \to \star) = \#\mathsf{Hom}(Q[H,I] \to \star) = \sum_{F \in \mathsf{Spasm}(H,I)} \lambda_F \cdot \#\mathsf{Hom}(F \to \star)\,,$$

for computable coefficients λ_F. □

Claim 4.4. $\#\textsc{Hom}(\mathsf{Spasm}(\mathcal{H})) \leq^{\mathrm{T}}_{\mathrm{fpt}} \#\textsc{PartInj}(\mathcal{H})$.

Proof. Let $F \in \mathsf{Spasm}(\mathcal{H})$ and G be an instance of $\#\textsc{Hom}(\mathsf{Spasm}(\mathcal{H}))$. As $\mathcal{H}$ is recursively enumerable, we can find $(H, I) \in \mathcal{H}$ with $F \in \mathsf{Spasm}(H, I)$ in time only depending on F. By complexity monotonicity (Theorem 3.13) we can use the oracle for

$$\#\mathsf{PartInj}(H, I \to \star) = \#\mathsf{Hom}(Q[H, I] \to \star)$$

to compute $\#\mathsf{Hom}(F \to G)$. □

The corollary then follows by Theorem 2.34. ■

The proof of Theorem 4.1 consists of two steps. In the first one, we will invoke Möbius inversion over the lattice of flats of the graphic matroic induced by the inequalities. This will allow us to construct the quantum graph $Q[H, I]$. In the second step, we will then use Rota's NBC Theorem (Theorem 2.26) to prove that the constituents of $Q[H, I]$ are precisely the spasms of (H, I). We emphasize that those two steps are equal to the proofs of Lemma 3.2 and Lemma 3.3 in case I contains every pair of different vertices, that is, in case of the full injectivity constraint.

Starting with the first step, we consider the *inequality graph* $\mathbb{I}(H, I)$ of a given graph with inequalities (H, I). The vertices of $\mathbb{I}(H, I)$ are $V(H)$ and the edges are I. We write $M(H, I)$ for the graphic matroid of $\mathbb{I}(H, I)$ and we write $L(H, I)$ for the lattice of flats of $M(H, I)$. Recall from Chapter 2.4.1 that a flat of $M(H, I)$ is a subset of I satisfying that adding any element $i \in I \setminus \rho$ to ρ will decrease the number of connected components by 1. Now observe that, given $\rho \in I$, we can define $\mathbb{I}(H, I)/\rho$ similar to H/ρ, as both, $\mathbb{I}(H, I)/\rho$ and H have the same vertex set $V(H)$. In particular, we write I/ρ for the edges of $\mathbb{I}(H, I)/\rho$ and obtain that $(H/\rho, I/\rho)$ is a graph with inequalities as well. This allows us to prove correctness of the following zeta transformation.

Lemma 4.5. *Let (H, I) be a graph with inequalities and let σ be a flat of $M(H, I)$. Then*

$$\#\mathsf{Hom}(H/\sigma \to \star) = \sum_{\rho \geq \sigma} \#\mathsf{PartInj}(H/\rho, I/\rho \to \star)\,,$$

where the sum is over flats ρ of $M(H, I)$.

Proof. Let G be a graph. We define

$$\mathsf{Hom}(H \to G)[\sigma] := \{h \in \mathsf{Hom}(H \to G) \mid \forall \{u, v\} \in \sigma : h(u) = h(v)\}\,.$$

Observe that $\#\mathsf{Hom}(H/\sigma \to G) = \#\mathsf{Hom}(H \to G)[\sigma]$. We will now just partition the set $\mathsf{Hom}(H \to G)[\sigma]$ by the inequalities that are satisfied. To this end, given some $h \in \mathsf{Hom}(H \to G)[\sigma]$, we set

$$\rho(h) := \{\{u,v\} \in I \mid h(u) = h(v)\} \tag{4.1}$$

and claim that $\rho(h)$ is a flat of $M(H,I)$ which is greater than, i.e., a superset of σ. Assume for contradiction that $\rho(h)$ is not a flat. Then there exists $\{a,b\} \in I \setminus \rho(h)$ such that $\rho(h)$ and $\rho(h) \cup \{a,b\}$ induce the same connected components of $\mathbb{I}(H,I)$. Consequently, there is a path from a to b in $\mathbb{I}(H,I)$ only consisting of edges in $\rho(h)$. Inductively applying (4.1) yields $h(a) = h(b)$ and hence $\{a,b\} \in \rho(h)$ which contradicts the assumption.

This allows us to define an equivalence relation on $\mathsf{Hom}(H \to G)[\sigma]$ by setting $h \sim \hat{h}$ if $\rho(h) = \rho(\hat{h})$. In particular, the equivalence classes are uniquely identified by flats greater or equal to σ. Furthermore, the elements of a class $[\![\rho]\!]$ for $\rho \geq \sigma$ are precisely those homomorphisms $h \in \mathsf{Hom}(H \to G)$ satisfying

$$\forall \{u,v\} \in I : h(u) = h(v) \Leftrightarrow \{u,v\} \in \rho\,,$$

and consequently

$$\#[\![\rho]\!] = \#\mathsf{PartInj}(H/\rho, I/\rho \to G)\,.$$

It follows that

$$\begin{aligned}\#\mathsf{Hom}(H/\sigma \to G) = \#\mathsf{Hom}(H \to G)[\sigma] &= \sum_{\rho \geq \sigma} \#[\![\rho]\!] \\ &= \sum_{\rho \geq \sigma} \#\mathsf{PartInj}(H/\rho, I/\rho \to G)\,,\end{aligned}$$

which concludes the proof. ∎

We are now ready to define the quantum graph for Theorem 4.1. To this end, let (H,I) be a graph with inequalities and let $L = L(H,I)$. We set

$$Q[H,I] := \sum_{F \in \mathsf{Spasm}(H,I)} \lambda_F \cdot F\,,$$

where

$$\lambda_F := \sum_{\substack{\rho \in L \\ F \simeq H/\rho}} \mu_L(\emptyset, \rho)\,. \tag{4.2}$$

Corollary 4.6. $\#\mathsf{Hom}(Q[H,I] \to \star) = \#\mathsf{PartInj}(H, I \to \star)$.

Proof. Invoking Möbius inversion (Theorem 2.25) on the zeta transformation given by Lemma 4.5 yields

$$\#\mathsf{PartInj}(H, I \to \star) = \sum_{\rho \geq \emptyset} \mu_L(\emptyset, \rho) \cdot \#\mathsf{Hom}(H/\rho \to \star)\,.$$

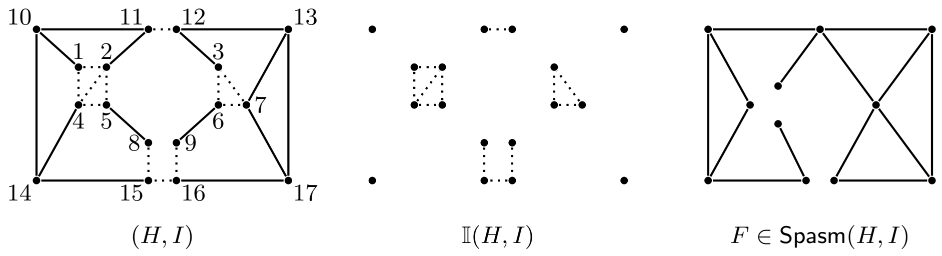


Figure 4.2: Illustration of Lemma 4.7. Despite the fact that there is more than one flat ρ in $M(H, I)$ for which $H/\rho \simeq F$, the lemma guarantees that $\lambda_F \neq 0$. Examples are given by $\rho_1 := \{\{11, 12\}, \{1, 4\}, \{3, 6\}, \{6, 7\}, \{7, 3\}, \{8, 15\}, \{9, 16\}\}$, as well as $\rho_2 := \{\{9, 16\}, \{15, 16\}, \{8, 15\}, \{1, 2\}, \{1, 4\}, \{2, 4\}, \{2, 5\}, \{4, 5\}\}$. Now both, ρ_1 and ρ_2 have rank 6, which equals $|V(H)| - |V(F)| = 17 - 11$. Hence $\mathsf{sign}(\lambda_F) = 1$. In particular, the number of vertices of $F \simeq H/\rho_1 \simeq H\rho_2$ equals the number of connected components induced by ρ_1 and ρ_2 in the inequality graph.

The claim then follows by deleting terms for which H/ρ contains a self-loop and collecting for isomorphic graphs afterwards. In particular, there exists a flat ρ for every spasm F such that $F \simeq H/\rho$: Recall that spasms of (H, I) are defined in such a way that they are isomorphic to $H/\hat{\rho}$ for some subset $\hat{\rho}$ of I that is not necessarily a flat. However, we can take the closure $\rho := \mathsf{cl}(\hat{\rho})$ of $\hat{\rho}$ (see Chapter 2.4.1), that is, we add elements of I to $\hat{\rho}$ as long as the induced connected components in $\mathbb{I}(H, I)$ do not change. We conclude by the fact that the closure $\rho = \mathsf{cl}(\hat{\rho})$ is a flat by definition and $H/\rho = H/\hat{\rho}$. ■

The following lemma is the main structural insight of this section. We show that all spasms of (H, I) are constituents of $Q[H, I]$. Note that this is far from obvious as there might be many flats ρ for which H/ρ is isomorphic to the same spasm F. Hence λ_F is the sum of different values of the Möbius function that might, a priori, differ in sign and cancel out to zero. In fact, we rely on a corollary (Theorem 2.26) of a deep result of Rota, known as the NBC Theorem [118], to prove that such cancellations are impossible.

Lemma 4.7. *Let F be a spasm of (H, I) and let λ_F be as in (4.2). Then*

$$\mathsf{sign}(\lambda_F) = (-1)^{|V(H)| - |V(F)|}.$$

We provide an illustration of the above lemma in Figure 4.2. Recall from Chapter 2.4.1 that the rank of a subset of edges X of a graphic matroid $M(G)$ is precisely $|V(G)| - \mathsf{comp}(G[X])$, where comp denotes the number of connected components.

Proof. Let $L = L(H, I)$ and let $\rho \in L$ be a flat such that F is isomorphic to H/ρ. It suffices to prove that

$$\mathsf{sign}(\mu_L(\emptyset, \rho)) = (-1)^{|V(H)| - |V(F)|}.$$

First, we observe that $|V(H/\rho)| = |V(F)|$ as otherwise H/ρ and F would not be isomorphic. Now recall that the graph H/ρ is obtained from H by identifying every pair of vertices u and v for which $\{u,v\} \in \rho$. Consequently, the number of connected components of the induced subgraph of $\mathbb{I}(H,I)$ with edges ρ is equal to $|V(H/\rho)|$. As furthermore $\mathbb{I}(H,I)$ is by definition the underlying graph of the graphic matroid $M(H,I)$, we have that

$$\mathsf{rk}(\rho) = |V(\mathbb{I}(H,I))| - |V(H/\rho)| = |V(H)| - |V(F)|\,.$$

Now we can invoke the corollary of Rota's NBC Theorem (Theorem 2.26) and obtain that

$$\mathsf{sign}(\mu_L(\emptyset,\rho)) = (-1)^{\mathsf{rk}(\rho)} = (-1)^{|V(H)|-|V(F)|}\,. \qquad \blacksquare$$

Proof (of Theorem 4.1). Let $Q[H,I]$ be as in (4.2). The claim follows then by Corollary 4.6 and Lemma 4.7. $\blacksquare$

One might wonder, whether the tractable cases in Theorem 4.1 are not only fixed-parameter tractable, but also polynomial-time solvable. We have seen that this is the case if we count homomorphisms without inequalities (Theorem 2.34) or homomorphisms with all inequalities, that is, subgraph embeddings (Remark 3.6). However, we will show that for partial injectivity constraints, there are fixed-parameter tractable cases that are not polynomial-time solvable unless $\mathsf{P} = \mathsf{NP}$. We will encounter an example for this phenomenon in the subsequent section, in which we consider the problem of counting locally injective homomorphisms.

4.1.1 Locally Injective Homomorphisms

A homomorphism h from H to G is *locally injective* if for every $v \in V(H)$ it holds that $h|_{N(v)}$ is injective. We denote $\mathsf{Li\text{-}Hom}(H \to G)$ as the set of all locally injective homomorphisms from H to G and given a class of graphs $\mathcal{H}$, we define #Li-Hom($\mathcal{H}$) as the problem of, given graphs $H \in \mathcal{H}$ and G, computing $\#\mathsf{Li\text{-}Hom}(H \to G)$; the parameter is $|V(H)|$. Locally injective homomorphisms have been studied by Nešetřil [104] and were applied in the context of distance constrained labelings of graphs (see [62] for an overview).

It is immediate to express local injectivity as partial injectivity.

Fact 4.8. *Let H be a graph and $I := \{\{u,v\} \mid \exists w \in V(H) : u,v \in N(w)\}$. Then the following holds true for every graph G:*

$$\mathsf{Li\text{-}Hom}(H \to G) = \mathsf{PartInj}(H, I \to G)\,.$$

Consequently, Theorem 4.1 completely resolves the complexity of counting locally injective homomorphisms. We provide an example using *windmill graphs*. To this end, given $n \in \mathbb{N}$, we define W_n as the graph obtained from the matching M_n of size n by adding a new vertex a and connecting it to all vertices of the matching.

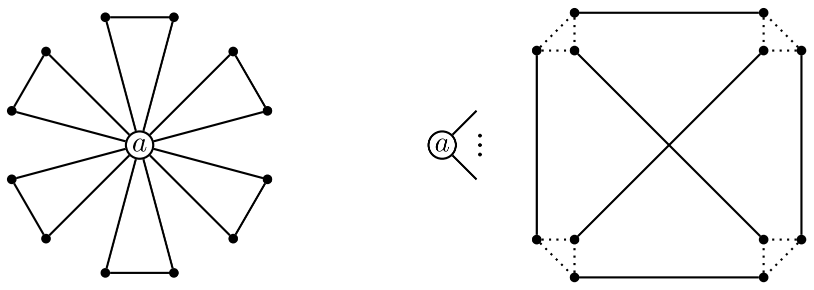


Figure 4.3: Illustration of the windmill graph W_6 and a spasm of (W_6, I) that contains the clique of size 4 as a minor: The dotted lines in the right picture constitute a flat of the inequality graph w.r.t. I as every pair of vertices of the matching has the common neighbor a. If the graph is contracted along this flat and afterwards a is removed, we obtain the desired K_4.

Corollary 4.9. *Let $\mathcal{W}$ be the set of all windmill graphs W_n for $n \in \mathbb{N}$. Then* #Li-Hom($\mathcal{W}$) *is* #W[1]*-equivalent.*

Proof. Let F be a graph with k edges and let $\hat{F}$ be the graph obtained from F by adding a new vertex a that is made adjacent to all vertices of F. We claim that $\hat{F}$ is a spasm of (W_k, I) where I is defined as in Fact 4.8. To see this, we observe that the edges of M_k in W_k can be arranged according to F and then be identified according to the inequalities between every pair of vertices of the matching; consult Figure 4.3 for an illustration. Therefore F is the minor of some spasm of (W_k, I) and consequently, the set of spasms of (W_k, I) for all $k \in \mathbb{N}$ contains all graphs as minors and is therefore of unbounded treewidth. The corollary follows hence by Theorem 4.1. ■

On the other hand, we obtain the following tractability result.

Corollary 4.10. *Let $\mathcal{T}$ be the set of all trees. Then* #Li-Hom($\mathcal{T}$) *is fixed-parameter tractable.*

Proof. Let T be a tree and let I as in Fact 4.8. Then every spasm of (T, I) is a tree as well and has hence treewidth 1. Fixed-parameter tractability follows by Theorem 4.1. ■

However, #Li-Hom($\mathcal{T}$) is an example for an instance of counting locally injective homomorphisms that is most likely not solvable in polynomial time.

Lemma 4.11. #Li-Hom($\mathcal{T}$) *is* #P*-hard.*

The class #P is the classical counting analogue of NP. A formal definition, including a rough introduction to classical counting, as well as the proof of the above lemma can be found in Appendix A. Roughly speaking, the idea is to reduce from subgraph isomorphism on trees, which is shown to be hard in Appendix A as well.

4.1.2 Quantum Graphs with Inequalities

We will now go one step further and consider linear combinations of partially injective homomorphisms. In particular, this allows for generalizing subgraph counting in the sense, that we do not only wish to count subgraphs isomorphic to a single graph H, but rather to count subgraphs that are isomorphic to some graph in a given set of graphs. An example for the latter is the problem of counting acyclic subgraphs of size k. We emphasize that the subsequent results for subgraphs have already been observed by Curticapean, Dell and Marx [41]. We extend their results to partial injectivity constraints.

It will be convenient to express the problems in this section by *quantum graphs with inequalities*, which are defined to be formal linear combinations of graphs with inequalities of finite support. We write

$$\mathcal{I} = \sum_{(H,I)} \lambda_{(H,I)} \cdot (H,I)\,,$$

where the sum is over all (isomorphism types of) graphs and inequality constraints. In particular, $\mathcal{I}$ might contain the constituents (H,I) and (H,I') for $I \neq I'$. Now counting partially injective homomorphisms extends to $\mathcal{I}$ linearly.

$$\#\mathsf{PartInj}(\mathcal{I} \to \star) := \sum_{(H,I)} \lambda_{(H,I)} \cdot \#\mathsf{PartInj}(H, I \to \star)\,.$$

If all constituents (H,I) of $\mathcal{I}$ satisfy that I is the full injectivity constraint, then $\#\mathsf{PartInj}(\mathcal{I} \to \star)$ computes a linear combination of subgraph embeddings, given by

$$\#\mathsf{PartInj}(\mathcal{I} \to \star) = \sum_{H} \lambda_H \cdot \#\mathsf{Emb}(H \to \star)\,.$$

where the sum is over all graphs and $\lambda_H = \lambda_{(H,I)}$ if (H,I) is a constituent of $\mathcal{I}$ and $\lambda_H = 0$ otherwise.

We point out that $\#\mathsf{PartInj}(\mathcal{I} \to \star)$ does not allow for complexity monotonicity in general, even if there are no negative coefficients. To see this, recall that counting homomorphisms is the zeta transformation of counting subgraphs (see Equation 2.2). In particular, we have that

$$\#\mathsf{Hom}(M_k \to \star) = \sum_{\rho} \#\mathsf{Emb}(M_k/\rho \to \star)\,,$$

where the sum is over the partition lattice of $V(M_k)$. Consequently, assuming that $k = \binom{r}{2}$, this constitutes a linear combination of subgraph embeddings, such that the r-clique is contained with a non-zero coefficient.

However, as matchings have treewidth 1, $\#\mathsf{Hom}(M_k \to \star)$ can be computed in time $\mathsf{poly}(k) \cdot n^{O(1)}$ by Theorem 2.35, whereas cliques of size r cannot be counted in time

$$\mathsf{poly}(r) \cdot n^{O(1)} = \mathsf{poly}(k) \cdot n^{O(1)}\,,$$

unless ETH fails (Theorem 2.18).

Now observe that quantum graphs with inequalities yield linear combinations of partially injective homomorphisms and the latter are again linear combinations of homomorphisms. Consequently, it is possible to express the number of partially injective homomorphisms from a quantum graph with inequalities as the number of homomorphisms from a quantum graph without inequalities; recall the definition of $Q[H,I]$ in (4.2):

$$\begin{aligned}
&\#\mathsf{PartInj}(\mathcal{I} \to \star) && (4.3)\\
&= \sum_{(H,I)} \lambda_{(H,I)} \cdot \#\mathsf{PartInj}(H, I \to \star) && (4.4)\\
&= \sum_{(H,I)} \lambda_{(H,I)} \cdot \#\mathsf{Hom}(Q[H,I] \to \star) && (4.5)\\
&= \sum_{(H,I)} \lambda_{(H,I)} \cdot \sum_{F \in \mathsf{Spasm}(H,I)} \left(\sum_{\substack{\rho \in L(H,I) \\ F \simeq H/\rho}} \mu_{L(H,I)}(\emptyset, \rho) \right) \#\mathsf{Hom}(F \to \star)\,, && (4.6)
\end{aligned}$$

where (4.4) is the definition of $\#\mathsf{PartInj}(\mathcal{I} \to \star)$, (4.5) is Corollary 4.6 and (4.6) is the definition of $Q[H,I]$. This induces the following quantum graph $Q[\mathcal{I}]$ (without inequalities) given by

$$Q[\mathcal{I}] := \sum_F \nu_F \cdot F\,,$$

where

$$\nu_F := \sum_{\substack{(H,I) \\ F \in \mathsf{Spasm}(H,I)}} \lambda_{(H,I)} \left(\sum_{\substack{\rho \in L(H,I) \\ F \simeq H/\rho}} \mu_{L(H,I)}(\emptyset, \rho) \right)\,.$$

We will now provide a criterion that, if satisfied, allows us to give the support of $Q[\mathcal{I}]$ explicitly.

Theorem 4.12. *Let $\mathcal{I}$ be a quantum graph with inequalities. Then*

$$\#\mathsf{PartInj}(\mathcal{I} \to \star) = \#\mathsf{Hom}(Q[\mathcal{I}] \to \star)\,.$$

If, additionally, no coefficient of $\mathcal{I}$ is negative and every pair $(H,I), (H',I')$ of constituents of $\mathcal{I}$ satisfies $|V(H)| = |V(H')|$, then

$$\mathsf{supp}(Q[\mathcal{I}]) = \bigcup_{(H,I) \in \mathsf{supp}(\mathcal{I})} \mathsf{Spasm}(H,I)\,.$$

Proof. $\#\mathsf{PartInj}(\mathcal{I} \to \star) = \#\mathsf{Hom}(Q[\mathcal{I}] \to \star)$ holds by (4.6) and collecting for isomorphic terms. For the second claim, we first observe that

$$\mathsf{supp}(Q[\mathcal{I}]) \subseteq \bigcup_{(H,I)\in\mathsf{supp}(\mathcal{I})} \mathsf{Spasm}(H,I)$$

by the definition of the coefficients ν_F. Now let $F \in \mathsf{Spasm}(H,I)$ for some constituent (H,I) of $\mathcal{I}$ and let $k = |V(H)|$. Then we have that

$$\mathsf{sign}\left(\sum_{\substack{\rho\in L(H,I)\\ F\simeq H/\rho}} \mu_{L(H,I)}(\emptyset,\rho)\right) = (-1)^{k-|V(F)|}$$

by Lemma 4.7. Consequently

$$\mathsf{sign}\left(\lambda_{(H,I)} \sum_{\substack{\rho\in L(H,I)\\ F\simeq H/\rho}} \mu_{L(H,I)}(\emptyset,\rho)\right) = (-1)^{k-|V(F)|}$$

as $\lambda_{(H,I)} \geq 0$ by assumption and $\lambda_{(H,I)} \neq 0$ since (H,I) is a constituent of $\mathcal{I}$. It follows that $\mathsf{sign}(\nu_F) = (-1)^{k-|V(F)|}$ as well and hence $\nu_F \neq 0$. ■

Note that it was crucial in the above proof that all graphs H for which (H,I) is a constituent of $\mathcal{I}$ have the same number of vertices. If this is not the case, then we cannot guarantee that the terms

$$\mathsf{sign}\left(\lambda_{(H,I)} \sum_{\substack{\rho\in L(H,I)\\ F\simeq H/\rho}} \mu_{L(H,I)}(\emptyset,\rho)\right) = (-1)^{|V(H)-|V(F)|}$$

are equal for different constituents and hence there might be cancellations. Going back to the example of computing $\#\mathsf{Hom}(M_k \to \star)$, we observe that the expression as a linear combination of subgraph embeddings does indeed contain constituents with different numbers of vertices.

We furthermore point out that the proof of Theorem 4.12 actually shows the following, more general statement.

Remark 4.13. The second claim of Theorem 4.12 holds true even if we only require that all non-zero coefficients have the same sign and that the number of vertices of constituents of $\mathcal{I}$ have the same parity.

Now, arguably, Theorem 4.12 might seem artificial at first glance. Let us hence provide a concrete application which shows its utility. We define

#VERTEXFORESTS as the problem of, given a graph G and a positive integer k, computing the number of acyclic subgraphs with k vertices[1] in G. The problem is parameterized by k. The following hardness result follows implicitly from [41]. We include a proof only for illustrating an application of Theorem 4.12.

Lemma 4.14. #VERTEXFORESTS *is* #W[1]*-equivalent.*

Proof. Let $\mathcal{F}_k$ be the set of all acyclic graphs with k vertices. Furthermore, given some graph F, we write $\mathsf{full}(F)$ for the set of all possible inequalities between vertices of H. Now the number of acyclic subgraphs of size k of a graph G equals

$$\begin{aligned}\sum_{F\in\mathcal{F}_k} \#\mathsf{Sub}(F\to G) &= \sum_{F\in\mathcal{F}_k} \#\mathsf{Aut}(F)^{-1}\cdot\#\mathsf{Emb}(F\to G)\\ &= \sum_{F\in\mathcal{F}_k} \#\mathsf{Aut}(F)^{-1}\cdot\#\mathsf{PartInj}(F,\mathsf{full}(F)\to G)\\ &= \#\mathsf{PartInj}(\mathcal{I}\to G)\,,\end{aligned}$$

where

$$\mathcal{I} = \sum_{\substack{(F,\mathsf{full}(F))\\ F\in\mathcal{F}_k}} \#\mathsf{Aut}(F)^{-1}\cdot(F,\mathsf{full}(F))$$

is a quantum graph with inequalities. As, by definition, all graphs in $\mathcal{F}_k$ have k vertices and the terms $\#\mathsf{Aut}(F)^{-1}$ are all greater than zero, we can invoke Theorem 4.12 and obtain the quantum graph $Q[\mathcal{I}]$ such that $\sum_{F\in\mathcal{F}_k}\#\mathsf{Sub}(F\to G) = \#\mathsf{Hom}(Q[\mathcal{I}]\to G)$ and

$$\mathsf{supp}(Q[\mathcal{I}]) = \bigcup_{F\in\mathcal{F}_k}\mathsf{Spasm}(F,\mathsf{full}(F))\,.$$

Now similarly to the proof of Corollary 4.2, we invoke complexity monotonicity and obtain that #VERTEXFORESTS is interreducible with #HOM($\mathcal{Q}$) with respect to parameterized Turing reductions, where

$$\mathcal{Q} = \left\{ H\in\mathsf{Spasm}(F,\mathsf{full}(F)) \;\middle|\; F\in\bigcup_{k\in\mathbb{N}}\mathcal{F}_k \right\}.$$

As for every $k\in\mathbb{N}$, $\mathcal{F}_{2k}$ contains the matching M_k, we conclude that $\mathcal{Q}$ contains for every k all (connected) graphs with k edges and is therefore of unbounded treewidth. The lemma hence holds by Theorem 2.34. ■

Remark 4.15. The problem #TREES of counting connected acyclic subgraphs with k vertices can be proved #W[1]-equivalent similarly [17, 41].

[1]We emphasize in the name of the problem, that we are interested in subgraphs with k vertices, not with k edges. The latter is also known to be hard, but the proof is more involved and does not use quantum graphs [17].

4.2 Edge-Injective Homomorphisms

Having completely classified the complexity of counting homomorphisms with binary inequality constraints, we will now turn to a further notion that constitutes an intermediate step between homomorphisms and subgraph embeddings: A homomorphism h from a graph H to a graph G is called *edge-injective* if, for any distinct (but not necessarily disjoint) edges $\{u, v\}$ and $\{\hat{u}, \hat{v}\}$ of H, the edges $\{h(u), h(v)\}$ and $\{h(\hat{u}), h(\hat{v})\}$ of G are distinct (but not necessarily disjoint). We write $\mathsf{EdgeInj}(H \to G)$ for the set of all edge-injective homomorphisms from H to G. Similarly to prior notions of homomorphism counting problems, we will study the complexity of counting edge-injective homomorphisms from general patterns. To this end, let $\mathcal{H}$ be a class of graphs and define $\#\textsc{EdgeInj}(\mathcal{H})$ as the problem of, given a graph $H \in \mathcal{H}$ and a graph G, computing $\#\mathsf{EdgeInj}(H \to G)$; the parameter is $|V(H)|$. For example, we can set $\mathcal{C}$ to be the class of all cycles. Then $\#\textsc{EdgeInj}(\mathcal{C})$ is equivalent to the problem of, given a positive integer k and a graph G, computing the number of edge-disjoint cycles of length k in G.

It will turn out that edge-injective homomorphisms show the limits of the framework of quantum graphs when it comes to explicit criteria for (fixed-parameter) tractability: While it is not hard to prove the existence of a quantum graph $Q_{\mathsf{EI}}[H]$ such that

$$\#\mathsf{Hom}(Q_{\mathsf{EI}}[H] \to \star) = \#\mathsf{EdgeInj}(H \to \star)\,,$$

we cannot hope to obtain an explicit representation of the support of $Q_{\mathsf{EI}}[H]$ for general graphs H, at least not without a new major insight. The reason for this is, roughly speaking, that the coefficients of the constituents of Q_{EI} cannot be given nicely as a sum of Möbius functions over some lattice for which Rota's NBC Theorem is applicable (see Theorem 2.26). Recall that in case of partially injective homomorphisms, we were able to use the matroid induced by the inequalities for the application of the NBC Theorem, which allowed us to prove that no cancellations occur when collecting for the coefficients of terms in the quantum graph. However, edge-injectivity cannot be expressed by *binary* inequality constraints, which seems to be the reason why a similar attempt fails.

On the positive side, the existence of $Q_{\mathsf{EI}}[H]$ yields at least the following implicit criterion.

Theorem 4.16. *Let $\mathcal{H}$ be a recursively enumerable class of graphs. Then $\#\textsc{EdgeInj}(\mathcal{H})$ is either fixed-parameter tractable or $\#\mathsf{W}[1]$-equivalent.*

Furthermore, we will be able to provide an explicit criterion for the case of hereditary classes of graphs. Here we say that $\mathcal{H}$ is *hereditary* if it is closed under taking induced subgraphs, that is, whenever a graph H is contained in $\mathcal{H}$ and $\hat{H}$ is an induced subgraph of H, then $\hat{H} \in \mathcal{H}$ as well. For the statement of the explicit criterion, we recall that a vertex cover of a graph H is a set of vertices $S \subseteq V(H)$ such that every edge of H is incident to at least one vertex in S. Now, given a graph H, we define the *weak vertex cover number* of H to be the minimum size of a vertex cover of the graph obtained from H by deleting isolated edges. Moreover, we say that a class of graphs $\mathcal{H}$ has *bounded weak vertex cover number* if there exists a constant d such that the weak vertex cover number of every graph $H \in \mathcal{H}$ is at most d.

Theorem 4.17. *Let $\mathcal{H}$ be a recursively enumerable class of graphs. If the weak vertex cover number of $\mathcal{H}$ is bounded, then $\#\textsc{EdgeInj}(\mathcal{H})$ is fixed-parameter tractable. Otherwise, if additionally the class $\mathcal{H}$ is hereditary, then $\#\textsc{EdgeInj}(\mathcal{H})$ is $\#\mathsf{W}[1]$-equivalent.*

As Theorem 4.17 leaves open an explicit criterion for non-hereditary classes $\mathcal{H}$ we will prove a separate hardness result for the particular non-hereditary classes of paths and cycles.

Theorem 4.18. *$\#\textsc{EdgeInj}(\mathcal{H})$ is $\#\mathsf{W}[1]$-equivalent if $\mathcal{H}$ is the class of all cycles or the class of all paths.*

The proofs of Theorem 4.16 are self-contained and can be found in Chapter 4.2.1 and Chapter 4.2.3, respectively. For the proof of Theorem 4.17, which constitutes the majority of this chapter, we will rely on a $\#\mathsf{W}[1]$-hardness result of counting k-matchings in bipartite graphs of high girth due to Curticapean [39, 42] as well as on a Ramsey argument for graphs of unbounded vertex cover number due to Curticapean and Marx [43]. The proof can be found in Chapter 4.2.2.

4.2.1 An Implicit Exhaustive Classification

We start by expressing edge-injective homomorphisms as a linear combination of subgraph embeddings. To this end, given a graph H, we call a partition ρ of $V(H)$ *edge-injective* if for every set $S \in \rho$, there is no edge between two vertices in S and, for every pair of sets $S, S' \in \rho$, there is at most one edge between S and S' in H. We write $\mathsf{EI\text{-}Part}(H)$ for the set of all edge-injective partitions of $V(H)$.

Lemma 4.19. *Let H and G be graphs. We have that*

$$\#\mathsf{EdgeInj}(H \to \star) = \sum_{\rho \in \mathsf{EI\text{-}Part}(H)} \#\mathsf{Emb}(H/\rho \to \star)\,.$$

Proof. There exists a bijection between edge-injective homomorphisms h from H to G and pairs (ρ, g) where $\rho \in \mathsf{EI\text{-}Part}(H)$ and g is an injective homomorphism from H/ρ to G. To define (ρ, g) from h, put vertices of H

into the same set S of ρ if and only if they map to the same vertex $v \in V(G)$ under h; then g maps that set S, which is a vertex in H/ρ, to v. Conversely, if (ρ, g) is a given pair, the canonical homomorphism f that maps H to H/ρ is edge-injective, and we set $h = f \circ g$. It is easy to check that this is indeed the required bijection. ■

Now let $Q_{\mathsf{EI}}[H] := \sum_F \lambda_F \cdot F$ be the quantum graph defined by

$$\lambda_F := \sum_{\rho \in \mathsf{EI\text{-}Part}(H)} \sum_{\substack{\sigma \in L(H/\rho) \\ F \simeq H/\rho/\sigma}} \mu_{L(H/\rho)}(\bot, \sigma)\,, \tag{4.7}$$

where $H/\rho/\sigma := (H/\rho)/\sigma$, $L(H/\rho)$ is the partition lattice of $V(H/\rho)$ and $\bot$ is the discrete partition.

Lemma 4.20. $\#\mathsf{EdgeInj}(H \to \star) = \#\mathsf{Hom}(Q_{\mathsf{EI}}[H] \to \star)$.

Proof. We have that

$$\begin{aligned}
&\#\mathsf{EdgeInj}(H \to \star) \\
&= \sum_{\rho \in \mathsf{EI\text{-}Part}(H)} \#\mathsf{Emb}(H/\rho \to \star) \\
&= \sum_{\rho \in \mathsf{EI\text{-}Part}(H)} \sum_{\sigma \in L(H/\rho)} \mu_{L(H/\rho)}(\bot, \sigma) \cdot \#\mathsf{Hom}(H/\rho/\sigma \to \star)\,,
\end{aligned}$$

where the first equality is Lemma 4.19 and the second equality holds by Möbius inversion over $L(H/\rho)$ (see Equation (2.4)). Consequently, the lemma follows by collecting for isomorphic $H/\rho/\sigma$ and deleting those with self-loops — recall again that $\#\mathsf{Hom}(F \to G)$ is zero whenever F contains a self-loop, as input graphs G are not permitted to have self-loops. ■

Proof (of Theorem 4.16). By Lemma 4.20 and the complexity monotonicity property (Theorem 3.13), the problems #EdgeInj($\mathcal{H}$) and #Hom($\mathcal{Q}[\mathcal{H}]$) are interreducible w.r.t. parameterized Turing reductions, where

$$\mathcal{Q}[\mathcal{H}] = \bigcup_{H \in \mathcal{H}} \mathsf{supp}(Q_{\mathsf{EI}}[H])\,.$$

In particular, $\mathcal{Q}[\mathcal{H}]$ is recursively enumerable as $\mathcal{H}$ is. By Theorem 2.34 #Hom($\mathcal{Q}[\mathcal{H}]$) is either polynomial-time solvable (and hence fixed-parameter tractable) or #W[1]-equivalent. ■

We remark that we do not obtain a polynomial-time algorithm for all fixed-parameter tractable cases, as the reduction $H \mapsto Q_{\mathsf{EI}}[H]$ might, a priori, take time $f(|V(H)|)$ for some function f which is not a polynomial. However, it can be shown that polynomial-time tractability is possible if $\mathcal{H}$ has bounded weak vertex cover number.

Theorem 4.21 ([42]). *Let $\mathcal{H}$ be a class of graphs with bounded weak vertex cover number. Then #EdgeInj($\mathcal{H}$) is solvable in polynomial time.*

4.2.2 Hereditary Graph Classes

While the proof of Theorem 4.21 relies on an involved dynamic programming algorithm that uses quantum graphs only implicitly, we will provide a short proof of the following weaker version for the sake of completeness.

Lemma 4.22. *Let $\mathcal{H}$ be a class of graphs with bounded weak vertex cover number. Then $\#\textsc{EdgeInj}(\mathcal{H})$ is fixed-parameter tractable.*

Proof. Let d be the bound on the weak vertex cover number of $\mathcal{H}$ and let $H \in \mathcal{H}$ and G be the input of $\#\textsc{EdgeInj}(\mathcal{H})$. We start with the following preprocessing.

Claim 4.23.

- *If v is an isolated vertex in H and $H-v$ is obtained by deleting v from H, then*
$$\#\mathsf{EdgeInj}(H \to G) = |V(G)| \cdot \#\mathsf{EdgeInj}(H - v \to G)\,.$$
- *If $e = \{u, v\}$ is an isolated edge in H, then*
$$\#\mathsf{EdgeInj}(H \to G) = 2\,(|E(G)| - |E(H)| + 1) \cdot \#\mathsf{EdgeInj}(H - u - v \to G)\,.$$

Proof. The first item is trivial. To prove the second item, first note that every edge-injective homomorphism h from $H-e$ to G has exactly $|E(H)|-1$ edges of G in its image. To extend h to an edge-injective homomorphism from H to G, we have to map e to an edge that is distinct (but not necessarily disjoint) from the edges in the image of h. There are $|E(G)| - |E(H)| + 1$ candidates for the image of e, and once an image e' of e has been determined, we can independently choose one of the two orientations to map u and v to the endpoints of e'. Since every edge-injective homomorphism from H to G is obtained in this way exactly once, the claim follows. □

It follows that we can assume H to contain neither isolated vertices, nor isolated edges. In particular, the vertex cover number of H is bounded by d. By Lemma 4.20, we can obtain the number $\#\mathsf{EdgeInj}(H \to G)$ by computing $\#\mathsf{Hom}(Q_{\mathsf{EI}}[H] \to G)$. Now all constituents of $Q_{\mathsf{EI}}[H]$ are of the form $H/\rho/\sigma$ for some $\rho \in \mathsf{EI\text{-}Part}(H)$ and $\sigma \in L(H/\rho)$. Consequently, the vertex cover number of all constituents is also bounded by d, as it cannot increase by taking a quotient graph. As hence the treewidth of all constituents is bounded by d as well, we can perform treewidth-based dynamic programming (see Chapter 2.5) to compute $\#\mathsf{Hom}(H/\rho/\sigma \to G)$ in time
$$\mathsf{poly}(|V(H)|) \cdot |V(G)|^{d+O(1)}\,.$$

Finally, we compute the linear combination induced by (4.7) and obtain an overall running time of
$$f(|V(H)|) \cdot |V(G)|^{d+O(1)} = f(|V(H)|) \cdot |V(G)|^{O(1)}\,,$$
for some computable function f. ■

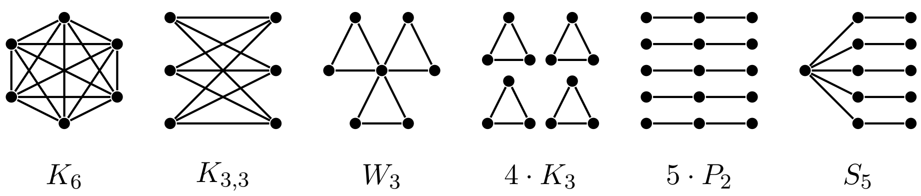


Figure 4.4: Example graphs from each of the six minimal graph classes that do not have bounded weak vertex-cover number according to Lemma 4.24.

The remainder of this section is devoted to prove that, in case $\mathcal{H}$ is hereditary and of unbounded weak vertex cover number, the problem #EdgeInj($\mathcal{H}$) is #W[1]-hard. To this end, we first show that every graph class of unbounded weak vertex cover number contains one of the six basic graph classes depicted in Figure 4.4 as induced subgraphs.

Recall that a graph is a windmill W_k of size k if it is a matching of size k with an additional *center vertex* adjacent to every other vertex. Moreover, the *subdivided star* S_k is a k-matching with a center vertex that is adjacent to exactly one vertex of each edge in the matching. A *triangle packing* $k \cdot K_3$ is the disjoint union of k triangles, a *wedge* is a path P_2 that consists of two edges, and a *wedge packing* $k \cdot P_2$ is the disjoint union of k wedges.

In what follows, we say that a class $\mathcal{H}$ contains another class $\mathcal{C}$ as induced subgraphs if, for every $C \in \mathcal{C}$, there is some $H \in \mathcal{H}$ such that H contains C as induced subgraph.

Lemma 4.24. *If $\mathcal{H}$ is a class of graphs with unbounded weak vertex-cover number, then $\mathcal{H}$ contains at least one of the following classes as induced subgraphs:*

(1) the class of all cliques,

(2) the class of all bicliques,

(3) the class of all subdivided stars,

(4) the class of all windmills,

(5) the class of all triangle packings, or

(6) the class of all wedge packings.

Proof. Let $\mathcal{H}$ be a graph class of unbounded weak vertex-cover number, and let $C \in \mathbb{N}$ be a constant such that all cliques, bicliques, subdivided stars, windmills, and triangle packings that occur as induced subgraphs in $\mathcal{H}$ have size at most C. To prove the lemma, we argue that $\mathcal{H}$ contains induced P_2-packings of unbounded size. To simplify the argument, we assume without loss of generality that $\mathcal{H}$ is closed under taking induced subgraphs.

Let $\mathcal{H}' \subseteq \mathcal{H}$ be the class of all graphs $H \in \mathcal{H}$ that do not contain isolated edges. Since $\mathcal{H}$ has unbounded weak vertex-cover number, the vertex-cover number of $\mathcal{H}'$ is unbounded. Curticapean and Marx [43, Lemma 5.2] prove that, in this situation, $\mathcal{H}'$ contains arbitrarily large cliques, induced bicliques, or induced matchings. By our assumption, the size of every clique and biclique is at most C. Thus for every k, there is a graph $H_k \in \mathcal{H}'$ such that H_k contains a k-matching $M_k \subseteq E(H_k)$ as an induced subgraph.

For every k and every $e \in M_k$, we choose an arbitrary vertex v_e in $V(H_k) \setminus V(M_k)$ that is adjacent in H_k to one or both endpoints of e. These vertices exist since e is not an isolated edge in H_k and M_k is an induced matching in H_k. Let $N_k = \{v_e \mid e \in M_k\}$ and note that v_e and $v_{e'}$ may coincide for distinct $e, e' \in M_k$. Let $A_v \subseteq M_k$ be the set of all $e \in M_k$ such that exactly one endpoint of e is adjacent to v, and let $B_v \subseteq M_k$ be the set of all $e \in M_k$ that have both their endpoints adjacent to v. If $A_v \neq \emptyset$, the graph $H_k[V(A_v) \cup \{v\}]$ is an induced subdivided star of size $|A_v|$. Similarly, if $B_v \neq \emptyset$, the graph $H_k[V(B_v) \cup \{v\}]$ is an induced windmill of size $|B_v|$. By our assumption on $\mathcal{H}$, the sets A_v and B_v have size at most C for all $v \in N_k$ and all k.

Before we argue that arbitrarily large P_2-packings exist as induced subgraphs, we apply Ramsey's theorem to obtain more structure. Since

$$M_k = \bigcup_{v \in N_k} (A_v \cup B_v),$$

the set N_k has size at least $k/(2C)$. Thus the graph class $\{H_k[N_k] \mid k \in \mathbb{N}\}$ is infinite, and since we assumed that every clique in $\mathcal{H}$ has bounded size, Ramsey's theorem guarantees the existence of independent sets $I_k \subseteq N_k$ whose sizes are unbounded as k grows.

Finally, we construct a large induced packing of triangles and paths of length 2 using the following greedy procedure: For each $v \in I_k$ with $B_v \neq \emptyset$, we select an arbitrary edge $e \in B_v$ to contribute one triangle with v, and we remove $A_v \cup B_v$ from M_k. Similarly, each $v \in I_k$ with $B_v = \emptyset$ and $A_v \neq \emptyset$ contributes one copy of P_2 and we delete $A_v \cup B_v$ from M_k. By definition of A_v and B_v, the vertex v is not adjacent to any edge in $M_k \setminus (A_v \cup B_v)$; moreover, it is not adjacent to any vertex in $I_k \setminus \{v\}$. Hence the constructed disjoint union of triangles and paths of length 2 is indeed an induced subgraph of H_k. Since all sets A_v and B_v are of size at most C, the number of components we constructed is at least $|I_k|/(2C)$, which is unbounded as k grows. By our assumption on $\mathcal{H}$, at most C of the components are triangles, and at least $|I_k|/(2C) - C$ components are copies of P_2. We conclude that $\mathcal{H}$ contains arbitrarily large induced P_2-packings. ■

As hereditary classes $\mathcal{H}$ are closed under induced subgraphs, Lemma 4.24 guarantees that any hereditary class $\mathcal{H}$ with unbounded weak vertex cover number contains at least one of the six graph families defined above as an actual subset of $\mathcal{H}$. We need to prove hardness for each of these six families.

Let us start with the following five:

Lemma 4.25. *If $\mathcal{H}$ is the class of all cliques, the class of all bicliques, the class of all subdivided stars, the class of all windmills, or the class of all triangle packings, then* $\#\textsc{EdgeInj}(\mathcal{H})$ *is* $\#\mathsf{W}[1]$*-hard.*

As we show in the following, every edge-injective homomorphism from a clique, a biclique, or a windmill into a graph is, in fact, an embedding. For these three graph families, counting edge-injective homomorphisms is thus equivalent to the corresponding subgraph counting problem. Since the families have unbounded vertex-cover number, Theorem 3.15 implies that the subgraph counting problem for these three graph families is $\#\mathsf{W}[1]$-hard.

Lemma 4.26. *Let G be a graph and let H be a clique, biclique, or windmill. Then every edge-injective homomorphism h from H to G is an embedding.*

Proof. Let h be an edge-injective homomorphism from H to G. For two distinct vertices x and y of H, we have $h(x) \neq h(y)$ if x and y are joined by an edge of H or if they have a common neighbor z in H. If H is a clique, then all $x, y \in V(H)$ with $x \neq y$ are adjacent in H. If H is a biclique or a windmill, then any two distinct vertices x and y are either adjacent or have a common neighbor. In either case, h is an embedding. ∎

Proposition 4.27. *The problem* $\#\textsc{EdgeInj}(\mathcal{H})$ *is* $\#\mathsf{W}[1]$*-hard if $\mathcal{H}$ is the class of all cliques, the class of all bicliques, or the class of all windmills.*

Proof. By Lemma 4.26, the problems $\#\textsc{EdgeInj}(\mathcal{H})$ and $\#\textsc{Emb}(\mathcal{H})$ are equivalent. Thus, since $\mathcal{H}$ has unbounded vertex-cover number, the problem is $\#\mathsf{W}[1]$-hard by Theorem 3.15. ∎

For the class of triangle packings, we devise a straightforward reduction from the problem of counting k-matchings in bipartite graphs, which was shown to be $\#\mathsf{W}[1]$-hard by Curticapean and Marx [43]. Recall that a graph G is *bipartite* if its vertices can be partitioned in two disjoint sets U and V such that every edge of G has one endpoint in U and one endpoint in V. We write $G = (U \cup V, E)$. Essentially, we will add an additional vertex that is adjacent to all other vertices, and since the original graph was bipartite, every triangle of the triangle packing must use the new vertex.

Proposition 4.28. *The problem* $\#\textsc{EdgeInj}(\mathcal{H})$ *is* $\#\mathsf{W}[1]$*-hard if $\mathcal{H}$ is the class of all triangle packings.*

Proof. We reduce from the problem of counting k-matchings in a bipartite graph. Given a bipartite graph $G = (U \cup V, E)$ and a number k, we construct a graph G' from G by adding a single apex a, that is, a new vertex a and the edges $\{a, v\}$ for all $v \in U \cup V$. Since G is bipartite, every triangle in G'

consists of a and some vertices $u \in U$ and $v \in V$. Such a triangle is denoted by $a_{v,u}$. We query the oracle for the instance (H, G'), where H is the graph $k \cdot K_3$. In either case, the edges of H partition into k triangles; let us fix this partition and an arbitrary order on the triangles.

Since G' is a graph without self-loops, the homomorphic image of a triangle is a triangle, and exactly one vertex of each of the k triangles is mapped to a. Let h be an edge-injective homomorphism from $k \cdot K_3$ to G'. Let the image of the i-th triangle be $\{a, u_i, v_i\}$. Since h is edge-injective and a is an apex, all u_i and v_i are mutually distinct. Moreover, the edges $\{u_i, v_i\}$ form a matching M_h of size k in G.

Finally, we claim that the number m_k of k-matchings of G can be derived from the number of edge-injective homomorphisms. The homomorphism h can first arbitrarily choose one vertex of each triangle to be mapped to a, which gives it 3^k choices. For the remaining matching of size k, the homomorphism h must map it to a matching in G by edge-injectivity. Thus it can choose one of the $2^k \cdot k!$ automorphisms of the k-matching. Overall, we get $6^k \cdot k!$ edge-injective homomorphisms h with $M_h = M$. Thus we have that the number of k-matchings in G equals

$$\frac{\#\mathsf{EdgeInj}(k \cdot K_3 \to G')}{6^k \cdot k!}.$$

The reduction takes polynomial time, increases the parameter from k to $|V(H)| = O(k)$, and requires only one query to the oracle. ∎

For the next reduction, we rely on the following #W[1]-hardness result.

Theorem 4.29 ([39, 42]). *The problem of counting matchings of size k in bipartite graphs is #W[1]-hard when parameterized by k, even if restricted to bipartite graphs whose right-side vertices have degree at most two and any two distinct left-side vertices have at most one common neighbor.*

Proposition 4.30. *The problem #EdgeInj($\mathcal{H}$) is #W[1]-hard if $\mathcal{H}$ is the class of all subdivided stars.*

Proof. We reduce from the problem of counting k-matchings in restricted bipartite graphs as defined above. Let (G, k) be an instance of this problem, and let $L(G)$ and $R(G)$ be the left and right vertex sets, respectively. Starting from G, we construct a new graph G' (see Figure 4.5):

1. Insert a vertex 0 that is adjacent to all vertices of $L(G)$.
2. For every vertex $v \in R(G)$ with $\deg(v) = 2$, remove v from the graph and add the set $N(v)$ as an edge to G'.
3. Add two special vertices 1 and 2, as well as the edges $\{0, 1\}$ and $\{1, 2\}$.

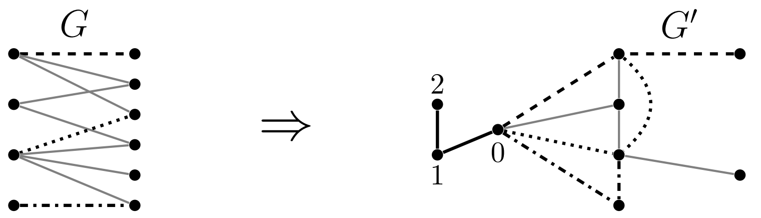


Figure 4.5: The construction of G' in the proof of Proposition 4.30, including the image of a homomorphism from the subdivided star S_4 such that vertex 2 is contained in the image: One ray of the subdivided star is mapped to the edges $\{0,1\},\{1,2\}$ and functions as an anchor. The other three rays (dashed, dotted, and dash-dotted) correspond to a 3-matching in G.

Since G is has neither multi-edges nor self-loops and any two distinct vertices $u,v \in L(G)$ have at most one common neighbor in G, the graph G' is also without multi-edges and self-loops.

Let m_k be the number of k-matchings in G, let H be the subdivided star of size $k+1$, and let $s(G) = \#\mathsf{EdgeInj}(H \to G)$. We claim that

$$(k+1)! \cdot m_k = s(G') - s(G' - \{2\}).$$

Clearly $s(G') - s(G' - \{2\})$ is exactly the number of edge-injective homomorphisms h from H to G' such that 2 is in the image of h. The claim is that there is a correspondence between such homomorphisms and the k-matchings in G.
Let h be an edge-injective homomorphism with $2 = h(z)$ for some $z \in V(H)$. Then z must be a degree-1 vertex in H since 2 has exactly one neighbor in G' and H does not contain isolated vertices. Let y be the neighbor of z and let x be the center vertex of the subdivided star. Then $h(y) = 1$ and $h(x) = 0$ holds. Next, let $y_1,\dots,y_k$ be the other degree-2 vertices of H, and let $z_1,\dots,z_k$ be the corresponding degree-1 vertices. Since h is an edge-injective homomorphism, the vertices $a_i := h(y_i)$ are mutually distinct and satisfy $a_i \in L(G)$.

Note that $h(z_i) \notin \{0,1,2\}$. Now let $M_h = \{\{a_1b_1\},\dots,\{a_kb_k\}\}$ be the matching defined as follows. If $h(z_i) \in R(G)$, then it is a degree-1 vertex of $R(G)$, and we set $b_i = h(z_i)$. Otherwise, $h(z_i) \in L(G)$ and the edge $h(\{y_i,z_i\})$ exists in G'; we let b_i be the unique vertex satisfying that $N_G(b_i) = \{a_i, h(z_i)\}$ and that caused this edge to be added to G' in the construction. The b_i are mutually distinct due to the edge-injectivity. Hence M_h is indeed a matching.

For every k-matching M of G, there are $(k+1)!$ distinct edge-injective homomorphisms h with $M = M_h$ since h can choose an arbitrary order for the $k+1$ rays of the subdivided star. This proves the claim. Overall, the reduction runs in polynomial time and queries the oracle exactly two times with parameter $|V(H)| = O(k)$. ■

Proof (of Lemma 4.25). Follows by Propositions 4.27, 4.28 and 4.30. ∎

The reader might wonder, why we did not consider the class of wedge packings so far. The reason for this is, that our #W[1]-hardness proof of counting edge-injective homomorphisms from wedge packings requires significantly more work than the previous five cases combined. Let us start with the formal statement.

Lemma 4.31. *The problem* #EDGEINJ($\mathcal{H}$) *is* #W[1]*-hard if* $\mathcal{H}$ *is the set of all wedge packings.*

Let us emphasize the following consequence of this lemma before going into the details of the proof. Recall that the *line graph* $\mathbb{L}(G)$ of a graph G has as vertices the edges of G and two different vertices are made adjacent if the corresponding edges share a common vertex in G.

Corollary 4.32. *The problem of counting matchings of size k in line graphs is* #W[1]*-hard when parameterized by k.*

Proof. The images of edge-injective homomorphisms from the wedge packing $k \cdot P_2$ to a graph G are precisely the k-matchings in $L(G)$. Consequently

$$\#\mathsf{Sub}(M_k \to \mathbb{L}(G)) = \#\mathsf{Aut}(k \cdot P_2)^{-1} \cdot \#\mathsf{EdgeInj}(k \cdot P_2 \to G)\,.$$

As $\mathbb{L}(G)$ can be computed from G in polynomial time, the above equation yields a parameterized Turing reduction from counting edge-injective homomorphisms from wedge packings of size k to counting k-matchings in line graphs. The claim follows as the former is #W[1]-hard by Lemma 4.31. ∎

Lemma 4.31 relies on the following delicate interpolation argument. We wish to point out that the proof is due to Johannes Schmitt and we are very grateful to Johannes for allowing us to include it for reasons of self-containment. In what follows, given $t \in \mathbb{N}$, we let $(x)_t$ denote the falling factorial, where

$$(x)_t = (x) \cdot (x-1) \cdots (x-t+1)\,.$$

Lemma 4.33. *For all $g, b \in \mathbb{N}$, let $a_{g,b} \in \mathbb{Q}$ be unknowns, and for all $r \in \mathbb{N}$, let $P_r(y)$ be the univariate polynomial such that*

$$P_r(y) = \sum_{k=0}^{r} \sum_{t=0}^{k} a_{t,k-t} \cdot \binom{r}{k} \cdot (y-t)_{2(r-k)}\,.$$

There is a polynomial-time algorithm that, given a number k and the coefficients of $P_r(y)$ for all $r \in \mathbb{N}$ with $r \le O(k)$,[2] computes the numbers $a_{t,k-t}$ for all $t \in \{0, \dots, k\}$.

[2]That is, r is bounded by dk for some overall constant d.

Proof. Let $t \in \{0, \dots, k\}$. For all $k, i \in \mathbb{N}$, let $I_{k,i}$ be defined as

$$I_{k,i} = \sum_{t=0}^{k} a_{t,k-t} \cdot t^i \,.$$

As an intermediate step, we construct a polynomial-time algorithm that allows us, given the coefficients of $P_r(y)$ and a number $m \in \mathbb{N}$, to compute $I_{k,i}$ for all $k, i \in \mathbb{N}$ with $2k + i \leq m$.

If $m = 0$, then $I_{0,0}$ is the only number we need to compute. We obtain it by observing that $P_0(y) = I_{0,0} = a_{0,0}$. Now suppose that $m > 0$ and that we inductively already computed the values $I_{k,i}$ for all $k, i \in \mathbb{N}$ with $2k+i \leq m$. We will compute the values $I_{k,i}$ with $2k + i = m + 1$.

Let r be an integer that satisfies $2r - (m + 1) \geq 0$. Furthermore, let C^r denote the coefficient of $y^{2r-(m+1)}$ in $P_r(y)$, which is given as input. We want to describe C^r as an expression in terms of the unknowns $a_{g,b}$. To this end, we investigate which of the summands

$$a_{t,k-t} \cdot \binom{r}{k} \cdot (y - t)_{2(r-k)}$$

contribute to C^r.

Claim 4.34. *If $k > \lfloor \frac{m+1}{2} \rfloor$ then $2(r - k) < 2r - (m + 1)$.*

Proof. We have $2(r - k) = 2r - 2k < 2r - 2\lfloor \frac{m+1}{2} \rfloor$. If $m + 1$ is even, we get $2(r - k) < 2r - (m + 1)$ as claimed. Otherwise $m + 1$ is odd, and we only get $2(r - k) < 2r - m$. However, since m and $2(r - k)$ are both even, we actually get $2(r - k) < 2r - (m + 1)$ as claimed. □

It follows that the summands with $k > \lfloor \frac{m+1}{2} \rfloor$ do not contribute to C^r.

Let us view $(y-t)_{2(r-k)}$ as a bivariate polynomial in y and t for a moment. Then, by expanding this polynomial in powers of y, there exist univariate polynomials $\sigma_i(t)$ for all $i \in \mathbb{N}$ with $i \leq 2(r - k)$ such that

$$(y - t)_{2(r-k)} = \sum_{i=0}^{2(r-k)} \sigma_i(t) \cdot y^{2(r-k)-i} \,.$$

Using bivariate interpolation (Theorem 2.28), we can easily compute all coefficients of $\sigma_i(t)$ for all $i \leq 2(r - k)$. Note that the coefficient of t^{m+1-2k} in σ_{m+1-2k} is

$$(-1)^{m+1} \cdot \binom{2(r - k)}{m + 1 - 2k} \,.$$

Let $c_0, \ldots, c_{m+1-2k-1}$ be the remaining coefficients. Since only terms that satisfy $k \leq \lfloor (m+1)/2 \rfloor$ contribute to C^r, we obtain that C^r equals

$$\begin{aligned}
&\sum_{k=0}^{\lfloor \frac{m+1}{2} \rfloor} \sum_{t=0}^{k} a_{t,k-t} \binom{r}{k} \sigma_{m+1-2k}(t) \\
&= \sum_{k=0}^{\lfloor \frac{m+1}{2} \rfloor} \sum_{t=0}^{k} a_{t,k-t} \binom{r}{k} \left[(-1)^{m+1} \binom{2(r-k)}{m+1-2k} t^{m+1-2k} + \sum_{j=0}^{m+1-2k-1} c_j t^j \right] \\
&= \sum_{k=0}^{\lfloor \frac{m+1}{2} \rfloor} \left[(-1)^{m+1} \sum_{t=0}^{k} a_{t,k-t} \binom{r}{k} \binom{2(r-k)}{m+1-2k} t^{m+1-2k} + \sum_{t=0}^{k} a_{t,k-t} \binom{r}{k} \sum_{j=0}^{m-2k} c_j t^j \right] \\
&= (-1)^{m+1} \left[\sum_{k=0}^{\lfloor \frac{m+1}{2} \rfloor} \binom{2(r-k)}{m+1-2k} \binom{r}{k} \sum_{t=0}^{k} a_{t,k-t} \cdot t^{m+1-2k} \right] + \sum_{k=0}^{\lfloor \frac{m+1}{2} \rfloor} \sum_{j=0}^{m-2k} c_j \binom{r}{k} \sum_{t=0}^{k} a_{t,k-t} t^j \\
&= (-1)^{m+1} \left[\sum_{k=0}^{\lfloor \frac{m+1}{2} \rfloor} \binom{2(r-k)}{m+1-2k} \binom{r}{k} I_{k,m+1-2k} \right] + \sum_{k=0}^{\lfloor \frac{m+1}{2} \rfloor} \sum_{j=0}^{m-2k} c_j \binom{r}{k} I_{k,j} \,.
\end{aligned}$$

Now consider the $I_{k,j}$ from the last sum. Since $2k + j \leq 2k + m - 2k = m$, we have already computed these $I_{k,j}$ recursively. We also know all of the c_j, so we can compute the number C'^r for any $r \geq \frac{m+1}{2}$, where

$$C'^r = \sum_{k=0}^{\lfloor \frac{m+1}{2} \rfloor} \binom{2(r-k)}{m+1-2k} \cdot \binom{r}{k} \cdot I_{k,m+1-2k} \,.$$

Finally, consider the matrix A corresponding to the above system of equations such that

$$A_{j,i} = \binom{2(r_j - i)}{m+1-2i} \cdot \binom{r_j}{i}$$

for $i = 0, \ldots, \lfloor \frac{m+1}{2} \rfloor$ and pairwise distinct and $r_0, \ldots, r_{\lfloor \frac{m+1}{2} \rfloor}$ large enough such that the binomial coefficient does not become zero. Column i is an evaluation vector of the polynomial

$$Q_i(r) = \binom{2(r-i)}{m+1-2i} \cdot \binom{r}{i} \,.$$

Each Q_i has degree $m + 1 - 2i + i = m + 1 - i$; in particular, the degree of Q_i is different for different i. This implies that the set $\{Q_i \mid i \in \mathbb{N}\}$ is a set of linearly independent polynomials, and thus the column vectors of A are linearly independent and A is invertible. This allows us to compute the unique solution for the $I_{k,m+1-2k}$ for all $k \in \mathbb{N}$ with $k \leq m/2$.

Finally, we argue how to compute the $a_{t,k-t}$ from the $I_{k,i}$. By definition, we have the following set of linear equations:

$$\sum_{t=0}^{k} t^0 \cdot a_{t,k-t} = I_{k,0}$$
$$\sum_{t=0}^{k} t^1 \cdot a_{t,k-t} = I_{k,1}$$
$$\cdots$$
$$\sum_{t=0}^{k} t^k \cdot a_{t,k-t} = I_{k,k}$$

The corresponding matrix B where $(B)_{i,j} = j^i$ for $i, j \in [k]$ is a Vandermonde matrix and thus invertible. Therefore we can compute the unique solution for the $a_{t,k-t}$. ■

We are now able to prove #W[1]-hardness of counting edge-injective homomorphisms from wedge packings. The proof uses the same idea as the hardness proof for subdivided stars. However, in case of wedge packings we cannot use a particular wedge as an anchor to enforce the image of the edge injective homomorphisms to have a specified structure. Instead, we rely on Lemma 4.33.

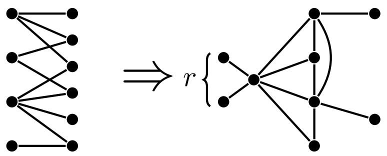
Proof (of Lemma 4.31). We reduce from the problem of counting matchings of size k in bipartite graphs whose right-side vertices have degree ≤ 2 and where any two distinct left-side vertices have at most one common neighbor. Note that this problem is #W[1]-hard by Theorem 4.29. Let (G, k) be an instance of this problem, and let $L(G)$ and $R(G)$ be the left and right vertex sets, respectively. For $r \in \mathbb{N}$, we construct a graph G^r as follows (see Figure 4.6):

1. Insert a vertex 0 that is adjacent to all vertices of $L(G)$.
2. Add r special vertices $1, \ldots, r$ as well as the edges $\{0, 1\}, \ldots, \{0, r\}$.
3. For every vertex $v \in R(G)$ with $\deg(v) = 2$, remove v and add the set $N(v)$ as an edge to G^r. Note that $|N(v)| = 2$, so $N(v)$ can indeed be considered as an edge.

Since G does neither contain multi-edges nor self-loops and any two distinct vertices $u, v \in L(G)$ have at most one common neighbor in G, the graph G^r does also neither contain multi-edges nor self-loops. Now let

$$H = H_1 \dot\cup \ldots \dot\cup H_k$$

be a wedge packing consisting of k vertex-disjoint copies of P_2.



Construction of G^r

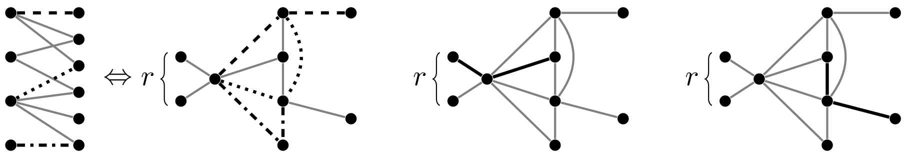


Figure 4.6: Example of the construction of G^r as used in the proof of Lemma 4.31. The second row illustrates the correspondence between a 3-matching in G and the image of an edge-injective homomorphism from a wedge packing of size 3 such that all wedges are *good*. Furthermore we give examples for the image of a *test* wedge and a *bad* wedge.

For an edge-injective homomorphism $h \in \mathsf{EdgeInj}(H \to G^0)$, we say that a wedge H_i is

- *test* if $h(H_i)$ contains two edges incident to 0,
- *good* if $h(H_i)$ contains exactly one edge incident to 0, and
- *bad* if $h(H_i)$ uses no edge incident to 0.

Let $\alpha_{g,b}$ be the number of edge-injective homomorphisms

$$h \in \mathsf{EdgeInj}(H \to G^0)$$

for which there are 0 test wedges, g good wedges, and b bad wedges.

Claim 4.35. *The number of k-matchings in G is equal to $\alpha_{k,0} \cdot \left(2^k \cdot k!\right)^{-1}$.*

Proof. The integer $\alpha_{k,0}$ is the number of all $h \in \mathsf{EdgeInj}(H \to G^0)$ such that the image of every H_i consists of a wedge that uses exactly one edge incident to 0.

Given any such h, we construct a k-matching M_h of G as follows. For each i, consider the wedge $h(H_i)$: It uses an edge $\{0, v\}$ for $v \in L(G)$ and an edge $\{v, w\}$ with $w \neq 0$. If $w \in R(G)$, then $N_G(w) = \{v\}$, and we add the edge e_i with $e_i = \{v, w\} \in E(G)$ to the matching. Otherwise, we have $w \in L(G)$, and so the edge $\{v, w\} \in E(G^0)$ corresponds to a vertex $u \in R(G)$ with $N_G(u) = \{v, w\}$, from which it was constructed. In this case, we add the edge e_i with $e_i = \{v, u\} \in E(G)$ to the matching. Note that e_i and e_j for i and j with $i \neq j$ are disjoint; for if they shared a vertex $v \in L(G)$, the edge $\{0, v\}$ would be used by both $h(H_i)$ and $h(H_j)$, and if they shared a vertex $v \in R(G)$, then either $N_G(v)$ or $N_G(v) \cup \{v\}$ would be an edge in G^0, which would be used by both $h(H_i)$ and $h(H_j)$. Thus the constructed set M_h is indeed a k-matching.

On the other hand, for each k-matching M, we have that there are exactly $2^k \cdot k!$ edge-injective homomorphisms $h \in \mathsf{EdgeInj}(H \to G^0)$ with $M = M_h$ since the automorphism group of H has this size. $\square$

We aim at determining the number $\alpha_{k,0}$ by using the provided oracle for #EdgeInj($\mathcal{H}$). Since we cannot directly ask the oracle to only count homomorphisms with a given number of bad and good wedges, we query the oracle multiple times and recover these numbers via the very specific form of interpolation fueled by Lemma 4.33. To apply the lemma, we observe the following identity.

Claim 4.36. *Let $k, r \in \mathbb{N}$. Then $\beta_k(G^r) := \#\mathsf{EdgeInj}(H \to G^r)$ satisfies*

$$\beta_k(G^r) = \sum_{\substack{t,g,b \in \mathbb{N} \\ t+g+b=k}} \alpha_{g,b} \cdot \binom{k}{g+b} \cdot (n + r - g)_{2t} \,.$$

Proof. We construct an element h of $\mathsf{EdgeInj}(H_1 \dot\cup \ldots \dot\cup H_k \to G^r)$ whose image consists of g good wedges, b bad wedges, and t test wedges, where $g+b+t = k$. There are $\binom{k}{g+b}$ possibilities to select the set of H_i that will be mapped to a good or a bad wedge. Once this selection has been done, there are $\alpha_{g,b}$ edge-injective homomorphisms that map the selected H_i to g good and b bad wedges; to see this, note that G^0 and G^r have exactly the same good and bad wedges. Finally, the test wedges can only be mapped to the edges incident to 0, for which reason only the star with center 0 is relevant for the test wedges. Each good wedge that has already been placed blocks one edge of the star. Hence the t wedges map into a star with $n + r - g$ leaves. The number of edge-injective homomorphisms that map t wedges into a star with ℓ leaves is $(\ell)_{2t}$. $\square$

Note that $\beta_k(G^r)$ is a polynomial in r of degree at most $2k$. Setting $y = n+r$, Claim 4.36 yields a polynomial identity that is exactly of the form required by Lemma 4.33, and thus we can compute the unknowns $\alpha_{g,b}$ for all $g, b \in \mathbb{N}$ with $g + b \le k$ from the polynomials $\beta_0, \ldots, \beta_{O(k)}$. Overall, the reduction runs in polynomial time, makes at most $O(k^2)$ queries to the oracle, and the parameter of each query is at most $O(k)$. This proves #W[1]-hardness. $\blacksquare$

We are now able to prove the explicit classification for counting edge-injective homomorphisms from hereditary graph classes.

Proof (of Theorem 4.17). If $\mathcal{H}$ has bounded weak vertex cover number, then #EDGEINJ($\mathcal{H}$) is fixed-parameter tractable by Lemma 4.22. Otherwise, if $\mathcal{H}$ is hereditary, then by Lemma 4.24 at least one of the six classes of cliques, bicliques, windmills, subdivided stars, triangle packings, or wedge packings is a subset of $\mathcal{H}$. For the first five, #W[1]-hardness holds by Lemma 4.25 and for the latter, #W[1]-hardness is given by Lemma 4.31. Finally, #W[1]-equivalence follows by Theorem 4.16. ■

4.2.3 Edge-Disjoint Paths and Cycles

The classification for #EDGEINJ($\mathcal{H}$) with hereditary graph classes $\mathcal{H}$ leaves open some non-hereditary graph classes of interest. In this final part of the paper, we investigate #EDGEINJ($\mathcal{H}$) for the class of cycles and that of paths and prove Theorem 4.18, that is, #W[1]-hardness for these problems.

We point out that the problems of counting edge-injective homomorphisms from C_k and P_k are equivalent to the problems of counting edge-disjoint k-cycles and edge-disjoint k-paths, respectively. In particular, for any graph G, we have that $\#\mathsf{EdgeInj}(C_k \to G)$ equals $2k$ times the number of edge-disjoint k-cycles in G, while $\#\mathsf{EdgeInj}(P_k \to G)$ equals twice the number of edge-disjoint k-paths in G.

We will first show that #EDGEINJ($\mathcal{C}$) is #W[1]-hard, where $\mathcal{C}$ is the class of all cycles. To this end, we consider the edge-weighted version of counting edge-injective homomorphisms in an intermediate step. Let H and G be graphs and let $w : E(G) \to \mathbb{N}$ a weight-function. The number of edge-weighted edge-injective homomorphisms is defined as follows

$$\#\mathsf{EdgeInj}(H \to G, w) := \sum_{h \in \mathsf{EdgeInj}(H \to G)} \prod_{e \in E(H)} w(h(e)) \,.$$

Then the problem #WEIGHTEDEDGEINJ($\mathcal{H}$) asks, given a graph $H \in \mathcal{H}$ and an arbitrary graph G with weight-function w, to compute this quantity. The parameter is

$$|V(H)| + \max\{w(e) \mid e \in E(G)\} \,,$$

that is, the edge-weights of G must be bounded by some function in the size of the pattern graph H.

Lemma 4.37. #WEIGHTEDEDGEINJ($\mathcal{C}$) *is* #W[1]*-hard.*

Proof. First we observe that, for all $k \in \mathbb{N}$, we have

$$\#\mathsf{EdgeInj}(C_k \to G, w) = 2k \cdot \sum_{c \in \mathsf{EC}_k(G)} \prod_{e \in c} w(e) \qquad (4.8)$$

where $\mathsf{EC}_k(G)$ denotes the set of all edge-disjoint cycles of length k in G.

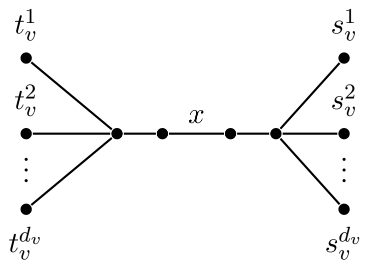


Figure 4.7: Gadget H_v for v of degree d_v as used in the proof of Lemma 4.37.

We show #W[1]-hardness by constructing a parameterized Turing reduction from #Sub($\mathcal{C}$) of counting simple cycles of length k, which is #W[1]-hard by Theorem 3.15 as $\mathcal{C}$ has unbounded vertex cover number.

On input a graph G and $k \in \mathbb{N}$, our reduction proceeds as follows: If $k < 3$, then return 0. Else consider the graph G_x obtained from G by substituting each node $v \in V$, of some degree d_v, by the gadget graph H_v constructed as follows: We start with a path of length 3 whose intermediate edge has weight x.[3] Next we add vertices $s_v^1, \ldots, s_v^{d_v}$ and connect each of them with an edge to one endpoint of the path. After that we add vertices $t_v^1, \ldots, t_v^{d_v}$ and connect each of them with an edge to the other endpoint.

Furthermore add edges $\{s_v^i, t_u^j\}$ and $\{s_u^j, t_v^i\}$ for every edge $\{u, v\}$ that is the i-th edge of v and the j-th edge of u. The resulting graph is shown in Figure 4.7. Consider G_x as a weighted graph were every edge has weight 1 except for the edges labeled with x as above. Now, querying the oracle for #WeightedEdgeInj($\mathcal{C}$) with input C_{6k} and G_x, and dividing by $12k$ yields a polynomial $p \in \mathbb{Z}[x]$.

Claim 4.38. *The degree of p is bounded by k. Furthermore the coefficient of x^k equals twice the number of simple k-cycles in G.*

Proof. The shortest edge-disjoint path between any pair of two different edges with weight x is at least 5 (excluding the two edges with weight x from the length). As we search for edge-disjoint cycles of length $6k$, the weight x can occur at most $\frac{6k}{5+1} = k$ times in one cycle. Therefore the degree of p is bounded by k. Furthermore the distance is equal to 5 if and only if the two edges belong to gadgets H_v and H_u such that $\{v, u\} \in E(G)$. In particular, this path either leaves H_v through s_v^i and enters H_u through t_u^j for some i and j or it leaves H_v through $t_v^{i'}$ and enters H_u through $s_u^{j'}$ for some i' and j'. Now consider an edge-disjoint cycle c of length $6k$ that includes k edges with weight x. It follows that

$$c = (e_1, P_1, \ldots, e_k, P_k, e_1)\,,$$

where each e_i has weight x and each P_i is a path of 5 edges with weight 1.

[3]Here, x is an indeterminate, so the quantity (4.8) is a polynomial in x.

Next let H_{v_i} be the gadget containing e_i and consider H_{v_1}. It holds that P_1 either passes through $s^i_{v_1}$ and $t^j_{v_2}$ for some i and j or through $t^{i'}_{v_1}$ and $s^{j'}_{v_2}$ for some i' and j'. However, if we fix one of these two options, only one possibility remains for all other $P_2, \dots, P_k$ as we cannot turn around in a gadget if we consider edge-disjoint cycles. Therefore there are exactly two edge-disjoint cycles $c_1 = (e_1, P_1, \dots, e_k, P_k, e_1)$ and $c_2 = (e_1, P'_1, \dots, e_k, P'_k, e_1)$ that correspond to the cycle $c = (v_1, \dots, v_k, v_1)$ in G and vice versa. Furthermore c is simple as c_1, c_2 are edge-disjoint, that is, the e_i's and therefore the v_i's are pairwise different. □

To conclude the proof of Lemma 4.37, we compute the coefficient of x^k in the degree-k polynomial p by means of polynomial interpolation from the evaluations $p(0), \dots, p(k)$ (see Theorem 2.28). These evaluations are obtained by oracle calls to #WEIGHTEDEDGEINJ($\mathcal{C}$) with input C_{6k} and G_b for $b = 0, \dots, k$ (and dividing by $12k$). As the edge-weights of every graph G_b are bounded by k, the overall parameter $|V(C_{6k})| + \max\{w(e) \mid e \in G_b\}$ is bounded by $7k$, proving that this reduction is indeed a parameterized Turing reduction. ■

We show hardness of the unweighted version by reduction from the weighted version; this requires us to devise a strategy for removing weights.

Lemma 4.39. #WEIGHTEDEDGEINJ($\mathcal{C}$) $\leq^{\mathrm{T}}_{\mathrm{fpt}}$ #EDGEINJ($\mathcal{C}$).

Proof. The input for the reduction is a number $k \in \mathbb{N}$ and an edge-weighted graph G whose edge weights are bounded by k. We assume $k \geq 4$, as we can otherwise solve the problem in polynomial time by brute-force. The following gadgets will be used in the reduction:

- G_1 is simply one undirected edge $e_1 := \{a_1, b_1\}$
- G_{i+1} is constructed from G_i as follows: We add vertices a_{i+1} and b_{i+1} and edges $\{a_{i+1}, a_i\}$ and $\{b_{i+1}, b_i\}$. Furthermore we add a path of length $2i + 1$ between a_{i+1} and b_{i+1} and denote the $i + 1$-th edge of this path e_{i+1}. G_{i+1} is depicted in Figure 4.8.

It is easy to see that $|V(G_k)| \leq O(k^2)$.

Claim 4.40. *For every $k \geq 1$, there are exactly k possibilities to construct an edge-disjoint walk from a_k to b_k in G_k, each of which has length $2k - 1$. Furthermore, for every $j = 1, \dots, k$, the edge e_j is contained in exactly one of this walks.*

Proof. We prove the claim by induction on k; it is obvious for $k = 1$. For the induction step, consider G_{k+1}: An edge-disjoint walk from a_{k+1} to b_{k+1} either takes the "left" way and therefore contains e_{k+1} or takes a way through G_k.

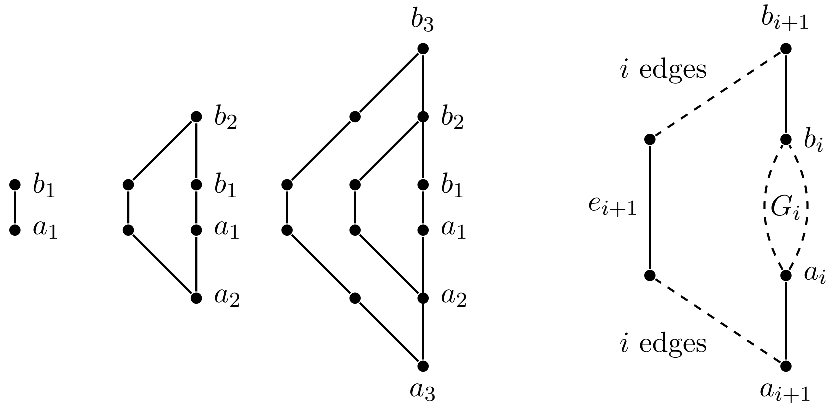


Figure 4.8: Graphs G_1, G_2 and G_3, as well as the inductive construction of the graph G_{i+1} as used in the proof of Lemma 4.39.

Now the "left" way has length exactly $2k + 1 = 2(k + 1) - 1$. Further, every way through G_k corresponds one-to-one to a closed walk from a_k to b_k in G_k. Applying the induction hypothesis we obtain that there are exactly k edge-disjoint walks from a_k to b_k in G_k, one for every e_j for $j = 1, \dots, k$. Furthermore each of this walks has length $2k - 1$. It follows that there are exactly k edge-disjoint walks from a_{k+1} to b_{k+1}, each of length

$$2 + 2k - 1 = 2(k + 1) - 1 .$$

The edge e_j is contained in exactly one of this walks for every $j = 1, \dots, k$. We conclude that the claim is fulfilled for G_{k+1}. □

It follows that the longest edge-disjoint cycle in G_k has length

$$2 \cdot (2k + 1) = 4k + 2 .$$

Let W be the maximum weight of an edge, where $W \leq k$. Now let H_i be the gadget constructed from G_W by removing edges $e_W, \cdots, e_{i+1}$. We have $H_W = G_W$. Applying Claim 4.40, we obtain that there are exactly i edge-disjoint walks from a_W to b_W in H_i. Furthermore each of this walks has length $2W - 1$. Finally, we construct G' from G by substituting each edge $e = \{a, b\}$ with $H_{w(e)}$ and edges $\{a, a_W\}$ and $\{b, b_W\}$.

Claim 4.41. *The number of edge-disjoint cycles of length $(2Wk + k)$ in G' equals*

$$\sum_{c \in \mathsf{EC}_k(G)} \prod_{e \in c} w(e) .$$

Proof. Consider an edge-disjoint cycle c of length $(2Wk + k)$ in G' and assume c does contain an edge $\{a, a_W\}$, that is, it is not entirely contained in one gadget. It then follows that c can cross every a_W and b_W at most once. To see this, observe that every time when such a node is reached we can consider the cycle coming from "outside the gadget", e.g. by choosing a fitting orientation of c. Since c is an edge-disjoint cycle, we have to continue by an edge-disjoint walk through the end of the gadget. This walk has length $2W - 1$ by Claim 4.40. Now we cannot turn around inside the gadget again and complete the cycle afterwards since otherwise we would have constructed a longer edge-disjoint walk from one endpoint of the gadget to the other, which contradicts Claim 4.40. It follows that each edge-disjoint cycle of length $(2Wk+k)$ that is not entirely contained in one gadget consists of $2W - 1$ walks through gadgets. Now, taking an edge $e = \{a, b\}$ with weight $w(e)$ in G corresponds to taking one of the $w(e)$ edge-disjoint walks $(a, a_W, \cdots, b_W, b)$ of length $2W - 1 + 2$ through $H_{w(e)}$ in G'. As $k \cdot (2W - 1 + 2) = (2Wk + k)$ it follows that an edge-disjoint cycle of length k in G corresponds to the edge-disjoint cycles of length $(2Wk+k)$ in G' that cross the gadgets corresponding to the weighted edges in G, but only if no edge-disjoint cycle of length $(2Wk+k)$ entirely fits in one gadget. However, the latter cannot be the case since the longest edge-disjoint cycle in G_W has length $4W + 2$ and for every $k > 2$ it holds that

$$4W + 2 < 4W \leq 2Wk < 2Wk + k\,. \qquad \square$$

Using Claim 4.41, Equation 4.8 and the fact that for every graph G and $k \in \mathbb{N}$, we have that $\#\mathsf{EdgeInj}(C_k \to G)$ equals $2k$ times the number of edge-disjoint k-cycles in G, we obtain that

$$\begin{aligned}\#\mathsf{EdgeInj}(C_{2Wk+k} \to G') &= 2(2Wk+k) \sum_{c \in \mathsf{EC}_k(G)} \prod_{e \in c} w(e) \\ &= \frac{2(2Wk+k)}{2k} \cdot \#\mathsf{EdgeInj}(C_k \to G, w) \\ &= (2W+1) \cdot \#\mathsf{EdgeInj}(C_k \to G, w)\,.\end{aligned}$$

The above reduction is indeed a parameterized Turing reduction, as G' can be constructed in time $O(n^2 \cdot k^2)$ and the value of the new parameter is $2Wk + k \leq O(k^2)$. This concludes the proof. ■

Corollary 4.42. *The problem* #EdgeInj($\mathcal{C}$) *is* #W[1]*-hard.*

Proof. Follows from #W[1]-hardness of #WeightedEdgeInj($\mathcal{C}$) as shown in Lemma 4.39 as well as from the reduction in the previous lemma. ■

It remains to show hardness for counting edge-injective homomorphisms from paths.

Lemma 4.43. *The problem* #EDGEINJ($\mathcal{P}$) *is* #W[1]*-hard.*

Proof. We will reduce from #EDGEINJ($\mathcal{C}$). First, we let $\mathsf{EC}_k(G, v)$ be the set of all edge-disjoint cycles of length k in G that contain $v \in V(G)$ and recall that $\mathsf{EC}_k(G)$ denotes the set of all edge-disjoint cycles of length k in G.

Claim 4.44. *It holds that*

$$\mathsf{EC}_k(G) = \bigcup_{i=1}^{|V(G)|} \mathsf{EC}_k(G - \{v_{i+1}, \dots, v_n\}, v_i),$$

where the union is in fact a pairwise disjoint union.

Proof. By induction on $|V(G)|$. If $|V(G)| = 0$, the union is empty and therefore the claim holds. Otherwise let $|V(G)| = n + 1$. It holds that

$$\begin{aligned}\mathsf{EC}_k(G) &= \mathsf{EC}_k(G, v_{n+1}) \,\dot\cup\, (\mathsf{EC}_k(G) \setminus \mathsf{EC}_k(G, v_{n+1}))\\ &= \mathsf{EC}_k(G, v_{n+1}) \,\dot\cup\, \mathsf{EC}_k(G - \{v_{n+1}\})\\ &= \mathsf{EC}_k(G, v_{n+1}) \,\dot\cup\, (\dot{\bigcup}_{i=1}^{n} \mathsf{EC}_k((G - \{v_{n+1}\}) - \{v_{i+1}, \cdots, v_n\}, v_i))\\ &= \mathsf{EC}_k(G, v_{n+1}) \,\dot\cup\, (\dot{\bigcup}_{i=1}^{n} \mathsf{EC}_k(G - \{v_{i+1}, \cdots, v_{n+1}\}, v_i))\\ &= \dot{\bigcup}_{i=1}^{n+1} \mathsf{EC}_k(G - \{v_{i+1}, \dots, v_{n+1}\}, v_i)\end{aligned}$$

Here, the third equality follows from the induction hypothesis. □

We hence have that

$$|\mathsf{EC}_k(G)| = \sum_{i=1}^{|V(G)|} |\mathsf{EC}_k(G - \{v_{i+1}, \dots, v_n\}, v_i)|\,. \tag{4.9}$$

Now let $G_i = G - \{v_{i+1}, \dots, v_n\}$. We show that $|\mathsf{EC}_k(G_i, v_i)|$ can be computed using an oracle for #EDGEINJ($\mathcal{P}$):

First, we construct the graph G'_i by adding vertices s and t and edges $\{s, v_i\}$ and $\{t, v_i\}$. For $M \subseteq \{s, t\}$ let $A_{i,M}$ be the set of edge-disjoint paths of length $k + 2$ that do not pass through a vertex $u \in M$ and let $G'_{i,M}$ be the graph obtained from G'_i by removing every vertex that lives in M. Note that $|A_M|$ can be computed by querying the oracle for P_{k+1} and $G'_{i,M}$ and dividing by 2. Now it holds that for all $i \in \{1, \dots, |V(G)|\}$:

$$|\mathsf{EC}_k(G_i, v_i)| = |A_{i,\emptyset} \setminus (A_{i,\{s\}} \cup A_{i,\{t\}})| = |A_{i,\emptyset}| - |A_{i,\{s\}}| - |A_{i,\{t\}}| + |A_{i,\{s,t\}}|\,,$$

where the last equality is an easy application of the inclusion-exclusion principle (Theorem 2.27). Finally the values of $|\mathsf{EC}_k(G_i, v_i)|$ for all i in $\{1, \dots, |V(G)|\}$ allow us to compute $|\mathsf{EC}_k(G)|$ (see Equation 4.9), which equals $1/2 \cdot \#\mathsf{EdgeInj}(C_k \to G)$. ■

Proof (of Theorem 4.18). #W[1]-hardness follows from Corollary 4.42 and Lemma 4.43. Equivalence for #W[1] hence holds by Theorem 4.16. ■

Chapter 5

Induced Subgraphs

In the current chapter, we turn to the problem of counting induced subgraphs. To this end, recall that $\mathsf{IndSub}(H \to G)$ denotes the set of all induced subgraphs of a graph G that are isomorphic to H or, in other words, the set of all vertex subsets $S \subseteq V(G)$ such that $G[S] \simeq H$. Similar to the previous chapters, we consider the problem of counting induced subgraphs with respect to the class of allowed graphs for H. More precisely, given a class of graphs $\mathcal{H}$, we let #IndSub($\mathcal{H}$) be the problem of, given $H \in \mathcal{H}$ and an arbitrary graph G, computing $\#\mathsf{IndSub}(H \to G)$; the problem is parameterized by $|V(H)|$.

Example 5.1. The problem #IndSub($\mathcal{H}$) is equivalent to

- #Clique if $\mathcal{H}$ is the class of all complete graphs,
- counting induced matchings if $\mathcal{H}$ is the class of all matchings,
- counting independent sets if $\mathcal{H}$ is the class of all graphs without edges,
- counting induced cycles if $\mathcal{H}$ is the class of all cycles.

Chen, Thurley and Weyer [35] established the following exhaustive classification for #IndSub($\mathcal{H}$).

Theorem 5.2 (Corollary 4 in [35]). *The problem #IndSub($\mathcal{H}$) is solvable in polynomial time if $\mathcal{H}$ is finite. Otherwise, if $\mathcal{H}$ is additionally recursively enumerable, #IndSub($\mathcal{H}$) is $\#\mathsf{W}[1]$-equivalent.*

Corollary 5.3. *All problems in Example 5.1 are $\#\mathsf{W}[1]$-equivalent.*

While Theorem 5.2 resolves the parameterized complexity of #IndSub($\mathcal{H}$), it is not applicable to the more general problem of counting induced subgraphs that satisfy a given graph property, such as the problem of counting connected induced subgraphs of size k. For this reason, Jerrum and Meeks [79, 80, 101, 81] introduced and studied the following problem.

Definition 5.4. Let Φ be a (computable) graph property, that is, Φ is a function from graphs to $\{0,1\}$ such that $\Phi(H) = \Phi(H')$ whenever H is isomorphic to H'.

Then the problem #INDSUB(Φ) asks, given a graph G and a natural number k, to count all induced subgraphs of size k in G that satisfy Φ; the problem is parameterized by k.

Strictly speaking, #INDSUB(Φ) is the unlabeled version of p-#INDUCED SUBGRAPH WITH PROPERTY(Φ), both of which have been introduced in [79]. However, as Jerrum and Meeks point out, those problems are equivalent for graph properties that are invariant under relabeling of vertices (see Section 1.3.1 in [79]). The generality of #INDSUB(Φ) allows to count almost arbitrary substructures in graphs, subsuming lots of parameterized counting problems that have been studied before, and hence the problem deserves a thorough complexity analysis with respect to the property Φ.

Let us revisit the prior results on the complexity of #INDSUB(Φ). In [79] Jerrum and Meeks prove the problem to be #W[1]-hard if Φ is the property of being connected. In [81] hardness is established for the property of having an even (or odd) number of edges and in [80] they prove the problem to be #W[1]-hard whenever the edge-density of graphs with k vertices for which Φ holds, grows asymptotically slower than k^2. Finally, it is shown by Meeks in [101] that whenever Φ is closed under the addition of edges, and the set of (edge-)minimal elements of Φ has unbounded treewidth, the problem is hard as well.

In what follows, we will construct quantum graphs that allow us to express the number of induced subgraphs of size k that satisfy Φ as a linear combination of homomorphisms. Complexity monotonicity will then allow for an almost exhaustive classification of #INDSUB(Φ), subsuming Theorem 5.2 as well as the hardness results in [79, 81] and parts of [101]. In particular, we will obtain matching lower bounds under ETH.

Let us be more precise. In Chapter 5.1, given a property Φ and a positive integer k, we will construct a quantum graph $Q[\Phi,k]$ satisfying that

$$\#\mathsf{Hom}(Q[\Phi,k] \to \star) = \#\mathsf{IndSub}(\Phi, k \to \star)\,, \tag{5.1}$$

where $\mathsf{IndSub}(\Phi, k \to G)$ is the set of induced subgraphs of size k in G that satisfy Φ. For monotone properties Φ, we will prove that the coefficient of K_k in $Q[\Phi,k]$ is, up to a factor of $\pm k!$, equal to the reduced Euler characteristic of a certain simplicial complex induced by Φ and k. This will relate the clique coefficient of the quantum graph to Karp's famous evasiveness conjecture; details and definitions are given in Chapter 5.1. In particular, we will be able to adapt the "topological approach to evasiveness" due to Kahn, Saks and Sturtevant [85] and use a fixed-point theorem (Theorem 2.33) to restrict #INDSUB(Φ) to instances that are invariant under the actions of prime-power groups. More concretely, we prove the following.

Theorem 5.5. *Let Φ be a computable monotone*[1] *property such that Φ and $\neg\Phi$ hold on infinitely many graphs. Then the problem $\#\textsc{IndSub}(\Phi)$ is $\#\mathsf{W}[1]$-equivalent and, assuming ETH, cannot be solved in time $f(k) \cdot n^{o(k)}$ for any computable function f if at least one of the following conditions is true*

(1) Φ is false for odd cycles.

(2) Φ is true for odd anti-holes.

(3) There exists $c \in \mathbb{N}$ such that for all H it holds that $\Phi(H) = 1$ if and only if H is not c-edge-connected.

(4) There exists a graph F such that for all H it holds that $\Phi(H) = 1$ if and only if there is no homomorphism from F to H.

Examples of properties that satisfy the first condition are the ones of being bipartite, cycle-free, disconnected and non-hamiltonian.[2] One example for the second condition is the property of having a chromatic number smaller or equal than half of the size of the graph (rounded up) and the fourth condition includes the properties of exclusion of a fixed complete graph as a subgraph. The results of Chapter 5.1 have been obtained in collaboration with Johannes Schmitt and are published in [120].

While the proof of Theorem 5.5 ultimately relies on the coefficient of K_k in $Q[\Phi, k]$, we strengthen the approach in Chapter 5.1 by considering further constituents of $Q[\Phi, k]$ with large treewidth in Chapter 5.2. For technical reasons, this requires to consider a color-prescribed variant of complexity monotonicity. We will then use the fact that there exist edge-transitive graphs with a prime-power number of edges and arbitrary large treewidth, the former property of which will allow us to rely on the theory of Sylow groups to obtain an easy strategy to prove that such graphs are contained in the support of implicitly H-colored quantum graphs. This algebraic approach will then induce, as a special case, the following exhaustive classification of $\#\textsc{IndSub}(\Phi)$ restricted to bipartite graphs.

Theorem 5.6. *Let Φ be a computable monotone*[3] *property such that Φ and $\neg\Phi$ hold on infinitely many bipartite graphs. Then $\#\textsc{IndSub}(\Phi)$ is $\#\mathsf{W}[1]$-equivalent and cannot be solved in time $f(k) \cdot n^{o(k)}$ for any computable function f unless the Exponential Time Hypothesis fails. This holds true even if the input graphs to $\#\textsc{IndSub}(\Phi)$ are restricted to be bipartite.*

[1]For properties that are edge-monotone but not monotone, e.g. being disconnected or non-hamiltonian, $\#\mathsf{W}[1]$-equivalence is obtained as well, but a tight lower bound under ETH requires a further condition. Details are given in the respective sections.

[2]See Footnote 1.

[3]See Footnote 1.

The results of Chapter 5.2 have been obtained in collaboration with Julian Dörfler, Johannes Schmitt and Philipp Wellnitz and will be published in [52]; a preliminary full version can be found in [53].
We will now introduce the most important notions and results that are required for both, the topological and the algebraic approach. First of all, we will from now on implicitly assume all graph properties to be computable. Recall that a property Φ is *edge-monotone* if it is closed under the removal of edges, that is, whenever H' is an edge-subgraph of H and Φ holds on H then Φ holds on H' as well. Φ is called *monotone* if it is closed under the removal of edges and vertices, that is, whenever H' is a subgraph of H and Φ holds on H then Φ holds on H' as well.

Given a property Φ, we write $\mathcal{K}[\Phi]$ for the set of all k such that Φ is neither constant 1 nor constant 0 on the set of all graphs with k vertices, that is,

$$\mathcal{K}[\Phi] := \left\{k \in \mathbb{N} \;\middle|\; \exists H, H' : |V(H)| = |V(H')| = k \wedge \Phi(H) \neq \Phi(H')\right\} .$$

We call a property Φ *trivial* if $\mathcal{K}[\Phi]$ is finite. This definition is motivated by the following lemma.

Lemma 5.7. #INDSUB(Φ) *is fixed-parameter tractable if* Φ *is trivial.*

Proof. Let B be a constant upper bound on positive integers k for which there exist graphs H and H' such that $\Phi(H) \neq \Phi(H')$. On input G and k, if $k \leq B$, we solve the problem by brute-force in time

$$f(k) \cdot \binom{n}{k} ,$$

by enumerating all vertex subsets of size k and testing in time $f(k)$ for some computable function f whether the property holds on the respective subgraph. If $k > B$ we check whether $\Phi(K_k)$ holds in time $f(k)$. If this is the case, then, by assumption, all graphs with k vertices satisfy Φ and we output $\binom{n}{k}$. Otherwise Φ holds on no graph with k vertices and we output 0. The overall running time is hence bounded by

$$f(k) \cdot \binom{n}{B} \leq f(k) \cdot n^B ,$$

where n is the number of vertices of G. ∎

To the day this thesis is written and to the best of the author's knowledge there is not a single example of a non-trivial property Φ for which #INDSUB(Φ) is fixed-parameter tractable. We conjecture that no such property exists unless all #W[1]-equivalent problems are fixed-parameter tractable. Weak evidence for this claim is given by the following, implicit classification of #INDSUB(Φ).

Theorem 5.8 ([41]). *The problem #IndSub(Φ) is either fixed-parameter tractable or #W[1]-equivalent for all properties Φ.*

Theorem 5.8 follows by the existence of the quantum graph $Q[\Phi, k]$ (5.1) which is constructed in Chapter 5.1, the complexity monotonicity property (see Theorem 3.13) and the classification of counting homomorphisms (see Theorem 2.34). In particular, and similar to the preceding chapters, this illustrates once more that proving the existence of a quantum graph is much less involved than proving which graphs are contained in its support.

For the fine-grained reductions that yield matching lower bounds under ETH for #IndSub(Φ) we rely on the notion of dense sets. Here, a set $\mathcal{K}$ is called *dense* if there exists a constant $c > 0$ such that for all but finitely many $k \in \mathbb{N}$ there exists $k' \in \mathcal{K}$ satisfying that $k \leq k' \leq ck$.

We conclude our preparations for the subsequent chapters with the following simple yet important facts about #IndSub(Φ).

Fact 5.9. *Let G be a graph with n vertices. Then*

$$\#\mathsf{IndSub}(\Phi, k \to G) = \binom{n}{k} - \#\mathsf{IndSub}(\neg\Phi, k \to G)\,.$$

Lemma 5.10 ([79]). #IndSub(Φ) *is* #W[1]*-easy.*

5.1 A Topological Approach to Hardness

To begin with, we prove the existence of a quantum graph that, given a graph property Φ and a positive integer k, allows to express the number of induced subgraphs of size k that satisfy Φ by a linear combination of homomorphisms. To this end, we say that a graph H is *labeled* if it comes with a bijective labeling

$$\ell : V(H) \to \big[|V(H)|\big]\,.$$

Furthermore we write E_k^Φ for the set of all edge-subsets A of the labeled complete graph K_k with k vertices such that Φ holds on the graph $K_k[A]$.

Theorem 5.11. *Let Φ be a graph property and let k be a positive integer. Then there exists a quantum graph $Q[\Phi, k]$ such that*

$$\#\mathsf{Hom}(Q[\Phi, k] \to \star) = \#\mathsf{IndSub}(\Phi, k \to \star)\,.$$

In particular, the mapping $k \mapsto Q[\Phi, k]$ is computable if Φ is fixed and the coefficient $\gamma[\Phi, k]$ of the clique of size k in $Q[\Phi, k]$ satisfies

$$k! \cdot \gamma[\Phi, k] = \pm \sum_{A \in \mathsf{E}_k^\Phi} (-1)^{\#A}\,.$$

Proof. Using the principle of inclusion-exclusion (Theorem 2.27) we can express the number of strong embeddings in terms of the number of embeddings [95, Chapt. 5.2.3]:

$$\#\mathsf{StrEmb}(H \to G) = \sum_{\substack{H' \supseteq H \\ V(H)=V(H')}} (-1)^{\#E(H')-\#E(H)} \cdot \#\mathsf{Emb}(H' \to G)\,, \tag{5.2}$$

where H' ranges over all graphs obtained from H by adding edges. Next we collect terms in (5.2) that correspond to isomorphic graphs. To this end we let $\#\{H' \supseteq H\}$ denote the number of possibilities to add edges to H such that the resulting graph is isomorphic to H'. Note that, in particular, $\#\{K_k \supseteq H\} = 1$ if H has k vertices. We obtain

$$\#\mathsf{StrEmb}(H \to G) = \sum_{H'} (-1)^{\#E(H')-\#E(H)} \cdot \#\{H' \supseteq H\} \cdot \#\mathsf{Emb}(H' \to G)\,, \tag{5.3}$$

where the sum is over all (isomorphism types of) graphs. Next we invoke Möbius inversion to translate the number of embeddings to a linear combination of homomorphisms as given by Equation (2.3) in Chapter 2.4.2:

$$\#\mathsf{Emb}(H' \to G) = \sum_{\rho \geq \emptyset} \mu(\emptyset, \rho) \cdot \#\mathsf{Hom}(H'/\rho \to G)\,, \tag{5.4}$$

where the sum and the Möbius function μ are over the partition lattice of $V(H')$. We observe that the coefficient of $\#\mathsf{Hom}(K_k \to G)$ in the above sum is $\mu(\emptyset, \emptyset) = 1$ if H' is isomorphic to K_k and zero otherwise as every quotient of a graph with k vertices that is not the complete graph cannot result in the complete graph with k vertices. Hence the coefficient of $\#\mathsf{Hom}(K_k \to G)$ in Equation (5.3) is precisely

$$(-1)^{\#E(K_k)-\#E(H)}\,.$$

Now let Φ_k denote the set of all graphs with k vertices that satisfy Φ. Using Fact 2.9, we invoke transformations (5.3) and (5.4) successively and obtain

$$\begin{aligned}
&\#\mathsf{IndSub}(\Phi, k \to G) \\
\overset{\text{(Def)}}{=} &\sum_{H \in \Phi_k} \#\mathsf{IndSub}(H \to G) \\
\overset{(2.9)}{=} &\sum_{H \in \Phi_k} \#\mathsf{Aut}(H)^{-1} \#\mathsf{StrEmb}(H \to G) \\
\overset{(5.3)}{=} &\sum_{H \in \Phi_k} \#\mathsf{Aut}(H)^{-1} \sum_{H'} (-1)^{\#E(H')-\#E(H)} \#\{H' \supseteq H\} \#\mathsf{Emb}(H' \to G) \\
\overset{(5.4)}{=} &\sum_{H \in \Phi_k} \#\mathsf{Aut}(H)^{-1} \sum_{H'} (-1)^{\#E(H')-\#E(H)} \#\{H' \supseteq H\} \sum_{\rho \geq \emptyset} \mu(\emptyset, \rho) \#\mathsf{Hom}(H'/\rho \to G)\,.
\end{aligned}$$

The quantum graph $Q[\Phi, k]$ can hence be constructed by collecting for isomorphic terms in the preceding sum and deleting quotients graphs with self-loops. In particular, the prior observations regarding the complete graph with k vertices, namely that $\#\{K_k \supseteq H\} = 1$ if H has k vertices and that no quotient of a graph $H \not\simeq K_k$ with k vertices is isomorphic to K_k, imply that

$$\gamma[\Phi, k] = \sum_{H \in \Phi_k} (-1)^{\#E(K_k) - \#E(H)} \cdot \#\mathsf{Aut}(H)^{-1} . \tag{5.5}$$

We now multiply this equation by $k!$, which we interpret as the number $\#\mathsf{Sym}_k$ of elements of the symmetric group of the k vertices. Taking also the absolute value on both sides allows us to drop the constant factor $(-1)^{\#E(K_k)}$ and we obtain

$$|k! \cdot \gamma[\Phi, k]| = \left| \sum_{H \in \Phi_k} (-1)^{\#E(H)} \cdot \frac{\#\mathsf{Sym}_k}{\#\mathsf{Aut}(H)} \right| . \tag{5.6}$$

For any graph H in the above sum choose a set A_0 of edges of the labeled complete graph K_k on k vertices such that the corresponding edge-subgraph $K_k[A_0]$ is isomorphic to H. The group Sym_k acts on the vertices and thus on the edges of K_k and by the definition of a graph automorphism, the stabilizer of the set A_0 has exactly $\#\mathsf{Aut}(H)$ elements. On the other hand the orbit of A_0 under Sym_k is the collection of all sets A such that $K_k[A] \simeq H$. By the Orbit-Stabilizer-Theorem (Theorem 2.29) we have

$$\frac{\#\mathsf{Sym}_k}{\#\mathsf{Aut}(H)} = \#\{A \subseteq E(K_k) \mid K_k[A] \simeq H\} .$$

Inserting in equation (5.6) we obtain

$$|k! \cdot \gamma[\Phi, k]| = \left| \sum_{H \in \Phi_k} \sum_{\substack{A \subseteq E(K_k) \\ K_k[A] \simeq H}} (-1)^{\#E(H)} \right| = \left| \sum_{A \in \mathsf{E}_k^\Phi} (-1)^{\#A} \right| . \tag{5.7}$$

■

The following hardness result is an immediate consequence of Theorem 5.11.

Corollary 5.12. *Let Φ be a graph property and let*

$$\mathcal{K} := \left\{ k \in \mathbb{N} \;\middle|\; \sum_{A \in \mathsf{E}_k^\Phi} (-1)^{\#A} \neq 0 \right\} .$$

If $\mathcal{K}$ is infinite, then $\#$IndSub(Φ) is $\#\mathsf{W}[1]$-hard. If additionally $\mathcal{K}$ is dense, $\#$IndSub(Φ) cannot be solved in time $f(k) \cdot \#V(G)^{o(k)}$ for any computable function f, unless ETH fails.

Proof. Complexity monotonicity (Theorem 3.13) induces a parameterized Turing reduction from $\#\textsc{Hom}(\{K_k \mid k \in \mathcal{K}\})$, that is,

$$\#\textsc{Hom}(\{K_k \mid k \in \mathcal{K}\}) \leq^{\mathrm{T}}_{\mathrm{fpt}} \#\textsc{IndSub}(\Phi)\,,$$

as by Theorem 5.11, $\gamma[\Phi, k] \neq 0$ whenever $k \in \mathcal{K}$. If $\mathcal{K}$ is infinite, then $\{K_k \mid k \in \mathcal{K}\}$ has unbounded treewidth and hence $\#\textsc{Hom}(\{K_k \mid k \in \mathcal{K}\})$ is $\#\mathsf{W}[1]$-hard by Theorem 2.34. For the lower bound under ETH we rely on Theorem 2.18 and construct a simple reduction

$$\#\textsc{Clique} \leq^{\mathrm{T}}_{\mathrm{fpt}} \#\textsc{Hom}(\{K_k \mid k \in \mathcal{K}\})\,.$$

In particular, density of $\mathcal{K}$ guarantees that, on input (k, G) for $\#\textsc{Clique}$, every oracle query $(K_{k'}, G')$ satisfies that

$$k' \in O(k) \;\wedge\; |V(G')| \leq f(k) \cdot |V(G)|\,.$$

A detailed construction of the reduction can be found in [120]. Consequently, any algorithm solving $\#\textsc{IndSub}(\Phi)$ and hence $\#\textsc{Hom}(\{K_k \mid k \in \mathcal{K}\})$ in time

$$f(k) \cdot \#V(G)^{o(k)}$$

yields an algorithm with similar running time for $\#\textsc{Clique}$ which refutes ETH by Theorem 2.18. ■

Recall that $\#\textsc{IndSub}(\Phi)$ is known to be $\#\mathsf{W}[1]$-hard if Φ is the property of having an odd (or even) number of edges [81]. We point out that Corollary 5.12 immediately implies this result as the sum

$$\sum_{A \in \mathsf{E}_k^{\Phi}} (-1)^{\#A}$$

is clearly non-zero for every positive integer k as all terms have the same sign. Further hardness results for $\#\textsc{IndSub}(\Phi)$ that are obtained by invoking Corollary 5.12 directly can be found in [120, Section 5]. In what follows we will concentrate on (edge-)monotone properties.

5.1.1 Simplicial Graph Complexes and Evasiveness

One of the most important structural insights of this thesis relates the clique coefficient $\gamma[\Phi, k]$ for monotone Φ to topological properties of simplicial graph complexes and, ultimately, to what is called the evasiveness conjecture. Before we go into the details, we which to point out that the subsequent connections have been discovered by implementing sequences of clique coefficients and searching in Sloane's On-Line Encyclopedia of Integer Sequences [122].

Recall that a simplicial complex Δ is a set of non-empty sets closed under taking non-empty subsets (see Chapter 2.4.4). Given an edge-monotone property Φ and a positive integer k, the set $\mathsf{E}_k^\Phi \setminus \{\emptyset\}$ is a simplicial complex, called the *simplicial graph complex*, denoted by $\Delta[\Phi, k]$. Recall further, that the reduced Euler characteristic of a simplicial complex is given by Fact 2.32:

$$\hat{\chi}(\Delta) = \sum_{i \geq 0} (-1)^i \cdot \#\{A \in \Delta \cup \{\emptyset\} \mid \#A = i\}\,. \tag{5.8}$$

Remark 5.13. For every edge-monotone property Φ and $k \in \mathcal{K}[\Phi]$ we have that Φ holds on the independent set of size k, i.e., $\Phi(K_k[\emptyset]) = 1$; assuming otherwise Φ would be trivially false on k-vertex graphs by edge-monotonicity and hence $k \notin \mathcal{K}[\Phi]$.

The preceding remark implies that for edge-monotone properties, the empty set is contained in E_k^Φ if $k \in \mathcal{K}[\Phi]$. In this case we obtain the following.

Lemma 5.14. $k! \cdot \gamma[\Phi, k] = \pm\hat{\chi}(\Delta[\Phi, k])$ *whenever* $k \in \mathcal{K}[\Phi]$.

Proof. Holds by Theorem 5.11, Fact 2.32 and Remark 5.13. ∎

Consequently, by Corollary 5.12 the problem #IndSub(Φ) is #W[1]-hard if $\hat{\chi}(\Delta[\Phi, k]) \neq 0$ and $k \in \mathcal{K}[\Phi]$ for infinitely many k. Fortunately, the reduced Euler characteristic of simplicial graph complexes is a very well studied object; going into the details of the topological interpretations is beyond the scope of this thesis, hence we refer the interested reader to the excellent survey of Miller [102]. However, we wish to take a small detour to Karp's evasiveness conjecture. The reason for this is the fact that some of the existing tools for proving special cases of this conjecture can be used in our setting as well.

Let us start with a rough introduction to decision tree complexity; consult [102, Chapt. 1 and 2.1] for a more detailed exposition. To this end, assume that the edges of a graph are only provided by an oracle. That is, an algorithm is given the set of vertices V of a graph, but it is required to pose an oracle query for two vertices u and v to find out whether $\{u, v\}$ is an edge of the graph. Such algorithms are called *decision tree algorithms* and we say that a decision tree algorithm *computes* a graph property Φ if, given a graph H, the algorithm correctly decides whether $\Phi(H)$ holds. The *decision tree complexity* of Φ is the function $D(\Phi)$ that maps a positive integer k to the value $D_k(\Phi)$ which is defined to be the minimum number of oracle queries a deterministic decision tree algorithm that computes Φ has to perform in the worst case on graphs with k vertices.

Definition 5.15. A property Φ is called *evasive on k-vertex graphs* if

$$D_k(\Phi) = \binom{k}{2}.$$

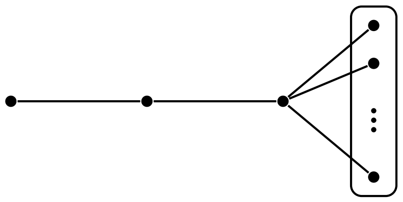


Figure 5.1: A scorpion graph. The leftmost vertex is called the *sting*, the vertex next to the sting is called the *tail*, and the vertex right to the tail is called the *body*. The remaining vertices of the graph may be connected to each other arbitrarily and must be connected to the body but not to sting and tail.

Intuitively, the preceding definition states that Φ is evasive if every possible edge needs to be queried to correctly decide whether Φ holds.

Example 5.16 ([11]).

- Connectivity is evasive on k-vertex graphs for $k > 0$.
- Planarity is evasive on k-vertex graphs for $k > 4$.

Example 5.17 ([11]). The property of being a scorpion graph (Figure 5.1) is *not* evasive on k-vertex graphs for all but finitely many k.

The study of evasive graph properties and decision tree algorithms goes back to the early 70s; the interested reader is again referred to Miller's survey [102, Chapt. 1.1] which provides a detailed historical background. We will focus only on the most important milestones.
To begin with, we observe that the third item of Example 5.17 proves the existence of non-trivial graph properties that are evasive on k-vertex graphs for all but finitely many k. Further, the property of being a scorpion graph is not edge-monotone. The following conjecture, known as Karp's evasiveness conjecture or the Aanderaa-Karp-Rosenberg conjecture states that there are no non-trivial properties in case of edge-monotonicity.

Conjecture 5.18 (Evasiveness conjecture [117, 102]). Let k be a positive integer and let Φ be an edge-monotone property that is neither true nor false on all k-vertex graphs. Then Φ is evasive on k-vertex graphs.

The asymptotic version of the evasiveness conjecture was proved by Rivest and Vuillemin [112]. The next major result was due to Kahn, Saks and Sturtevant [85] in their breakthrough paper "A Topological Approach to Evasiveness":

Theorem 5.19 ([85]). *Conjecture 5.18 is true for prime-powers k.*

Their idea was to use topological properties of simplicial graph complexes induced by the edge-monotone properties. In particular, they relied on the (reduced) Euler characteristic in the following way.[4].

Theorem 5.20 ([85]). *If $\hat{\chi}(\Delta[\Phi,k]) \neq 0$ then Φ is evasive on k-vertex graphs.*

Corollary 5.21. *If Φ is edge-monotone and $\gamma[\Phi,k] \neq 0$ then Φ is evasive on k-vertex graphs.*

Proof. If $\gamma[\Phi,k] \neq 0$ then, by Theorem 5.11,

$$\sum_{A \in \mathsf{E}_k^\Phi} (-1)^{\#A} \neq 0\,.$$

Therefore, E_k^Φ is neither empty, nor the set of all edge subsets of the labeled complete graph K_k. It follows that $k \in \mathcal{K}[\Phi]$. The claim holds hence by Lemma 5.14 and Theorem 5.20. ∎

Consequently, the existence of the clique K_k as a constituent of the quantum graph $Q[\Phi,k]$ does not only rule out an $f(k)\cdot n^{o(k)}$ algorithm for computing $\#\mathsf{IndSub}(\Phi, k \to \star)$ under ETH (see Corollary 5.12) but also, unconditionally, rules out a decision tree algorithm for computing Φ that beats brute-force.

Unfortunately, the converse of Theorem 5.20 is not true. A counterexample is given in Chapt. 10.6 in Jonsson's book [84]. Therefore, we cannot adapt all tools of the topological approach to evasiveness [85] to our case. However, Smith's fixed-point theorem (see Theorem 2.33) will suffice for our purposes. To this end, recall that, given a simplicial complex Δ and a group Γ that acts on the ground set of Δ , we say that Δ is a Γ-simplicial complex if for every $A \in \Delta$ and $g \in \Gamma$ then the set $g \triangleright A := \{g \triangleright a \mid a \in A\}$ is contained in Δ as well. Recall further that the fixed-point complex Δ^Γ of a Γ-simplicial complex Δ is defined as

$$\Delta^\Gamma := \left\{ S \subseteq \{1,\dots,k\} \;\middle|\; S \neq \emptyset \wedge \bigcup_{i \in S} \mathcal{O}_i \in \Delta \right\},$$

where $\mathcal{O}_i$ are the orbits of the group action. Now Theorem 2.33 states that

$$\hat{\chi}(\Delta) \equiv \hat{\chi}(\Delta^\Gamma) \mod p\,, \tag{5.9}$$

if Γ is of order p^ℓ for some prime p and positive integer ℓ. For our purposes it will suffice to use the groups $\mathbb{Z}_p$ for prime numbers p, explained as follows.

[4]In fact, Kahn, Saks and Sturtevant show that any non-evasive complex is collapsible. However, every collapsible complex has a reduced Euler characteristic of zero (see e.g. [96]). Hence the contraposition implies the Theorem 5.20 as stated.

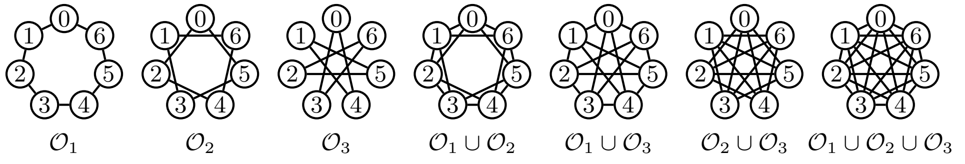

$\mathcal{O}_1$
$\mathcal{O}_2$
$\mathcal{O}_3$
$\mathcal{O}_1 \cup \mathcal{O}_2$
$\mathcal{O}_1 \cup \mathcal{O}_3$
$\mathcal{O}_2 \cup \mathcal{O}_3$
$\mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{O}_3$

Figure 5.2: Non-empty unions of orbits of the action of $\mathbb{Z}_7$ on the edge set of the labeled graph with 7 vertices. If Φ is trivially true then $\Delta^{\mathbb{Z}_7}[\Phi, 7]$ contains all of the above subsets of orbits. If Φ holds only for bipartite graphs then none of the above subsets is contained in $\Delta^{\mathbb{Z}_7}[\Phi, 7]$. If Φ is planarity then $\Delta^{\mathbb{Z}_7}[\Phi, 7] = \{\mathcal{O}_1, \mathcal{O}_2, \mathcal{O}_3\}$. More exotically, if Φ is the property of not being 5-edge-connected then $\Delta^{\mathbb{Z}_7}[\Phi, 7]$ contains every subset of orbits except for $\mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{O}_3$.

Recall that the ground set of $\Delta[\Phi, p]$ is the set of all edges of the labeled complete graph on p vertices. Now $b \in \mathbb{Z}_p$ is interpreted as a relabeling $x \mapsto x + b$ of the vertices[5], which induces an action on the edges by mapping the edge $\{x, y\}$ to the edge $\{x + b, y + b\}$. We remark that this group was also used in an intermediate step in [85]. It can easily be verified that this mapping is a group action. Furthermore $\Delta[\Phi, p]$ is a $\mathbb{Z}_p$-simplicial complex with respect to this action as Φ is invariant under relabeling of vertices. Hence the fixed-point complex

$$\Delta^{\mathbb{Z}_p}[\Phi, p] := \Delta[\Phi, p]^{\mathbb{Z}_p}$$

is well-defined. Furthermore observe that every orbit of the group action is a Hamilton cycle. We illustrate $\Delta^{\mathbb{Z}_7}[\Phi, 7]$ for some properties Φ in Figure 5.2. Note that, given a prime $p > 2$, the ground set of $\Delta^{\mathbb{Z}_p}[\Phi, p]$ consists of exactly $\frac{1}{2}(p-1)$ elements. In particular those are the Hamiltonian cycles

$$\begin{aligned} H_1 &= (0, 1, 2, \dots) \\ H_2 &= (0, 2, 4, \dots) \\ H_3 &= (0, 3, 6, \dots) \\ &\vdots \\ H_{\frac{1}{2}(p-1)} &= \left(0, \frac{1}{2}(p-1), p-1, \dots\right). \end{aligned}$$

Equivalently, H_i is the orbit of the (labeled) edge $\{0, i\}$ under the action of $\mathbb{Z}_p$ for $i \in \{1, \dots, \frac{1}{2}(p-1)\}$ and it can easily be verified that those are all orbits of the group action. In what follows, given a non-empty set

$$P \subseteq \{1, \dots, \frac{1}{2}(p-1)\},$$

we write H_P for the graph with vertices (labeled with) $[p]$ and edges $\bigcup_{i \in P} H_i$.

[5]Here $+$ is addition modulo p.

Fact 5.22. *Let P be non-empty subset of $\{1,\dots,\frac{1}{2}(p-1)\}$. Then*

$$P \in \Delta^{\mathbb{Z}_p}[\Phi,p] \Leftrightarrow \Phi(H_P)=1\,.$$

Now we have everything we need to prove all cases of Theorem 5.5. In particular, we emphasize that using the fixed-point complex $\Delta^{\mathbb{Z}_p}[\Phi,p]$ instead of the primal complex $\Delta[\Phi,p]$ allows us to give surprisingly simple proofs, while at the same time Theorem 5.5 significantly extends prior results on the complexity of #IndSub(Φ). We start with edge-monotone properties that are false on odd cycles or true on odd antiholes.

Lemma 5.23. *If Φ is edge-monotone and does not hold on odd cycles then*

$$\hat{\chi}(\Delta^{\mathbb{Z}_p}[\Phi,p]) \not\equiv 0 \mod p$$

for every prime $p>2$ in $\mathcal{K}[\Phi]$.

Proof. If Φ does not hold on odd cycles then $\Delta^{\mathbb{Z}_p}[\Phi,p]=\emptyset$, because all non-empty unions of orbits contain an odd cycle as edge-subgraph and Φ is edge-monotone. Hence, by (5.8),

$$\hat{\chi}(\Delta^{\mathbb{Z}_p}[\Phi,p]) = 1-0=1\not\equiv 0 \mod p\,.$$

■

Lemma 5.24. *If Φ is edge-monotone and holds on odd anti-holes[6] then*

$$\hat{\chi}(\Delta^{\mathbb{Z}_p}[\Phi,p]) \not\equiv 0 \mod p$$

for every prime $p\in\mathcal{K}[\Phi]$.

Proof. If Φ holds on odd anti-holes then

$$\Delta^{\mathbb{Z}_p}[\Phi,p]=\{P \mid \emptyset \subsetneq P \subsetneq \{1,\dots,\frac{1}{2}(p-1)\}\}\,,$$

because every union of all but one orbit constitutes an anti-hole and Φ is edge-monotone and, furthermore, Φ must be false on the union of all orbits which constitutes a clique, because otherwise, by edge-monotonicity Φ would hold on all graphs with p vertices and hence $p\notin\mathcal{K}[\Phi]$. Consequently,

$$\begin{aligned}
&\hat{\chi}(\Delta^{\mathbb{Z}_p}[\Phi,p])\\
&=\sum_{i\geq 0}(-1)^i\cdot\#\{P\in\Delta^{\mathbb{Z}_p}[\Phi,p]\cup\{\emptyset\}\mid \#P=i\}\\
&=\sum_{i\geq 0}(-1)^i\cdot\#\{P\subsetneq\{1,\dots,\frac{1}{2}(p-1)\}\mid \#P=i\}\\
&=\left(\sum_{P\subseteq\{1,\dots,\frac{1}{2}(p-1)\}}(-1)^{\#P}\right)-(-1)^{\frac{1}{2}(p-1)}=(-1)^{\frac{1}{2}(p-1)+1}\not\equiv 0 \mod p\,.
\end{aligned}$$

Note that the first equality holds by (5.8). ■

[6]Recall that an anti-hole is the complement of a cycle, that is a graph (V,E) such that its complement $(V,\overline{E})$ is a cycle.

Now let us combine the previous lemmas to obtain the following hardness result.

Theorem 5.25. *Let Φ be an edge-monotone graph property and let $\mathcal{P}$ be the set of all primes $p > 2$ in $\mathcal{K}[\Phi]$. Then $\#\textsc{IndSub}(\Phi)$ is $\#\mathsf{W}[1]$-equivalent if $\mathcal{P}$ is infinite and*

(1) Φ is false on odd cycles, or

(2) Φ is true on odd anti-holes.

If additionally $\mathcal{P}$ is dense then the problem cannot be solved in time

$$f(k) \cdot |V(G)|^{o(k)}$$

for any computable function unless ETH fails.

Proof. We invoke Corollary 5.12 for $\mathcal{K} = \mathcal{P}$ which requires us to show that

$$\sum_{A \in \mathsf{E}_p^\Phi} (-1)^{\#A} \neq 0 \tag{5.10}$$

for all $p \in \mathcal{P}$. By Theorem 5.11 and Lemma 5.14 we have that (5.10) is equivalent to

$$\hat{\chi}(\Delta[\Phi, p]) \neq 0\,,$$

because $p \in \mathcal{P} \subseteq \mathcal{K}[\Phi]$ by assumption. Now the latter inequality holds by application of Smith's fixed-point theorem (5.9) and Lemma 5.23 if Φ is false on odd cycles, and Lemma 5.24 if Φ is true on odd anti-holes. This shows $\#\mathsf{W}[1]$-hardness; $\#\mathsf{W}[1]$-equivalence follows by Lemma 5.10. ∎

Corollary 5.26. *The problem $\#\textsc{IndSub}(\Phi)$ is $\#\mathsf{W}[1]$-equivalent and cannot be solved in time $f(k) \cdot |V(G)|^{o(k)}$ for any computable function unless ETH fails, if Φ is one of the following graph properties.*

- *$\Phi(H) = 1$ if and only if H is non-hamiltonian.*
- *$\Phi(H) = 1$ if and only if H is disconnected.*
- *$\Phi(H) = 1$ if and only if H is bipartite.*
- *$\Phi(H) = 1$ if and only if H is acyclic.*

The same holds true for the complementary properties.

Proof. Each property satisfies the first item of Theorem 5.25. In particular, the set $\mathcal{P}$ contains all primes $p > 2$ for each property. By Bertrand's postulate $\mathcal{P}$ is hence dense. Finally, the result holds true for the complementary properties by Fact 5.9. ∎

The following lemma simplifies the statement of the previous theorem in case Φ is also closed under the removal of vertices.

Lemma 5.27. *Let Φ be a monotone property such that Φ and $\neg\Phi$ hold on infinitely many graphs. Then $\mathcal{K}[\Phi]$ contains all but finitely many positive integers.*

Proof. Let $k \in \mathbb{N}$. By assumption there exists a graph H with at least k vertices that satisfies Φ. Now the graph $K_k[\emptyset]$, that is, the independent set of size k, is a subgraph of H and hence, by monotonicity of Φ, it holds that $\Phi(K_k[\emptyset]) = 1$. Next let H_0 be a graph such that $\Phi(H_0) = 0$. Further, let $k_0 = |V(H_0)|$ and observe that $\Phi(K_k) = 0$ for every $k \geq k_0$ as Φ is monotone. Consequently, $\mathcal{K}[\Phi]$ contains every $k \geq k_0$. ■

The application of Lemma 5.27 allows us to infer the first two cases of Theorem 5.5 from Theorem 5.25.

Proof (Case (1) and (2) of Theorem 5.5). Follows from Theorem 5.25, as well as from Lemma 5.27. In particular, density of $\mathcal{P}$ is given by Bertrand's postulate. ■

We continue with one more exotic property which illustrates the utility of the topological approach by exploiting the simple structure of $\Delta^{\mathbb{Z}_p}[\Phi, p]$. To this end, recall that a graph is called *c-edge-connected* if it cannot be disconnected by removing at most $c-1$ edges.

Lemma 5.28. *Let $c \in \mathbb{N}$ be an arbitrary constant and let Φ be the graph property of being not $(c+1)$-edge-connected. Then*

$$\hat{\chi}(\Delta^{\mathbb{Z}_p}[\Phi, p]) \not\equiv 0 \mod p$$

for every prime $p > c+3$.

Proof. We rely on the following observation.

Claim 5.29. *The graph H_P is $(c+1)$-edge-connected if and only if*

$$\#P > \left\lfloor \frac{c}{2} \right\rfloor .$$

Proof. If $\#P \leq \lfloor \frac{c}{2} \rfloor$ then every vertex in H_P has degree at most c, hence H_P is not $(c+1)$-edge-connected. If $\#P > \lfloor \frac{c}{2} \rfloor$ then H_P contains at least $\lfloor \frac{c}{2} \rfloor + 1$ pairwise edge-disjoint Hamilton cycles. Disconnecting the graph would require to remove at least two edges from every Hamilton cycle, i.e., at least $2 \cdot (\lfloor \frac{c}{2} \rfloor + 1) \geq c+1$ edges. Hence H_P is $(c+1)$-edge-connected. □

It follows from the Claim that

$$\Delta^{\mathbb{Z}_p}[\Phi, p] = \left\{ P \subseteq \left\{1, \dots, \frac{1}{2}(p-1)\right\} \;\middle|\; P \neq \emptyset \wedge \#P \leq \left\lfloor \frac{c}{2} \right\rfloor \right\}.$$

Hence

$$\begin{aligned}
&\hat{\chi}(\Delta^{\mathbb{Z}_p}[\Phi, p]) \\
&= \sum_{i \geq 0} (-1)^i \cdot \#\{P \in \Delta^{\mathbb{Z}_p}[\Phi, p] \cup \{\emptyset\} \mid \#P = i\} \\
&= \sum_{i=0}^{\lfloor \frac{c}{2} \rfloor} (-1)^i \cdot \#\{P \subseteq \{1, \dots, \frac{1}{2}(p-1)\} \mid \#P = i\} \\
&= \sum_{i=0}^{\lfloor \frac{c}{2} \rfloor} (-1)^i \cdot \binom{\frac{1}{2}(p-1)}{i} = (-1)^{\lfloor \frac{c}{2} \rfloor} \cdot \binom{\frac{1}{2}(p-1) - 1}{\lfloor \frac{c}{2} \rfloor} \not\equiv 0 \mod p\,.
\end{aligned}$$

Note that the first equality holds by (5.8). ■

We are now able to prove the third case of Theorem 5.5, that is, #W[1]-equivalence of #INDSUB(Φ) for the property Φ of not being c-edge-connected.

Proof (Case (3) of Theorem 5.5). We invoke Corollary 5.12 for $\mathcal{K}$ being the set of all primes $p > c + 2$, which requires us to show that

$$\sum_{A \in \mathsf{E}_p^{\Phi}} (-1)^{\#A} \neq 0 \tag{5.11}$$

for all primes $p > c + 2$. By Theorem 5.11 and Lemma 5.14 we have that (5.11) is equivalent to

$$\hat{\chi}(\Delta[\Phi, p]) \neq 0\,,$$

because $p \in \mathcal{K}[\Phi]$ by assumption. The latter inequality holds by by application of Smith's fixed-point theorem (5.9) and Lemma 5.28. This shows #W[1]-hardness; #W[1]-equivalence follows by Lemma 5.10. ■

The last case of Theorem 5.5 follows from a known explicit formula for the reduced Euler characteristic of the graph complex induced by the property of excluding the existence of a homomorphism from a graph F.

Lemma 5.30 ([27]). *Let F be a graph and let $\Phi^{[F]}$ be the graph property that holds true on a graph H if and only if $\mathsf{Hom}(F \to H) = \emptyset$, i.e., there is no homomorphism from F to H. Furthermore let*

$$T_F := \min\{2^{2^t} - 1 \mid 2^{2^t} \geq \#V(F)\}$$

and let $k \in \mathbb{N}$ such that $k \equiv 1 \mod T_F$. Then $\chi(\Delta[\Phi^{[F]}, k]) \equiv 0 \mod 2$ and hence $\hat{\chi}(\Delta[\Phi^{[F]}, k]) \equiv 1 \mod 2$.

Proof (Case (4) of Theorem 5.5). Follows from Lemma 5.30, Theorem 5.11, Corollary 5.12 and Lemma 5.14. Details are given in [120]. ■

5.2 An Algebraic Approach to Hardness

So far, we only considered the coefficient $\gamma[\Phi, k]$ of the k-clique in the quantum graph $Q[\Phi, k]$. However, by complexity monotonicity, it suffices to find an arbitrary graph with large treewidth as constituent of $Q[\Phi, k]$ to establish hardness of counting induced subgraphs of size k satisfying Φ. In the current section, we will analyze the coefficients of graphs H with large treewidth that satisfy the following constraints.

(1) H is edge-transitive, that is, $\mathsf{Aut}(H)$ acts on $E(H)$ transitively (see Chapter 2.4.4), and

(2) the number of edges of H is a prime power p^ℓ.

We will call such graphs *p-edge-transitive* graphs. The algebraic properties of those graphs will allow us to invoke and adapt the results of Rivest and Vuillemin [112] concerning transitive boolean functions over a domain of prime power cardinality. Examples of p-edge-transitive graphs are the bicliques $K_{t,t}$ for prime powers $t = p^\ell$. A complete characterization of p-edge-transitive graphs is given in Chapter 5.2.1. In particular, we will establish an approach for proving hardness of #INDSUB(Φ) if Φ is non-trivial on an infinite set of p-edge-transitive graphs. The following theorems are obtained by the application of this approach to properties of bipartite graphs. Recall that a graph G is called *bipartite* if $V(G)$ can be partitioned in disjoint sets L and R such that every edge of G has one endpoint in L and one endpoint in R; equivalently, G is bipartite if $\mathsf{Hom}(G \to P_1) \neq \emptyset$.

Theorem 5.31. *Let Φ be a computable graph property and let $\mathcal{K}$ be the set of all prime powers t such that $\Phi(\mathsf{IS}_{2t}) \neq \Phi(K_{t,t})$. If $\mathcal{K}$ is infinite then #INDSUB(Φ) is #W[1]-hard. If additionally $\mathcal{K}$ is dense then it cannot be solved in time $f(k) \cdot n^{o(k)}$ for any computable function f unless ETH fails. This holds true even if the input graphs to #INDSUB(Φ) are restricted to be bipartite.*

Recall that a set $\mathcal{K}$ is *dense* if there exists a constant c such that for every $m \in \mathbb{N}$, there exists a $k \in \mathcal{K}$ such that $m \leq k \leq cm$. While the hypotheses of Theorem 5.31 sound technical, the theorem applies in many situations. In particular, it is applicable to properties that are neither (edge-)monotone nor the complement thereof: Let Φ be the property of being Eulerian. The graph $K_{t,t}$ contains an Eulerian cycle if $t = 2^s$ for $s \geq 1$. Hence we can apply Theorem 5.31 with $\mathcal{K} = \{2^s \mid s \geq 1\}$, which is infinite and dense.

Corollary 5.32. *Let Φ be the property of being Eulerian. Then the problem #INDSUB(Φ) is #W[1]-hard and cannot be solved in time $f(k) \cdot n^{o(k)}$ for any computable function f unless the ETH fails. This holds true even if the input graphs to #INDSUB(Φ) are restricted to be bipartite.*

In case Φ is edge-monotone, the condition $\Phi(\mathsf{IS}_{2t}) \neq \Phi(K_{t,t})$ is equivalent to non-triviality and if Φ is monotone, we obtain a more concise statement as given by Theorem 5.6, which we restate for convenience.

Theorem 5.33 (Theorem 5.6 restated). *Let Φ be a computable monotone graph property such that Φ and $\neg\Phi$ hold on infinitely many bipartite graphs. Then* #INDSUB(Φ) *is* #W[1]*-hard and cannot be solved in time $f(k) \cdot n^{o(k)}$ for any computable function f unless the Exponential Time Hypothesis fails. This holds true even if the input graphs to* #INDSUB(Φ) *are restricted to be bipartite.*

Let us illustrate further consequences of the previous theorems with respect to (edge-)monotone properties. First of all, most of the prior hardness results [79, 80, 101, 81, 120] are shown to hold in the restricted case of bipartite graphs. We provide three examples:

Corollary 5.34. *The problem* #INDSUB(Φ)*, restricted to* bipartite input graphs*, is* #W[1]*-hard and cannot be solved in time $f(k) \cdot |V(G)|^{o(k)}$ for any computable function f unless ETH fails, if Φ is one of the properties of being disconnected, planar or non-hamiltonian.*

One example of a monotone property Φ for which the parameterized complexity of #INDSUB(Φ) was unknown, even for general graphs, is given by the following corollary of Theorem 5.33.

Corollary 5.35. *Let F be a fixed bipartite graph with at least one edge and define $\Phi(G) = 1$ if G does not contain a subgraph isomorphic to F. Then* #INDSUB(Φ) *is* #W[1]*-hard and cannot be solved in time $f(k) \cdot |V(G)|^{o(k)}$ for any computable function f unless ETH fails. This holds true even if the input graphs of* #INDSUB(Φ) *are restricted to be bipartite.*

By Fact 5.9, *all of the previous result remain true for the complementary properties $\neg\Phi$.*

5.2.1 Alternating Enumerators of p-Edge-Transitive Graphs

We wish to emphasize that the group theoretic results in this subsection are due to Johannes Schmitt (see also [53, Section 3]); we include them only for reasons of self-containment. Given a property Φ and a graph H, the *alternating enumerator* of Φ and H is defined to be

$$\hat{\chi}(\Phi, H) := \sum_{S \subseteq E(H)} \Phi(H[S]) \cdot (-1)^{\#S}.$$

Roughly speaking, it will turn out that the coefficient of H in the quantum graph $Q[\Phi, \#V(H)]$ is closely related to $\hat{\chi}(\Phi, H)$. We furthermore point out that, in case Φ is closed under the removal of edges, the alternating enumerator $\hat{\chi}(\Phi, H)$ equals the reduced Euler characteristic of the simplicial complex with ground set $E(H)$ and simplices $S \subseteq E(H)$ for which $\Phi(H[S])$ holds.

For what follows, we recall that the automorphism group of a graph H induces a group action on the edges of H, given by $h\{u, v\} := \{h(u), h(v)\}$. Recall further that group action is *transitive* if there exists only one orbit and a graph H is called *edge-transitive* if the group action on the edges is transitive, that is, if for every pair of edges $\{u, v\}$ and $\{\hat{u}, \hat{v}\}$ there exists an automorphism $h \in \mathsf{Aut}(H)$ such that $h\{u, v\} = \{\hat{u}, \hat{v}\}$.

Lemma 5.36. *Let Φ be a graph property and let H be a p-edge-transitive graph such that $\Phi(H[\emptyset]) \neq \Phi(H)$. Then we have that*

$$\hat{\chi}(\Phi, H) = (\pm 1) \mod p\,.$$

Lemma 5.36 is implicitly proved in [112, Theorem 4.3]; for completeness we will include a short and self-contained proof, demonstrating a first application of the machinery of Sylow subgroups that we will need later.

For the proofs in this section, let us recall some key results from group theory. Given a prime number p, a finite group Γ' is called a *p-group* if the order $\#\Gamma'$ is a power of p. The following is a well-known and central result from the theory of finite groups.

Theorem 5.37 (Sylow theorems). *Let Γ be a finite group of order*

$$\#\Gamma = p^k m$$

for a prime p and an integer $m \geq 1$ coprime to p. Then Γ contains a subgroup Γ' of order p^k. Moreover, every other subgroup Γ'' of Γ of order p^k is conjugate to Γ', that is there exists $g \in \Gamma$ with $\Gamma'' = g\Gamma' g^{-1}$. In particular, the groups Γ', Γ'' are isomorphic (via the conjugation by g).

Finally, every subgroup $\tilde{\Gamma} \subseteq \Gamma$ which is a p-group is actually contained in some conjugate $g\Gamma' g^{-1}$ of the group Γ'.

A subgroup $\Gamma' \subseteq \Gamma$ as above is called a *p-Sylow subgroup* of Γ. The following result is a first important application of the Sylow theorems. It can be found as Exercise (E28) in [3]; we include a proof for completeness.

Lemma 5.38. *Let Γ be a finite group acting transitively on a set T such that $\#T = p^l$ for some $l \geq 0$. Then the induced action of any p-Sylow subgroup $\Gamma' \subseteq \Gamma$ on T is still transitive.*

Proof. Let $t_0 \in T$ be any element, then T is the orbit of t_0 under Γ. Let $\mathrm{Stab}_\Gamma(t_0) = \{g \in \Gamma : gt_0 = t_0\}$ be the stabilizer of t_0 under the action of Γ. Then by the Orbit-Stabilizer-Theorem (Theorem 2.29), we have

$$\#\Gamma = (\#\Gamma t_0) \cdot (\#\,\mathrm{Stab}_\Gamma(t_0)) = (\#T) \cdot (\#\,\mathrm{Stab}_\Gamma(t_0)). \tag{5.12}$$

As in the Sylow theorems, write $\#\Gamma = p^k m$ with m not divisible by p and let $\Gamma' \subseteq \Gamma$ be a p-Sylow subgroup of Γ, which is of order p^k. The stabilizer of t_0 under the induced action of the subgroup $\Gamma' \subseteq \Gamma$ is given by

$$\mathrm{Stab}_{\Gamma'}(t_0) = \{g \in \Gamma' : gt_0 = t_0\} = \mathrm{Stab}_{\Gamma}(t_0) \cap \Gamma'.$$

Clearly this is a subgroup of the group Γ' and by Lagrange's theorem, the order of $\mathrm{Stab}_{\Gamma'}(t_0)$ divides the order p^k of Γ'. Thus it is itself a power of p, say $\#\,\mathrm{Stab}_{\Gamma'}(t_0) = p^n$.

On the other hand, $\mathrm{Stab}_{\Gamma'}(t_0)$ is also a subgroup of $\mathrm{Stab}_{\Gamma}(t_0)$. Inserting the order of Γ and the size of T in equation (5.12) we obtain

$$p^k m = p^l \cdot (\#\,\mathrm{Stab}_{\Gamma}(t_0)), \tag{5.13}$$

and thus $\#\,\mathrm{Stab}_{\Gamma}(t_0)$ can at most contain a factor of p^{k-l}. Again, by Lagrange's theorem, the order p^n of the subgroup $\mathrm{Stab}_{\Gamma'}(t_0)$ divides the order of $\mathrm{Stab}_{\Gamma}(t_0)$ and thus $n \leq k - l$.
Finally, by the Orbit-Stabilizer-Theorem applied to the action of Γ' on t_0, we have

$$p^k = \#\Gamma' = (\#\Gamma' t_0) \cdot (\#\,\mathrm{Stab}_{\Gamma'}(t_0)) = (\#\Gamma' t_0) \cdot p^n. \tag{5.14}$$

Thus, on the one hand we obtain $\#\Gamma' t_0 = p^{k-n} \geq p^{k-(k-l)} = p^l$. On the other hand we obtain $\Gamma' t_0 \subseteq T$ and thus $\#\Gamma' t_0 \leq \#T = p^l$. Hence we have the equality $\#\Gamma' t_0 = p^l = \#T$ and thus $\Gamma' t_0 = T$. In other words, the action of Γ' on T is transitive, finishing the proof. ■

This result allows us to give a short proof of Lemma 5.36 above.

Proof (of Lemma 5.36). Let $\Gamma = \mathsf{Aut}(H)$ be the automorphism group of the graph H, then by assumption its action on the set $E(H)$ of edges of H is transitive. By Lemma 5.38, any p-Sylow subgroup $\Gamma' \subseteq \Gamma$ still acts transitively on $E(H)$. Now consider the sum

$$\hat{\chi}(\Phi, H) = \sum_{S \subseteq E(H)} \Phi(H[S]) \cdot (-1)^{\#S}.$$

The action of Γ' on $E(H)$ induces an action of Γ' on the set of subsets $\mathcal{P}(E(H)) := \{S \subseteq E(H)\}$ of $E(H)$. Indeed, for $S \subset E(H)$ and $g \in \Gamma'$ we define $gS = \{gs : s \in S\}$. For this action, the set $\mathcal{P}(E(H))$ can be written as a disjoint union of the orbits $\Gamma' S_0$ of a set $\mathcal{S} \subseteq \mathcal{P}(E(H))$ of representatives S_0. (Recall that for a group action two orbits are either disjoint or equal.) This allows us to write the sum above as

$$\hat{\chi}(\Phi, H) = \sum_{S_0 \in \mathcal{S}} \sum_{S \in \Gamma' S_0} \Phi(H[S]) \cdot (-1)^{\#S}.$$

Until now we have just reordered the summands above, combining all summands for S in the same Γ' orbit.

Now since all elements $g \in \Gamma' \subseteq \mathsf{Aut}(H)$ act by graph automorphisms on H, we have that the graphs $H[gS_0]$ and $H[S_0]$ are isomorphic, so in particular $\Phi(H[gS_0]) = \Phi(H[S_0])$. Applying this to the formula for $\hat{\chi}(\Phi, H)$ above, we get

$$\hat{\chi}(\Phi, H) = \sum_{S_0 \in \mathcal{S}} (\#\Gamma' S_0) \cdot \Phi(H[S_0]) \cdot (-1)^{\# S_0} . \tag{5.15}$$

Now by the Orbit-Stabilizer-Theorem (Theorem 2.29), the size $\#\Gamma' S_0$ of the orbit of S_0 divides the order p^k of Γ', so $\#\Gamma' S_0$ is itself a power of p. Further, unless $S_0 \subseteq E(H)$ is invariant under Γ', the size of its orbit $\#\Gamma' S_0$ is a *positive* power of p and thus congruent to 0 mod p. However, the only two sets $S_0 \subseteq E(H)$ invariant under Γ' are $S_0 = \emptyset$ and $S_0 = E(H)$: Consider for the sake of contradiction the case where S_0 is invariant under Γ' and nonempty, but not the whole set $E(H)$. Then S_0 contains an element e_0, and since S_0 is Γ'-invariant, S_0 also contains the entire orbit of e_0 under Γ'. But since Γ' acted transitively on $E(H)$, S_0 must have been the whole set $E(H)$, yielding a contradiction.

To summarize, when computing $\hat{\chi}(\Phi, H)$ modulo p all but two summands in the sum in Equation (5.15) are congruent to 0. Hence, we can simplify Equation (5.15) to

$$\hat{\chi}(\Phi, H) = \Phi(H[\emptyset]) + \Phi(H[E(H)]) \cdot (-1)^{\# E(H)} = \Phi(H[\emptyset]) - \Phi(H) \mod p .$$

Note that we use the fact that for $p > 2$ we have that $\#E(H)$ is odd since it is a prime power and for $p = 2$ we have $-1 = 1$ modulo p. Now, the condition $\Phi(H[\emptyset]) \neq \Phi(H)$ exactly gives us $\Phi(H[\emptyset]) - \Phi(H) = \pm 1 \mod p$. ■

There are two main examples for p-edge-transitive graphs. The first example is the class of the complete, bipartite graphs K_{p^l, p^m} with $l, m \geq 0$. The graph K_{p^l, p^m} has p^{l+m} edges and the automorphism group clearly acts transitively on the edges of that graph. The second example is the class of wreath graphs W_{p^k} for $k \geq 1$. The graph W_{p^k} has p^k vertices that can be decomposed in disjoint sets $V_0, \ldots, V_{p-1}$ of order p^{k-1} each, and edges $\{v_i, v_{i+1}\}$ for each $i = 0, \ldots, p-1$ and vertices $v_i \in V_i, v_{i+1} \in V_{i+1}$ (where it is understood that $V_p = V_0$). Thus in total, W_{p^k} has p^{2k-1} edges, except for $p = 2$ where it has 2^{2k-2} edges. The graph W_{p^k} can be seen as the lexicographical product of a p-cycle with a graph consisting of p^{k-1} disjoint vertices. For $k = 1$ we exactly obtain the p-cycle. To see that W_{p^k} is edge-transitive, we observe that on the one hand, for fixed i we can apply an arbitrary permutation on V_i leaving the graph invariant.

On the other hand, there exists a "rotational action" sending V_j to V_{j+1} for $j = 0, \dots, p-1$, which also leaves the graph invariant. Using these two types of automorphisms, we can map every edge to every other edge.

The following result tells us that in a certain sense the graphs K_{p^l,p^m} and W_{p^k} are the maximal p-edge-transitive graphs. A graph G is called *vertex-transitive* if its automorphism group $\mathsf{Aut}(G)$ acts transitively on its set of vertices $V(G)$.

Theorem 5.39. *Let G be a connected p-edge-transitive graph. Then either G is bipartite (and thus a subgraph of a graph of the form K_{p^l,p^m} for some $l, m \geq 0$) or G is vertex-transitive and an edge-subgraph of W_{p^k} for $k \geq 1$ (or both).*

For the proof of Theorem 5.39, we will make use of the following well-known result about the relation between edge and vertex-transitivity [12, Proposition 15.1].

Lemma 5.40. *Let G be a connected graph and let $\Gamma \subseteq \mathsf{Aut}(G)$ be a subgroup acting transitively on the set of edges $E(G)$. Then either Γ acts transitively on the set of vertices $V(G)$ (and thus G is vertex-transitive) or G is bipartite (or both).*

The proof from [12] carries over verbatim to the setting of the previous lemma, by replacing the full group $\mathsf{Aut}(G)$ with the subgroup Γ. We include it for completeness

Proof. Let $e_0 = \{v_1, v_2\} \in E(G)$ be some edge. Then for each vertex w_1 and some edge $\{w_1, w_2\}$ incident to w_1 there exists an automorphism $g \in \Gamma$ sending $\{v_1, v_2\}$ to $\{w_1, w_2\}$, thus either $w_1 = gv_1$ or $w_1 = gv_2$. This proves that the set $V(G)$ is the union of the orbits of v_1, v_2 under Γ. If these two orbits intersect, then in fact $\Gamma v_1 = \Gamma v_2$ (orbits under a group action are either disjoint or equal), and so $\Gamma v_1 = V(G)$ since the two sets above cover $V(G)$. On the other hand, if the two orbits are disjoints, then they form a partition making G into a bipartite graph. Indeed, any edge of G is in the orbit of the edge $\{v_1, v_2\}$ and thus connects an element of Γv_1 to an element of Γv_2. ■

Proof (of Theorem 5.39). Let G be a p-edge-transitive, non-bipartite graph. Then by Lemma 5.38 any p-Sylow subgroup $\Gamma \subseteq \mathsf{Aut}(G)$ still acts transitively on the edges $E(G)$ of G. By Lemma 5.40, since G is not bipartite, the group Γ acts transitively on the set of vertices $V(G)$ (and thus G is also vertex-transitive). We observe that in this case, by the Orbit-Stabilizer-Theorem (Theorem 2.29), we have $\#V(G) = p^k$ for some $k \geq 1$. We claim that then G is a subgraph of W_{p^k}.

To see this, let us reformulate our situation slightly: We identify the vertex set $V(G)$ with the set $[p^k] = \{1, \ldots, p^k\}$. Then we can canonically identify $\mathsf{Aut}(G)$ as a subgroup of S_{p^k}, the symmetric group on $[p^k]$ (this is because a graph automorphism is uniquely determined by its action on the vertices of a graph). Inside $\mathsf{Aut}(G)$ we have the subgroup Γ, which is a p-group. By the Sylow theorem, there exists a p-Sylow subgroup $\Gamma' \subseteq S_{p^k}$ containing Γ. Since the action of Γ is transitive on the set of edges $E(G)$, we can obtain $E(G)$ by starting with some edge $e_0 = \{v_1, v_2\} \in E(G)$ with $v_1, v_2 \in [p^k]$ and taking its orbit $\{\{gv_1, gv_2\} : g \in \Gamma\} = E(G)$. But note that by instead taking the orbit of e_0 under $\Gamma' \subseteq S_{p^k}$ we get at least this set of edges and maybe more. Denote by G' the graph with vertices $[p^k]$ and edges $\{\{gv_1, gv_2\} : g \in \Gamma'\}$. We claim that $G' \cong W_{p^k}$.

To show this we will explicitly identify the p-Sylow subgroup $\Gamma' \subseteq S_{p^k}$ (recall that by the Sylow theorem it is unique up to conjugation, that is reordering of the elements of $[p^k]$).

First note that S_{p^k} has $(p^k)!$ elements. Inductively one sees that the highest power of p appearing in this number is $p^{e(k)}$ for

$$e(k) = p^{k-1} + p^{k-2} + \ldots + p + 1\,.$$

We will inductively construct a subgroup $\Gamma(p, k)$ of S_{p^k} with $p^{e(k)}$ elements, which then is a p-Sylow subgroup. We note that a description of such a p-Sylow subgroup is given in [132].

For $k = 1$ we have $e(k) = 1$ and a p-Sylow subgroup $\Gamma(p, 1) \subseteq S_p$ is generated by a cyclic permutation $1 \mapsto 2, 2 \mapsto 3, \ldots, p \mapsto 1$ of the elements of $[p]$. The group $\Gamma(p, 1)$ is isomorphic to the cyclic group $\mathbb{Z}/p\mathbb{Z}$.

Now assume we constructed $\Gamma(p, k-1)$ for some $k \geq 2$, then we first note that a product of p copies $\prod_{i=0}^{p-1} \Gamma(p, k-1)$ of $\Gamma(p, k-1)$ acts on $[p^k]$ where the i-th factor acts by permutations on the elements

$$ip^{k-1} + 1, ip^{k-1} + 2, \ldots, ip^{k-1} + p^{k-1} = (i+1)p^{k-1}\,.$$

All of these actions commute, so we can see the product $\prod_{i=0}^{p-1} \Gamma(p, k-1)$ as a subgroup of S_{p^k}. However, there is a further action of $\mathbb{Z}/p\mathbb{Z}$ on $[p^k]$ sending j to $j + p^{k-1}$ (modulo p^k). This action cyclically permutes the p blocks of p^{k-1} elements in $[p^k]$ on which the p factors of $\prod_{i=0}^{p-1} \Gamma(p, k-1)$ act. Thus these two actions do not commute, but indeed they induce an action of the semidirect product

$$\Gamma(p, k) = \left(\prod_{i=0}^{p-1} \Gamma(p, k-1) \right) \rtimes \mathbb{Z}/p\mathbb{Z},$$

where $\mathbb{Z}/p\mathbb{Z}$ acts on $\prod_{i=0}^{p-1} \Gamma(p, k-1)$ by permuting the factors of the product.

We claim that $\Gamma(p,k)$ is the desired p-Sylow subgroup of S_{p^k}. Indeed, as a semidirect product its number of elements is

$$\#\Gamma(p,k) = (\#\Gamma(p,k-1))^p \cdot p = (p^{e(k-1)})^p \cdot p = p^{pe(k-1)+1} = p^{e(k)},$$

so it has the correct number of elements and is indeed a subgroup of S_{p^k}.

Now recall what we want to show: for a pair $\{v_1, v_2\}$ of vertices forming an edge of our original graph G, we want to show that the graph G' with edges $\{\{gv_1, gv_2\} : g \in \Gamma' \cong \Gamma(p,k)\}$ is isomorphic to the wreath graph W_{p^k}. By relabeling the vertices (that is performing a conjugation in S_{p^k}) we may assume that $\Gamma' = \Gamma(p,k)$. Furthermore, by a translation in the group $\Gamma(p,k)$, which acts transitively on the elements of $[p^k]$, we may assume that $v_1 = 1$. Now if v_2 were in the first block $[p^{k-1}]$ of vertices, on which the first factor $\Gamma(p,k-1)$ operates, then it is easy to see that the resulting graph G' would not be connected: the first factor $\Gamma(p,k-1)$ would send the edge $\{1, v_2\}$ only to edges within the first block $[p^{k-1}]$ and then the cyclic permutation by the factor $\mathbb{Z}/p\mathbb{Z}$ would send this pattern of edges to the $p-1$ other blocks, giving us a disjoint union of p graphs. This is not possible, since our original graph G is a subgraph of G' and also was assumed to be connected.

Thus we may assume that v_2 is in one of the other blocks

$$P(a) = [p^{k-1}] + ip^{k-1},$$

for $a = 1, \dots, p-1$. Now we want to argue that we can reorder these blocks, sending $P(a)$ to $P(1)$ and leaving $P(0)$ invariant, such that the group action of $\Gamma(p,k)$ is respected. And indeed, let $b \in \mathbb{Z}/p\mathbb{Z}$ be the multiplicative inverse of a (such that $ab = 1 \mod p$), then there is a permutation of $[p^k]$ sending the block $P(i)$ to $P(i \cdot b \mod p)$ (where the block is just translated as a whole, not permuting the elements inside). And indeed, we see that $P(a)$ is sent to $P(1)$. The reason why this permutation respects the form of the action[7] of $\Gamma(p,k)$ is that multiplication by b induces a group isomorphism $\mathbb{Z}/p\mathbb{Z} \to \mathbb{Z}/p\mathbb{Z}$ on the semidirect factor $\mathbb{Z}/p\mathbb{Z}$ of $\Gamma(p,k)$.

To summarize, we can assume without loss of generality that we start with an edge $\{1, v_2\}$ with v_2 in the second block of vertices. But then it is easy to see that the graph G' obtained by taking the orbit of $\{1, v_2\}$ under $\Gamma(p,k)$ is indeed the wreath graph W_{p^k}. Indeed, the group $\Gamma(p,k)$ acts transitively within each of the p blocks of vertices (since the i-th factor $\Gamma(p,k-1)$ above acts transitively there), so every edge from the first to the second block is in the orbit of $\{1, v_2\}$. Then finally the cyclic permutation action of $\mathbb{Z}/p\mathbb{Z}$ sends these edges to the set of all edges between blocks i and $i+1$, which exactly gives the set of edges of the wreath graph. This finishes the proof. $\blacksquare$

[7] To be precise, what happens is the following: the map sending $P(i)$ to $P(i \cdot b \mod p)$ is a permutation of $[p^k]$, that is an element $\sigma \in S_{p^k}$. What we are claiming is that the subgroup $\Gamma(p,k) \subseteq S_{p^k}$ is stable under the conjugation by σ, that is $\Gamma(p,k) = \sigma\Gamma(p,k)\sigma^{-1}$. So σ is a relabeling of the vertices of our graph G' which leaves the graph itself invariant.

From Homomorphisms to Induced Subgraphs

In what follows we will construct a sequence of reductions, starting from $\#\text{Hom}(\mathcal{H})$ and ending in $\#\text{IndSub}(\Phi)$. Here, $\mathcal{H}$ is a recursively enumerable set of p-edge-transitive graphs and Φ is a graph property such that for every graph $H \in \mathcal{H}$ we have that $\Phi(H[\emptyset]) \neq \Phi(H)$. For technical reasons, we cannot directly rely on the quantum graph $Q[\Phi, k]$.[8] For this reason, we will take a detour over color-prescribed homomorphisms and induced subgraphs as given by the following reduction sequence:

$$\begin{aligned} \#\text{Hom}(\mathcal{H}) &\leq^{\mathrm{T}}_{\mathrm{fpt}} \#\text{cp-Hom}(\mathcal{H}) && \text{Lemma 5.46} \\ &\leq^{\mathrm{T}}_{\mathrm{fpt}} \#\text{cp-IndSub}(\Phi) && \text{Lemma 5.45} \\ &\leq^{\mathrm{T}}_{\mathrm{fpt}} \#\text{IndSub}(\Phi) && \text{Lemma 5.47} \end{aligned} \tag{5.16}$$

Here, the definition of $\#\text{cp-IndSub}(\Phi)$ requires to introduce color-prescribed induced subgraphs. Recall that, given graphs G and H, we say that G is *H-colored* if G comes with a homomorphism c from G to H, called an *H-coloring*. Note that, in particular, every edge-subgraph of H can be H-colored by the identity function on $V(H)$, which is assumed to be the given coloring whenever we consider H-colored edge-subgraphs of H in this chapter. Given an edge-subgraph F of H and a homomorphism h from F to a H-colored graph G, we overload notation and say that h is *color-prescribed* if for all $v \in V(F) = V(H)$ it holds that $c(h(v)) = v$. We write $\mathsf{cp\text{-}Hom}(F \to G)$ for the set of all color-prescribed homomorphisms from F to G. $\mathsf{cp\text{-}StrEmb}(F \to G)$ is defined similarly for color-prescribed strong embeddings. We point out that a definition of $\mathsf{cp\text{-}Emb}(F \to G)$ is obsolete as every color-prescribed homomorphism is injective by definition and hence an embedding. Furthermore, we write $\mathsf{cp\text{-}Sub}(F \to G)$ and $\mathsf{cp\text{-}IndSub}(F \to G)$ for the sets of images of color-prescribed embeddings and strong embeddings from F to G, respectively. Elements of $\mathsf{cp\text{-}Sub}(F \to G)$ and $\mathsf{cp\text{-}IndSub}(F \to G)$ are referred to as color-prescribed subgraphs and induced subgraphs.[9] Given a graph property Φ and an H-colored graph G, we write $\mathsf{cp\text{-}IndSub}(\Phi \to G)$ for the set of all color-prescribed induced subgraphs of size $|V(H)|$ in G that satisfy Φ. Finally, we define $\#\text{cp-IndSub}(\Phi)$ to be the problem of, given a graph G that is H-colored for some graph H, computing $\#\mathsf{cp\text{-}IndSub}(\Phi \to G)$ and parameterize it by $\kappa(G) := |V(H)|$ — note that the H-coloring of G is part of the input and hence κ is well-defined.

[8]The transformation given by Theorem 5.11 relies in an intermediate step on the terms $\#\{H' \supseteq H\}$, i.e., the number of possibilities to add edges to H such that a graph isomorphic to H' is obtained. While $\#\{K_k \supseteq H\}$ is either 1 or 0, the case for arbitrary graphs H' is much less clear.

[9]The observant reader might have noticed that the sets $\mathsf{cp\text{-}Sub}(F \to G)$ and $\mathsf{cp\text{-}Hom}(F \to G)$ as well as $\mathsf{cp\text{-}IndSub}(F \to G)$ and $\mathsf{cp\text{-}StrEmb}(F \to G)$ are essentially the same as a color-prescribed homomorphism is uniquely identified by its image. However, we decided to distinguish those notions in order to make the subsequent combinatorial arguments more accessible.

The proofs of Lemma 5.46 and Lemma 5.47 in Reduction Sequence 5.16 are straightforward. For this reason, we will begin with the intermediate step (Lemma 5.45). The reduction from color-prescribed homomorphisms to color-prescribed induced subgraphs will make implicit use of an H-colored variant of quantum graphs. More precisely, given an H-colored graph G and a graph property Φ, we will express $\#\mathsf{cp}\text{-}\mathsf{IndSub}(\Phi \to G)$ as a linear combination of color-prescribed homomorphisms, that is, terms of the form $\#\mathsf{cp}\text{-}\mathsf{Hom}(H[S] \to G)$. In a first step, we show complexity monotonicity for linear combinations of color-prescribed homomorphisms. While this property allows a quite simple proof, a second step, in which we study the coefficient of $\#\mathsf{cp}\text{-}\mathsf{Hom}(H \to G)$ requires a thorough understanding of the alternating enumerator of Φ and H. In case of p-edge-transitive graphs, the latter is provided by Lemma 5.36.

We start by introducing a colored variant of the tensor product of graphs (see e.g. Chapter 5.4.2 in [95]). Given two H-colored graphs G and $\hat{G}$ with colorings c and $\hat{c}$ we define their *color-prescribed* tensor product $G \times_H \hat{G}$ as the graph with vertices $V = \{(v, \hat{v}) \in V(G) \times V(\hat{G}) \mid c(v) = \hat{c}(\hat{v})\}$ and edges between two vertices $(v, \hat{v})$ and $(u, \hat{u})$ if and only if $\{v, u\} \in E(G)$ and $\{\hat{v}, \hat{u}\} \in E(\hat{G})$. The next lemma states that $\#\mathsf{cp}\text{-}\mathsf{Hom}(F \to \star)$ is linear with respect to $\times_H$.

Lemma 5.41. *Let H be a graph, let F be an edge-subgraph of H and let G and $\hat{G}$ be H-colored. Then we have that*

$$\#\mathsf{cp}\text{-}\mathsf{Hom}(F \to G \times_H \hat{G}) = \#\mathsf{cp}\text{-}\mathsf{Hom}(F \to G) \cdot \#\mathsf{cp}\text{-}\mathsf{Hom}(F \to \hat{G}) .$$

Proof. It can easily be verified that the function $b(h, \hat{h})(v) := (h(v), \hat{h}(v))$ that assigns elements in $\mathsf{cp}\text{-}\mathsf{Hom}(F \to G) \times \mathsf{cp}\text{-}\mathsf{Hom}(F \to \hat{G})$ to elements in $\mathsf{cp}\text{-}\mathsf{Hom}(F \to G \times_H \hat{G})$ is a well-defined bijection. ■

We are now prepared to prove the color-prescribed variant of complexity monotonicity.

Lemma 5.42 (Complexity monotonicity). *Let H be a graph and let a be a function from edge-subgraphs of H to rationals. There exists an algorithm $\mathbb{A}$ that is given an H-colored graph G as input and has oracle access to the function*

$$\sum_{S \subseteq E(H)} a(H[S]) \cdot \#\mathsf{cp}\text{-}\mathsf{Hom}(H[S] \to \star) ,$$

and computes $\#\mathsf{cp}\text{-}\mathsf{Hom}(H[S] \to G)$ for all S such that $a(H[S]) \neq 0$ in time $f(|H|) \cdot |V(G)|$ where f is a computable function. Furthermore, every oracle query $\hat{G}$ satisfies $|V(\hat{G})| \leq f(|H|) \cdot |V(G)|$.

Proof. Using Lemma 5.41 we have that for every H-colored graph F it holds that

$$\sum_{S\subseteq E(H)} a(H[S]) \cdot \#\mathsf{cp\text{-}Hom}(H[S] \to (G \times_H F)) \tag{5.17}$$

$$= \sum_{S\subseteq E(H)} a(H[S]) \cdot \#\mathsf{cp\text{-}Hom}(H[S] \to G) \cdot \#\mathsf{cp\text{-}Hom}(H[S] \to F)\,, \tag{5.18}$$

which we can evaluate for $F = H[\emptyset], \dots, H[E(H)]$. This induces a system of linear equations and the corresponding matrix is proved to be non-singular in Appendix B (see Lemma B.2). Consequently, the numbers

$$a(H[S]) \cdot \#\mathsf{cp\text{-}Hom}(H[S] \to G)$$

are uniquely determined and can be computed by solving the system using Gaussian elimination. Finally, we obtain the numbers $\#\mathsf{cp\text{-}Hom}(H[S] \to G)$ by multiplying with $a(H[S])^{-1}$ whenever $a(H[S]) \neq 0$. ■

It remains to express the number of color-prescribed induced subgraphs that satisfy a property Φ as a linear combination of color-prescribed homomorphisms.

Lemma 5.43. *Let H be a graph, let Φ be a graph property and let G be an H-colored graph. Then we have that*

$$\#\mathsf{cp\text{-}IndSub}(\Phi \to G) = \sum_{S\in E(H)} \Phi(H[S]) \sum_{J\subseteq E(H)\setminus S} (-1)^{\#J} \cdot \#\mathsf{cp\text{-}Hom}([H[S \cup J] \to G)\,.$$

Moreover, the absolute values of the coefficient of $\#\mathsf{cp\text{-}Hom}(H \to G)$ and the alternating enumerator $\hat{\chi}(\Phi, H)$ are equal.

Proof. We start by establishing the following claim.

Claim 5.44. *Let H be graph, let $S \subseteq E(H)$ and let G be an H-colored graph. Then we have that*

$$\#\mathsf{cp\text{-}IndSub}(H[S] \to G) = \sum_{J\subseteq E(H)\setminus S} (-1)^{\#J} \cdot \#\mathsf{cp\text{-}Sub}(H[S \cup J] \to G)\,.$$

Proof. We have that

$$\mathsf{cp\text{-}IndSub}(H[S] \to G) = \mathsf{cp\text{-}Sub}(H[S] \to G) \setminus \bigcup_{e\in E(H)\setminus S} \mathsf{cp\text{-}Sub}(H[S \cup \{e\}] \to G)\,.$$

Hence, by inclusion-exclusion (Theorem 2.27),

$$\#\mathsf{cp\text{-}IndSub}(H[S] \to G) = \sum_{J\subseteq E(H)\setminus S} (-1)^{\#J} \cdot \#\mathsf{cp\text{-}Sub}(H[S \cup J] \to G)\,. \quad \square$$

Now we have that

$$\#\mathsf{cp\text{-}IndSub}(\Phi \to G) = \sum_{S \in E(H)} \Phi(H[S]) \cdot \#\mathsf{cp\text{-}IndSub}(H[S] \to G) \tag{5.19}$$

$$= \sum_{S \in E(H)} \Phi(H[S]) \sum_{J \subseteq E(H) \setminus S} (-1)^{\#J} \cdot \#\mathsf{cp\text{-}Sub}(H[S \cup J] \to G) \tag{5.20}$$

$$= \sum_{S \in E(H)} \Phi(H[S]) \sum_{J \subseteq E(H) \setminus S} (-1)^{\#J} \cdot \#\mathsf{cp\text{-}Hom}(H[S \cup J] \to G) \tag{5.21}$$

where (5.19) follows from the definition of $\mathsf{cp\text{-}IndSub}(\Phi \to G)$, (5.20) is Claim 5.44 and (5.21) holds as color-prescribed homomorphisms are injective and a color-prescribed embedding is uniquely identified by its image. Collecting for the coefficient of $\#\mathsf{cp\text{-}Hom}(H \to G)$ yields

$$\sum_{S \in E(H)} \Phi(H[S]) \cdot (-1)^{\#E(H) - \#S} = (-1)^{\#E(H)} \cdot \hat{\chi}(\Phi, H)\,. \tag{5.22}$$

■

The application of complexity monotonicity for color-prescribed homomorphisms (Lemma 5.42) requires non-zero coefficients. However, this can be guaranteed for the coefficient of interest in case of p-edge-transitive graphs as shown in Lemma 5.36. Formally, the reduction is constructed as follows.

Lemma 5.45. *Let Φ be a graph property and let H be a p-edge-transitive graph such that $\Phi(H[\emptyset]) \neq \Phi(H)$. There exists an algorithm $\mathbb{A}$ that is given an H-colored graph G as input and has oracle access to the function $\#\mathsf{cp\text{-}IndSub}(\Phi \to \star)$ and computes $\#\mathsf{cp\text{-}Hom}(H \to G)$ in time $f(|H|) \cdot |V(G)|$ where f is a computable function. Furthermore, every oracle query $\hat{G}$ is H-colored as well and satisfies $|V(\hat{G})| \leq f(|H|) \cdot |V(G)|$.*

Proof. Using Lemma 5.43 we can express $\#\mathsf{cp\text{-}IndSub}(\Phi \to \star)$ as a linear combination of color-prescribed homomorphisms. In particular, the coefficient of $\#\mathsf{cp\text{-}Hom}(H \to \star)$ is $(\pm 1) \cdot \hat{\chi}(\Phi, H)$ and by Lemma 5.36 we have that this number is non-zero whenever H is p-edge-transitive and $\Phi(H[\emptyset]) \neq \Phi(H)$. Hence we can use the algorithm from Lemma 5.42 to compute $\#\mathsf{cp\text{-}Hom}(H \to G)$ in the desired running time. ■

Proofs of Lemma 5.46 and Lemma 5.47

In the first reduction we are given graphs H and G and the goal is to compute $\#\mathsf{Hom}(H \to G)$ using an oracle for $\#\mathsf{cp\text{-}Hom}(H \to \star)$. This can be done by taking precisely $|V(H)|$ copies of the vertices of G, that is, one for each vertex in H and then adding an edge between two vertices u and v if they have been adjacent in G and the vertices of H corresponding to the copies of $V(G)$ that contain u and v are adjacent in H as well. The construction is formalized in the proof of the following lemma. In particular it is shown that the resulting graph $\hat{G}$ is H-colored.

Lemma 5.46. *Let H be a graph. There exists an algorithm $\mathbb{A}$ that is given a graph G as input and has oracle access to the function $\#\textsf{cp-Hom}(H \to \star)$ and computes $\#\textsf{Hom}(H \to G)$ in time $f(|V(H)|) \cdot |V(G)|$ where f is a computable function. Furthermore, every oracle query $\hat{G}$ satisfies that $|V(\hat{G})| \leq f(|V(H)|) \cdot |V(G)|$.*

Proof. Let $k = |V(H)|$. It will be convenient to assume that $V(H) = [k]$. Given G, we construct a graph $\hat{G}$ as follows. The vertex set of $\hat{G}$ is defined to be

$$V(\hat{G}) = \bigcup_{i=1}^{k} V_i\,,$$

where $V_i = \{v_i \mid v \in V(G)\}$ is a copy of $V(G)$ identified with $i \in V(H)$. We add an edge $\{u_i, v_j\}$ to $\hat{G}$ if and only if $\{i,j\} \in E(H)$ and $\{u,v\} \in E(G)$. Now it can easily be verified that the function $c : V(\hat{G}) \to V(H)$ given by $c(v_i) := i$ is an H-coloring of $\hat{G}$. Furthermore it is easy to see that $\#\textsf{cp-Hom}(H \to \hat{G}) = \#\textsf{Hom}(H \to G)$. ■

The last part of the reduction sequence allows us to get rid of the colors. More precisely, we will reduce the problem of counting color-prescribed induced subgraphs of an H-colored graph to the problem of counting uncolored induced subgraphs of size $|V(H)|$ in a graph, both with respect to some property Φ. The proof is a straightforward application of the inclusion-exclusion principle (Theorem 2.27).

Lemma 5.47. *Let Φ be a graph property and let H be a graph with k vertices. There exists an algorithm $\mathbb{A}$ that is given an H-colored graph G as input and has oracle access to the function $\#\textsf{IndSub}(\Phi, k \to \star)$ and computes $\#\textsf{cp-IndSub}(\Phi \to G)$ in time $f(k) \cdot |V(G)|$ where f is a computable function. Furthermore, every oracle query $\hat{G}$ satisfies $|V(\hat{G})| \leq |V(G)|$ and, in particular, $\hat{G}$ allows an H-coloring as well.*

Proof. It will be convenient to assume that $V(H) = [k]$. We first check whether the H-coloring c of G is surjective. If this is not the case then there exists some vertex $i \in V(H)$ such that $i \notin \textsf{im}(c)$ and hence there is no color-prescribed induced subgraph of G, so $\mathbb{A}$ can just output 0. Otherwise, the H-coloring of G induces a partition of $V(G)$ in k many non-empty and pairwise disjoint subsets, each associated with some "color" $i \in V(H)$. This allows us to equivalently express $\textsf{cp-IndSub}(\Phi \to G)$ in terms of vertex-colorful induced subgraphs:

$$\textsf{cp-IndSub}(\Phi \to G) = \left\{ S \subseteq \binom{V(G)}{k} \;\middle|\; c(S) = [k] \,\wedge\, \Phi(G[S]) = 1 \right\}$$

By the principle of inclusion and exclusion (Theorem 2.27) we obtain that

$$\#\mathsf{cp\text{-}IndSub}(\Phi \to G) = \sum_{J \subseteq [k]} (-1)^{\#J} \cdot \#\mathsf{IndSub}(\Phi, k \to G_J)\,,$$

where G_J is the graph obtained from G by deleting all vertices that are colored with some color in J. Hence we can compute $\#\mathsf{cp\text{-}IndSub}(\Phi \to G)$ using 2^k oracle calls. Finally, we observe that H-colored graphs are closed under the removal of vertices and therefore every G_J allows an H-coloring. ■

Non-trivial bipartite graph properties

Finally, we apply the algebraic approach which was laid out in the preceding sections to bipartite graph properties. This will allow us to prove our main result. Recall that a set $\mathcal{K} \subseteq \mathbb{N}$ is called *dense* if there exists a constant c such that for every $k' \in \mathbb{N}$ there exists $k \in \mathcal{K}$ such that $k' \leq k \leq ck'$. Furthermore, we write IS_k for the graph with k isolated vertices. The following theorem is obtained by invoking the reduction sequence (5.16) to complete bipartite graphs $K_{t,t}$ for prime powers $t = p^k$, which are p-edge-transitive (see Section 5.2.1).

Theorem 5.48 (Theorem 5.31 restated). *Let Φ be a computable graph property and let $\mathcal{K}$ be the set of all prime powers t such that $\Phi(\mathsf{IS}_{2t}) \neq \Phi(K_{t,t})$. If $\mathcal{K}$ is infinite then $\#\textsc{IndSub}(\Phi)$ is $\#\mathsf{W}[1]$ hard. If additionally $\mathcal{K}$ is dense then it cannot be solved in time $f(k) \cdot n^{o(k)}$ for any computable function f unless ETH fails. This holds true even if the input graphs to $\#\textsc{IndSub}(\Phi)$ are restricted to be bipartite.*

The proof of the previous theorem requires the following intermediate lemma. In particular, we use a trick inspired by Lemma 1.11 in [39] to make the reduction parsimonious which is required for the extension in Appendix B.

Lemma 5.49. *There exists an algorithm that, given an integer $\ell > 1$ and a graph G with n vertices, computes in time $O(\ell n)$ a $K_{\ell,\ell}$-colored graph G' with at most $O(\ell n)$ vertices such that the number of cliques of size ℓ in G equals $\#\mathsf{cp\text{-}Hom}(K_{\ell,\ell} \to G')$.*

Proof. Let the vertex set of G be $\{v_i \mid 1 \leq i \leq n\}$ and let that of $K_{\ell,\ell}$ be $\{a_i, b_i \mid 1 \leq i \leq \ell\}$. We now construct the graph G' on the vertex set $\{u_{i,j}, w_{i,j} \mid 1 \leq i \leq \ell, 1 \leq j \leq n\}$ with a $K_{\ell,\ell}$-coloring given by $c(u_{i,j}) = a_i$ and $c(w_{i,j}) = b_i$. We add an edge between $u_{i,j}$ and $w_{i',j'}$ if and only if

- either $(i,j) = (i',j')$,
- or $i < i'$, $j < j'$ and the vertices v_j and $v_{j'}$ are adjacent,
- or $i > i'$, $j > j'$ and the vertices v_j and $v_{j'}$ are adjacent.

Let $\{v_{i_1}, \dots v_{i_\ell}\}$ be an ℓ-clique in G. Assume w.l.o.g. that $i_k < i_{k'}$ for $k < k'$. Then the set $\{u_{1,j_1}, \dots u_{\ell,j_\ell}, w_{1,i_1}, \dots w_{\ell,j_\ell}\}$ forms a colorful biclique in G', so it gives rise to a color-prescribed homomorphism $h \in \mathsf{cp\text{-}Hom}(K_{\ell,\ell} \to G')$. Now let $h' \in \mathsf{cp\text{-}Hom}(K_{\ell,\ell} \to G')$ be a color-prescribed homomorphism. Then there has to be the following colorful biclique in G':

$$\{u_{1,\alpha_1}, \dots u_{\ell,\alpha_\ell}, w_{1,\beta_1}, \dots w_{\ell,\beta_\ell}\}\,.$$

We first see that for every i we have $\alpha_i = \beta_i$ since there has to be an edge between u_{i,α_i} and w_{i,β_i}. Furthermore the edges enforce $\alpha_j < \beta_{j'}$ for every $j < j'$. Thus $\{v_{\alpha_1}, \dots, v_{\alpha_\ell}\}$ is a clique of size ℓ in G. Since every homomorphism yields $\beta_j = \alpha_j < \beta_{j'} = \alpha_{j'}$ for $j < j'$ there is exactly one homomorphism in $\mathsf{cp\text{-}Hom}(K_{\ell,\ell} \to G')$ corresponding to each ℓ-clique in G.■

Proof (of Theorem 5.48). Let Φ and $\mathcal{K}$ be as given in Theorem 5.48. We define a class of graphs $\mathcal{H}$ as follows:

$$\mathcal{H} = \{K_{t,t} \mid t \in \mathcal{K}\}\,.$$

By Reductions Sequence (5.16), given by Lemma 5.46, Lemma 5.45 and Lemma 5.47, we obtain that $\#\text{Hom}(\mathcal{H}) \leq^{\mathrm{T}}_{\mathrm{fpt}} \#\text{IndSub}(\Phi)$. As Φ is computable, $\mathcal{H}$ is recursively enumerable. Furthermore, as $\mathcal{K}$ is infinite, we have that there are arbitrary large bicliques in $\mathcal{H}$ and, in particular, the treewidth of $\mathcal{H}$ is unbounded. Therefore $\#\text{Hom}(\mathcal{H})$, and hence $\#\text{IndSub}(\Phi)$, are $\#\mathsf{W}[1]$-hard by Theorem 2.34. For the tight bound under ETH, we reduce from the *decision problem* Clique which asks, given G and k, to *decide* whether G contains a clique of size k and which cannot be solved in time $f(k) \cdot n^{o(k)}$ for any computable function f, unless ETH fails [33, 34]. Now assume that $\mathcal{K}$ is dense and let (G, k) be an instance of Clique. By density of $\mathcal{K}$, there exists $\ell \in \mathcal{K}$ such that $k \leq \ell \leq ck$ for some overall constant c independent of k. We construct the graph $\hat{G}$ from G by adding $\ell - k$ further vertices and adding edges between all new vertices as well as between every pair of an old and a new vertex. It can then easily be verified that G contains a clique of size k if and only if $\hat{G}$ contains a clique of size ℓ.

Next we apply Lemma 5.49 to $\hat{G}$ and ℓ, and obtain an $K_{\ell,\ell}$-colored graph G' satisfying that the number of ℓ-cliques in $\hat{G}$ is equal to the number $\#\mathsf{cp\text{-}Hom}(K_{\ell,\ell} \to G')$. Finally, we invoke Lemma 5.45 and Lemma 5.47 to conclude the reduction. In particular, all reductions are tight in the sense that every oracle call for $\#\text{IndSub}(\Phi)$ in the final part of the reduction is a pair $(\tilde{G}, 2\ell)$ where the number of vertices of $\tilde{G}$ is bounded by $O(\ell \cdot |V(G)|)$. As $\ell \leq ck$ we conclude that every algorithm that solves $\#\text{IndSub}(\Phi)$ in time $f(k) \cdot n^{o(k)}$ can be used to solve Clique in time $f(k) \cdot n^{o(k)}$ — just check in the end whether the output is a number greater than zero.

Finally, we point out that for both ($\#\mathsf{W}[1]$ and ETH) hardness results, the last part of the reduction, that is, Lemma 5.47 only queries for graphs that are $K_{t,t}$-colorable and hence bipartite. ■

Note that, in case Φ or its complement is edge-monotone, we only have to find infinitely many prime powers t for which Φ is neither true nor false on the set of all edge-subgraphs of $K_{t,t}$, which is the case for all sensible, non-trivial properties that do not rely on the number of vertices in some way. If Φ (or its complement) is monotone, that is, not only closed under the removal of edges, but also under the removal of vertices, then such artificial properties do not exist and we can state the result more clearly as follows.

Corollary 5.50 (Theorem 5.33 restated). *Let Φ be a computable monotone graph property such that Φ and $\neg\Phi$ hold on infinitely many bipartite graphs. Then $\#\textsc{IndSub}(\Phi)$ is $\#\mathsf{W}[1]$-hard and cannot be solved in time $f(k) \cdot n^{o(k)}$ for any computable function f unless ETH fails. This holds true even if the input graphs to $\#\textsc{IndSub}(\Phi)$ are restricted to be bipartite.*

Proof. If Φ is monotone and Φ and $\neg\Phi$ hold on infinitely many bipartite graphs, then $\Phi(\mathsf{IS}_k) = 1$ for all positive integers k and $\Phi(K_{t,t}) = 0$ for all but finitely many t. Hence we can apply Theorem 5.48 and, in particular, the set $\mathcal{K}$ will contain all but finitely many prime powers and is therefore dense. ∎

Chapter 6

Existential First-order Formulas

The last chapter of this thesis is devoted to the study of the parameterized and fine-grained complexity of the problem of counting answers to existential (and universal) first-order formulas. Very roughly speaking, this family of counting problems generalizes problems that have been studied prior in this thesis by one additional quantifier alternation. It will turn out that counting answers to conjunctive queries, which constitute the most basic class of first-order formulas (see Chapter 2.6), takes the role of the homomorphism counting problem in the setting of the current chapter. In particular, building upon prior work of Chen, Durand and Mengel [58, 31, 32], we will provide an extensive classification result for the parameterized problem of counting answers to conjunctive queries that subsumes the classification for counting homomorphisms (Theorem 2.34) as a special case. After that, we will adapt complexity monotonicity to conjunctive queries by introducing what we call *quantum queries*, which constitute a natural generalization of quantum graphs to existential first-order formulas. This will then allow us to lift the classification for counting answers to conjunctive queries to general existential first-order formulas.

One major difference to the previous chapters is the fact that we will encounter problems that are *not* #W[1]-easy, and not even #W[2]-easy under standard assumptions. Instead, we need to consider the #A-classes to fully understand the parameterized complexity of counting answers to existential first-order formulas. The reader is encouraged to recall the introduction to first-order logic in Chapter 2.6; in what follows, we assume familiarity with the notions introduced there.

The results of this chapter have been obtained in collaboration with Holger Dell and Philip Wellnitz and are published in [48]; a full version can be found in [49].

6.1 The #A-Hierarchy

We begin with the necessary background on the #A-classes. A detailed exposition can be found in [65, Chapt. 14]. Similarly to #W[1] and #W[2], we use complete problems for the definitions.

Definition 6.1. Let $t > 0$ be a fixed positive integer. A parameterized counting problem (P, κ) is called

- #A[t]-*hard* if $\#\text{P-MC}(\Pi_{t-1}) \leq^{\mathrm{T}}_{\mathrm{fpt}} (P, \kappa)$,
- #A[t]-*easy* if $(P, \kappa) \leq^{\mathrm{T}}_{\mathrm{fpt}} \#\text{P-MC}(\Pi_{t-1})$, and
- #A[t]-*equivalent* if it is both, #A[t]-hard and #A[t]-easy.

Flum and Grohe [65, Chapt. 14.2] proved that #W[1]-easy $\Leftrightarrow$ #A[1]-easy, and

$$\#\mathsf{W}[1]\text{-easy} \Rightarrow \#\mathsf{W}[2]\text{-easy} \Rightarrow \#\mathsf{A}[2]\text{-easy}.$$

However, it is conjectured that the backward directions do not hold. In particular, this implies that the sets of #W[1]-equivalent problems, #W[2]-equivalent problems and #A[2]-equivalent problems are pairwise different. By definition, counting answers to existential first-order formulas is #A[2]-easy. We complement this result by proving that the problem is #A[2]-hard, even when restricted to conjunctive queries over the signature of graphs, excluding those with self-loops. This requires us to lift a technical normalization result from decision to counting; the proof can be found in Appendix C.

6.2 Conjunctive Queries

Conjunctive query evaluation is a core problem in database theory. Recall from Chapter 2.6 that conjunctive queries can be expressed by formulas of the form

$$x_1 \dots x_k \exists y_1 \dots \exists y_\ell (a_1 \wedge \cdots \wedge a_m)\,, \tag{6.1}$$

where the x_i are the *free variables*, the y_i are the (existentially) *quantified variables*, and the a_i are *atomic formulas* (such as edge $E(x_1, y_4)$ or relational $R(x_7, y_3, y_6)$ constraints on the variables). Conjunctive queries exactly correspond to select-project-join queries; a detailed introduction can be found in the textbook of Abiteboul, Hull, and Vianu [4].

Perhaps the most naïve way to study the complexity of counting answers to conjunctive queries is via its *combined complexity*, in which both the query and the structure are considered to be worst-case inputs. Since conjunctive queries generalize the clique problem on graphs, the problem is clearly NP-hard in this setting [28]. In the real world, however, the structure models a relational database and is much larger than the query, and thus the combined complexity may fixate on instances that we do not care about. For this reason the framework of parameterization fits best to study the complexity of the problem. Furthermore, we will consider its *data complexity* which allows for a fine-grained complexity analysis. In particular, the data complexity considers the query to be completely fixed and only the structure to be worst-case input. If the query is fixed, the number of variables $k + \ell$ is a constant, and so the problem is polynomial-time solvable: even the exhaustive search algorithm just needs to try out and check all $n^{k+\ell}$ possible assignments to the variables, where n is the size of the universe. Unsurprisingly, exhaustive search is not the best strategy for every query. For example, Chekuri and Rajaraman [30] showed that the decision and counting problems can be solved in time $O(n^{t+1})$ where t is the treewidth of the query's Gaifman graph, that is, the graph containing a vertex for every variable and an edge between two vertices whenever the corresponding variables are contained in a common constraint. Their algorithm is a natural generalization of the treewidth-based dynamic programming algorithm for counting homomorphisms (see Theorem 2.35). Since $t + 1$ is typically much smaller than $k + \ell$, this algorithm is better than exhaustive search. For each fixed query Q, the guiding question for a fine-grained understanding of data complexity is this: What is the smallest constant c_Q such that the query evaluation problem can be solved in time $O(n^{c_Q})$?

Now consider a conjunctive query as given in (6.1) and observe that instances with $k = 0$ constitute decision problems. For this reason, it is not surprising that there are families Ψ of conjunctive queries for which $\#\text{P-MC}(\Psi)$ is equivalent to a decision problem. It is hence necessary to consider the decision version of $\#\mathsf{W}[1]$ to fully understand the complexity of counting answers to conjunctive queries. To this end, we let Clique be the problem of, given a graph G and a positive integer k, computing 1 if G contains a clique of size k as a subgraph and 0 otherwise; the parameterization is given by k.

Definition 6.2. A parameterized counting problem (P, κ) is called $\mathsf{W}[1]$-*equivalent* if it is interreducible with Clique with respect to parameterized Turing reductions.

We build upon the work of Chen, Durand and Mengel [58, 31] who established the following classification.

Theorem 6.3. *Let Ψ be a class of conjunctive queries of bounded arity.*[1] *Then $\#\text{P-MC}(\Psi)$ is either polynomial-time solvable, or $\mathsf{W}[1]$-equivalent, or $\#\mathsf{W}[1]$-hard.*

[1]Unbounded arity is not meaningful as the size of a structure might in this case not be bounded by a polynomial in the number of vertices. We address this issue formally in Appendix C.

The criteria for the above theorem are given explicitly; we will provide an exposition in the subsequent section. Furthermore, Chen and Mengel [32] extended the previous theorem to the following more general case.

Theorem 6.4. *Let Σ be a class of existential first-order formulas of bounded arity without negations. Then $\#\text{P-MC}(\Sigma)$ is either fixed-parameter tractable, or $\mathsf{W}[1]$-equivalent, or $\#\mathsf{W}[1]$-hard.*

In proving the previous theorem, Chen and Mengel implicitly used complexity monotonicity. In contrast to Theorem 6.3, they did not provide explicit criteria for the three different cases.

Note that neither of Theorem 6.3 and Theorem 6.4 does fully reveal the complexity of $\#\text{P-MC}(\Psi)$ and $\#\text{P-MC}(\Sigma)$, respectively, as the general problem of counting answers to conjunctive queries is $\#\mathsf{A}[2]$-equivalent.

As already indicated before, we make simultaneous progress on two fronts: Our complexity classifications are finer than previous work, and we can prove the classification for more general classes of queries. An important feature of our work is that the proofs are modular and largely self-contained: We first prove the complexity results for counting partial homomorphisms[2], then lift them to conjunctive queries, and then further to a more general class of queries. So what is the most general class of queries that we study? We allow queries φ of the form

$$x_1 \dots x_k \exists y_1 \dots \exists y_\ell : \psi \,, \tag{6.2}$$

where ψ is a quantifier-free formula in first-order logic and all negations in ψ must be directly applied to constraints that only involve free variables (e.g. $E(x_1, x_7) \vee (R(x_7, y_7, y_9) \wedge \neg R(x_1, x_4, x_9)))$. Constraints of the form $\neg R(x_1, x_4, x_9)$ are referred to as *non-monotone constraints* in the remainder of the paper. Furthermore φ may be equipped with a set of inequalities over the free variables (eg. $x_3 \neq x_5$), the semantics of which are formally defined in the respective section.

All of our theorems also apply to the corresponding *universal* queries, where each $\exists$ in (6.2) is replaced with $\forall$, but for the sake of readability we will often omit this fact.

To study the data complexity of the problem, we employ the Strong Exponential Time Hypothesis. The problem #DOMSET can be easily expressed as a (universal) conjunctive query, and recall that this problem cannot be solved in time $O(n^{k-\varepsilon})$ unless SETH is false (see Theorem 2.21). We are able to lift this hardness result to all queries φ that have the k-dominating set query as a *query minor*, a notion that we translate from graphs and formalize later. The *dominating star size* $\mathsf{dss}(\varphi)$ of a conjunctive query φ is the maximum number k such that the k-dominating set query is a query minor.

[2]Not to be confused with partially injective homomorphisms.

Equivalently, this means that some connected component in the quantified variables of φ has k neighbors in the free variables.[3] We obtain the following result:

Theorem 6.5 (Data complexity). *Let φ be a fixed query of the form* (6.2) *such that* $\mathsf{dss}(\varphi) \geq \max\{3, \mathsf{arity}(\varphi)\}$*. Given a structure* G *with n vertices, we wish to compute* $\#\varphi(\mathrm{G})$*. This problem cannot be solved in time*

$$O(n^{\mathsf{dss}(\varphi)-\varepsilon})$$

for any $\varepsilon > 0$, unless SETH fails.

Neglecting many technical details, the proof of Theorem 6.5 reduces #DomSet to the model counting problem for φ by following operations of the query minor. If φ is a query of the form (6.2), then it can be represented by a quantum query with constituents φ'; in this case, we define $\mathsf{dss}(\varphi)$ as the maximum $\mathsf{dss}(\varphi')$ over all φ'. We formalize those notions in Chapter 6.3.

Theorem 6.5 is similar in spirit to other known conditional lower bounds for first-order model checking, such as the one of Williams [136] and Gao et al. [68]. One of their results is that first-order sentences with $k+1$ variables cannot be decided in time $O(m^{k-\varepsilon})$, where m is the size of the structure, unless SETH fails. However, these results are incomparable to Theorem 6.5 for several reasons: The results in [136, 68] allow negations and consider the decision problem, while we allow only limited negations and consider the counting problem. More fundamentally, however, Theorem 6.5 gives a hardness result for every fixed query φ, while the results in [136, 68] show that there exists a query φ that is hard. Moreover, the lower bounds in [136, 68] are in terms of the size m of the structure, not merely the size n of the domain.

Regarding the parameterized complexity of counting answers to conjunctive queries, we show that the dominating star size, i.e., the parameter considered in Theorem 6.5, is a structural parameter for conjunctive queries that, if unbounded, makes the problem #$\mathsf{W}[2]$-hard and that, if bounded, keeps the problem #$\mathsf{W}[1]$-easy.

Furthermore, we study which families of conjunctive queries constitute the hardest parameterized counting problems, even harder than the #$\mathsf{W}[2]$-hard cases unless #$\mathsf{A}[2]$ = #$\mathsf{W}[2]$ holds, which seems unlikely.[4] We prove that families of conjunctive queries are #$\mathsf{A}[2]$-hard if their *linked matching number* is unbounded. Intuitively a conjunctive query φ with free variables X and quantified variables Y has a large linked matching if there is a large well-linked set in Y that cannot be separated from X by removing a small number of variables. It is formally defined in Chapter 6.3. We obtain the following refined complexity classification.

[3]The dominating star size coincides with the *strict star size* from [31].

[4]See Chapt. 8 and 14 in [65] for a discussion.

Theorem 6.6 (Parameterized complexity). *Let Φ be a family of conjunctive queries of bounded arity. The problem* $\#\textsc{p}\text{-MC}(\Phi)$ *is*

1. $\#\mathsf{W}[1]$*-easy if the dominating star size of* Φ *is bounded,*
2. $\#\mathsf{W}[2]$*-hard if the dominating star size of* Φ *is unbounded, and*
3. $\#\mathsf{A}[2]$*-equivalent if the linked matching number of* Φ *is unbounded.*

It is instructive to provide examples for the application of the above theorem. First consider the problem of, given a graph G and a positive integer k, computing the number of cliques of size k that are not maximal. While the problem of counting cliques of size k is #W[1]-equivalent, adding the non-maximality constraint makes the problem hard for #W[2]. To see this, we will express the problem as a conjunctive query

$$\varphi_k := x_1 \dots x_k \exists y : \bigwedge_{1 \le i < j \le k} E(x_i, x_j) \wedge \bigwedge_{1 \le i \le k} E(x_i, y)\,. \tag{6.3}$$

Note that the number of solutions to φ_k in G is precisely $k!$ times the number of non-maximal cliques of size k in G. Furthermore, it holds that φ_k has dominating star size k and hence that $\Phi = \{\varphi_k \mid k \in \mathbb{N}\}$ has unbounded dominating star size. By Theorem 6.6 the problem of counting answers to queries in Φ is #W[2]-hard. Furthermore, invoking Theorem 6.5, we obtain that counting non-maximal cliques of size k cannot be done in time $O(n^{k-\varepsilon})$ for any $\varepsilon > 0$. Note that this is also in sharp contrast to the problem of counting (not necessarily non-maximal) cliques of size k which can be done in time $O(n^{\omega k/3})$ (see Theorem 2.20). Furthermore deciding the existence of a non-maximal clique of size k is equivalent to deciding the existence of a clique of size $k+1$ and hence the lower bound under SETH crucially depends on the fact that we count the solutions.

On the other hand, counting non-maximal cliques of size k is most likely not #A[2]-hard as it is #W[2]-easy[5]. An example for a #A[2]-hard problem would be the following. Assume a graph G and a positive integer k are given. Then the goal is to compute the number of k-vertex sets that can be (perfectly) matched to a k-clique. Let us express the problem as a conjunctive query

$$\psi_k := x_1 \dots x_k \exists y_1 \dots \exists y_k : \bigwedge_{1 \le i < j \le k} E(y_i, y_j) \wedge \bigwedge_{1 \le i \le k} E(x_i, y_i)\,. \tag{6.4}$$

We point out that ψ_k does not correspond directly to the vertex sets we would like to count as x_i and x_j could be the same vertex in G.

[5]If there is a constant bound on the number of quantified variables then the problem of counting answers to conjunctive queries is reducible to a #W[2]-equivalent problem w.r.t. parameterized Turing reductions. We omit a proof of this statement but point out that it can be done by lifting the results of Chapt. 7.4 in [65] to the realm of counting problems.

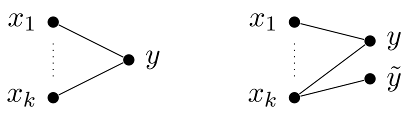


Figure 6.1: *Left:* Graphical representation of the conjunctive query in (6.5). *Right:* A graphical conjunctive query that is "equivalent" to the example on the left in the sense that an assignment $a\colon \{x_1, \dots, x_k\} \to V(G)$ is a partial homomorphism from the left graph to G if and only if it is a partial homomorphism from the right graph to G.

However, it can be shown along the lines of Chapter 6.3.4 that an oracle for counting answers to ψ_k allows us to compute the desired number efficiently and vice versa. Finally, as the linked matching number of ψ_k is not bounded for $k \to \infty$, #A[2]-hardness follows from Theorem 6.6.

Building up on Theorem 6.6 and using the framework of quantum queries, we obtain the following, extensive classification result.

Theorem 6.7. *Let Φ be a family of existential or universal positive formulas with inequalities and non-monotone constraints, both over the free variables. If the arity of Φ is bounded then the problem* #P-MC(Φ) *is either fixed-parameter tractable,* W[1]*-equivalent,* #W[1]*-equivalent,* #W[2]*-hard or* #A[2]*-equivalent.*

Note that allowing the inequalities and non-monotone constraints over all variables, not just the free ones, would in particular include the subgraph decision problem. However, the parameterized complexity of finding a subgraph in G that is isomorphic to a small pattern graph P is a long-standing open question in parameterized complexity (see e.g. [57, Chapt. 33.1]).

6.2.1 Graphical Conjunctive Queries and Colorings

It is instructive to first focus on conjunctive queries with one relation symbol E of arity two. After that, we will generalize the results to arbitrary structures in Chapter 6.5. An example of such a query is the following formula:

$$x_1 \dots x_k \exists y : Ex_1y \wedge \dots \wedge Ex_ky\,. \tag{6.5}$$

The relation E corresponds to a graph G and the free and quantified variables will be assigned vertices of G. In (6.5), an assignment $a_1, \dots, a_k$ in $V(G)$ to the free variables satisfies the formula if and only if the vertices $a_1, \dots, a_k$ have a common neighbor in G. It will be convenient for us to view the formula as a graph H as depicted in Figure 6.1. The vertices of H are partitioned into a set $X = \{x_1, \dots, x_k\}$ of free variables and a set $Y = \{y\}$ of quantified variables. An assignment to the free variables corresponds to a function $a : X \to V(G)$, and such an assignment satisfies the formula if it can be consistently extended to a homomorphism from H to G. This motivates the following definition, where we only consider simple graphs without loops, so we do not allow atoms of the form Ezz.

Definition 6.8. A *graphical conjunctive query* (H, X) consists of a graph H and a set X of vertices of H. We let $\mathsf{Hom}(H, X \to G)$ be the set of all mappings from X to $V(G)$ that can be extended to a homomorphism from H to G, and we call these mappings partial homomorphisms. Formally, the set of partial homomorphisms is defined via

$$\mathsf{Hom}(H, X \to G) = \{a : X \to V(G) \mid \exists h \in \mathsf{Hom}(H \to G) : h|_X = a\} .$$

Given a set Δ of graphical conjunctive queries, the problem #Hom(Δ) asks, given $(H, X) \in \Delta$ and a graph G, to compute $\#\mathsf{Hom}(H, X \to G)$ and is parameterized by $|V(H)|$.

Given two different graphical conjunctive queries (H, X) and $(\hat{H}, \hat{X})$ it might be the case that $\#\mathsf{Hom}(H, X \to \star)$ and $\#\mathsf{Hom}(\hat{H}, \hat{X} \to \star)$ are the same functions. An example for this is given in Figure 6.1. In this case, we say that (H, X) and $(\hat{H}, \hat{X})$ are *equivalent*, denoted as $(H, X) \sim (\hat{H}, \hat{X})$, and the subgraph-minimal elements of the induced equivalence classes are called *minimal*. An explicit notion of equivalence is given in Chapter 6.4. In our proofs, we make use of the following property of minimal queries, whose elementary proof we defer to Chapter 6.5.2, where we generalize it to arbitrary structures.

Lemma 6.9. *Let (H, X) be a minimal and let h be an endomorphism of H. If h maps X bijectively to itself then h is an automorphism.*

While we are ultimately interested in the complexity of computing the number of partial homomorphisms, our hardness proofs become much more pleasant if we consider vertex-colored graphs (see Definition 2.44). Recall, that a graph G is *H-colored* if there is a homomorphism c from G to H. The value $c(v)$ for a vertex $v \in V(G)$ is called the *color* of v and given a vertex $u \in V(H)$ we write $c^{-1}(u)$ for the set of all vertices in G that are mapped to u. Similar to the second part of Chapter 2.5, we consider *color-prescribed homomorphisms*, which are homomorphisms $h \in \mathsf{Hom}(H \to G)$ with the additional property that every vertex $u \in V(H)$ maps to a vertex $h(u)$ whose color is u. Recall that $\mathsf{cp\text{-}Hom}(H \to G)$ denotes the set of all color-prescribed homomorphisms. Now given $X \subseteq V(H)$, we write $\mathsf{cp\text{-}Hom}(H, X \to G)$ for the set of *partial* color-prescribed homomorphisms, that is, functions $a : X \to V(G)$ that can be extended to color-prescribed homomorphisms. The problem #cp-Hom(Δ) is defined analogously to #Hom(Δ). Note that the set $\mathsf{cp\text{-}Hom}(H \to G)$ is empty if the H-coloring of G is not surjective. Therefore it will be convenient to assume that the input graphs of the functions $\#\mathsf{cp\text{-}Hom}(H \to \star)$ and $\#\mathsf{cp\text{-}Hom}(H, X \to \star)$ are surjectively H-colored.

Color-prescribed Homomorphisms Under Taking Minors

Recall that a *minor* of a graph H is any graph that can be obtained from H by deleting vertices and edges and by contracting edges. In this section, we extend the minor relation to graphical conjunctive queries. As it turns out, if M is a minor of a graphical conjunctive query (H, X), then M can be reduced to (H, X).

Recall further, that given a graph H and an edge $e \in E(H)$, we write $H - e$ to denote the graph obtained from H by deleting e and H/e for the graph H where e is contracted. Note that any multiple edges and self-loops are deleted. Similarly, for an isolated vertex $v \in V(H)$ we write $H - v$ for the graph resulting from H by deleting v. A minor of H is any graph that can be obtained from H by iteratively applying these operations.

The deletion and contraction operations extend to graphical conjunctive queries (H, X) in the natural way, but we must decide each time how to modify the set X. For an isolated vertex $v \in V(H)$, we set

$$(H, X) - v := (H - v, X \setminus \{v\}),$$

and for an edge $e \in E(H)$, we set

$$(H, X) - e := (H - e, X).$$

For the contraction operation, let $e \in E(H)$ and let w_e be the vertex that e is contracted to in H/e. The contraction of two quantified variables yields a quantified variable, but as soon as one endpoint of e is a free variable, the contracted variable is a free variable. Formally, we define

$$(H, X)/e = (H/e, X'),$$

where

$$X' = \begin{cases} X\,, & \text{if } e \text{ is disjoint from } X, \\ (X \setminus e) \cup \{w_e\}\,, & \text{otherwise.} \end{cases}$$

Here w_e denotes the vertex that e got contracted to. Similar as to graphs, we say that $(\hat{H}, \hat{X})$ is a *minor* of (H, X) if $(\hat{H}, \hat{X})$ can be obtained from (H, X) by iteratively applying these deletion and contraction operations. We now prove that color-prescribed homomorphisms are "minor-closed", that is, if we know the color-prescribed homomorphisms from (H, X) then we know them from any minor as well. In particular, we emphasize that the following lemma is a generalization of Lemma 2.47 from graphs to graphical conjunctive queries.

Lemma 6.10. *Let (H, X) be a graphical conjunctive query and let $(\hat{H}, \hat{X})$ be a minor of (H, X). Given an $\hat{H}$-colored graph G, we can in polynomial time compute an H-colored graph G' with $|V(G')| \le |V(H)| \cdot |V(G)|$ and*

$$\#\mathsf{cp\text{-}Hom}(\hat{H}, \hat{X} \to G) = \#\mathsf{cp\text{-}Hom}(H, X \to G').$$

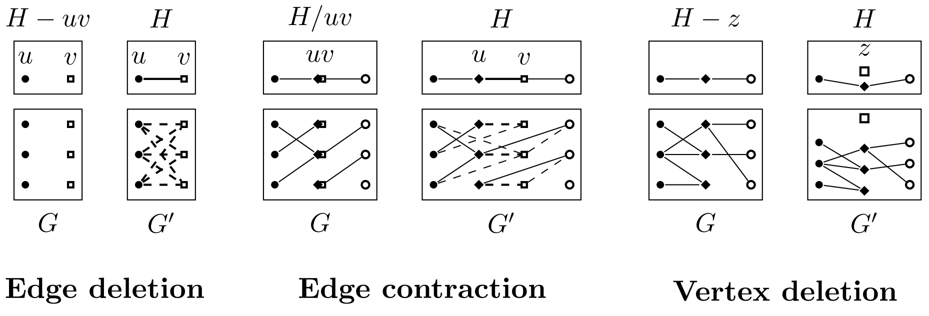


Figure 6.2: Illustration of the reduction for each operation as demonstrated in Lemma 6.10. Edges that are added in the reduction are dashed.

Proof. The claim is trivial if $(\hat{H}, \hat{X})$ and (H, X) are equal. We prove the claim in case $(\hat{H}, \hat{X})$ is obtained from (H, X) by a single deletion or contraction operation, the full result then follows by induction. Figure 6.2 illustrates the proof for each of the three operations. In what follows, it will be convenient to just write uv for an edge $\{u, v\}$.

Edge deletions. Let $e \in E(H)$ be an edge with $e = uv$ and suppose that $\hat{H} = H - e$ and $\hat{X} = X$. Let G be a $\hat{H}$-colored graph given as input, together with the coloring $c\colon V(G) \to V(\hat{H})$. To construct G', we start from G and simply add all possible edges between the color classes $c^{-1}(u)$ and $c^{-1}(v)$; clearly this construction takes polynomial time, G' has the same number of vertices as G, and c is a homomorphism from G' to H. To verify the correctness, we show that $\mathsf{cp\text{-}Hom}(\hat{H}, X \to G) = \mathsf{cp\text{-}Hom}(H, X \to G') =$ holds. Indeed, let $h : V(H) \to V(G)$ a color-prescribed mapping. Since $h(e) \in E(G')$ holds by construction and e is the only constraint where H and $\hat{H}$ differ, the addition of the edge e does not matter. Hence h is an element of $\mathsf{cp\text{-}Hom}(\hat{H}, X \to G)$ if and only if it is an element of $\mathsf{cp\text{-}Hom}(H, X \to G')$. Moreover, the set of partial color-prescribed homomorphisms stays the same.

Vertex deletions. Let $z \in V(H)$ be an isolated vertex and suppose that $\hat{H} = H - z$ and $\hat{X} = X \setminus \{z\}$. Let G be a $\hat{H}$-colored graph given as input, together with the coloring $c\colon V(G) \to V(\hat{H})$. To construct G', we start from G and simply add an isolated vertex z' to it, whose color $c(z')$ we define as z; clearly c is now a homomorphism from G' to H. To verify the correctness, observe that $\#\mathsf{cp\text{-}Hom}(\hat{H}, \hat{X} \to G) = \#\mathsf{cp\text{-}Hom}(H, X \to G')$ holds: Any color-prescribed homomorphism h from H to G' remains a color-prescribed homomorphism from $\hat{H}$ to G by restricting h to $V(\hat{H})$. Conversely, any h from $\hat{H}$ to G can be extended in exactly one color-prescribed way by setting $h(z) = z'$. Thus the number of partial color-prescribed homomorphisms stays the same.

Edge contractions. Let $e \in E(H)$ be an edge with $e = uv$, and suppose $(\hat{H}, \hat{X}) = (H, X)/e$. Contracting the edge e in H identifies the vertices u and v; let us call the new vertex $w \in V(\hat{H})$. Let G be a $\hat{H}$-colored graph given as input, together with the coloring $c\colon V(G) \to V(\hat{H})$. We want to use G' ensure that any color-prescribed homomorphism h from H to G' assigns u and v to the same value, that is, satisfies the equality constraint $h(u) = h(v)$. To do this, we simply put an induced perfect matching in G' between the color class of u and the color class of v. More formally, we start from G and split every vertex $x \in c^{-1}(w)$ into an edge $x_u x_v$ in G', but we leave their neighborhoods intact, that is, we have

$$N_{G'}(x_u) \cap V(G) = N_{G'}(x_v) \cap V(G) = N_G(x)\,.$$

Clearly G' is now H-colored, and it has exactly $|c^{-1}(w)|$ vertices more than G. To verify correctness, again observe that

$$\#\mathsf{cp\text{-}Hom}(\hat{H}, \hat{X} \to G) = \#\mathsf{cp\text{-}Hom}(H, X \to G')$$

holds: Our construction forces any color-prescribed homomorphism h from H to G' to satisfy $h(u) = h(v)$ and thus gives rise to a color-prescribed homomorphism $\hat{h}$ from $\hat{H}$ to G by setting $\hat{h}(w) = h(u)$; this mapping $h \mapsto \hat{h}$ is a bijection. If e is disjoint from X, then $X = \hat{X}$ holds and the set of partial homomorphisms is the same because $h|_X = \hat{h}|_{\hat{X}}$ holds. If e is not disjoint from X, then $\hat{X} \subsetneq X$ holds, but still the mapping $h|_X \mapsto \hat{h}|_{\hat{X}}$ is bijective. In any case, the number of partial homomorphisms is the same. ■

Reducing Color-prescribed to Uncolored Homomorphisms

We show that the number of color-prescribed homomorphism for graphical conjunctive queries can be expressed by using the number uncolored homomorphisms. For the reduction, we need yet another type of homomorphisms as in intermediate step, namely *colorful homomorphisms.* Again, we refer to the second part of Chapter 2.5 where we introduced the notion for graphs. Recall that, in contrast to color-prescribed homomorphisms, *colorful homomorphisms* are homomorphisms $h \in \mathsf{Hom}(H \to G)$ with the less prescriptive property that the image of h contains a vertex for each color, that is, we have $c(h(V(H))) = V(H)$. Recall further, that $\mathsf{cf\text{-}Hom}(H \to G)$ denotes the set of all colorful homomorphisms. Now, given $X \subseteq V(H)$, we write $\mathsf{cf\text{-}Hom}(H, X \to G)$ for the set of *partial* colorful homomorphism, that is, functions $a : X \to c^{-1}(X)$ that can be extended to a colorful homomorphism from H to G. We point out that we only consider functions a satisfying $c(a(X)) = X$. The problem $\#\textsc{cf-Hom}(\Delta)$ is defined analogously to $\#\textsc{Hom}(\Delta)$.

We emphasize the following modifications of our notation prior to the subsequent reductions. The reason for this is the fact that the proofs become much more pleasant to read.

Notation. We write $[k]$ for the set $\{1, \dots, k\}$ and we write $h \circ g$ for the functional composition $x \mapsto h(g(x))$ in the remainder of this thesis.

For the reductions, we need a simple observation about the relationship between H-colored graphs and homomorphisms into them.

Fact 6.11. *Let c be the H-coloring of a graph G and let $h \in \mathsf{Hom}(H \to G)$. Then the function $\pi : v \mapsto c(h(v))$ is an endomorphism of H. If h is colorful and satisfies $c(h(X)) = X$ for a set $X \subseteq H$, then π is an automorphism that maps X to X in such a way that the function composition $h \circ \pi^{-1}$ is a color-prescribed homomorphism.*

Proof. The first statement holds as π is the composition of two homomorphisms $h : V(H) \to V(G)$ and $c : V(G) \to V(H)$. For the second statement observe that colorfulness of h implies that $\pi(V(H)) = V(H)$ and hence, together with the assumption that $c(h(X)) = X$, the endomorphism π is an automorphism that maps X to X. Finally we have that

$$c(h \circ \pi^{-1}(v)) = c(h \circ (c \circ h)^{-1}(v)) = c(h(h^{-1}(c^{-1}(v)))) = v\,,$$

and hence that $h \circ \pi^{-1}$ is color-prescribed. ■

Using Fact 6.11 and by defining a suitable equivalence relation, we obtain the first part of the reduction, namely from #CP-HOM(Δ) to #CF-HOM(Δ). In particular, the following lemma generalizes Lemma 2.51 to graphical conjunctive queries.

Lemma 6.12. *Let (H, X) be a graphical conjunctive query and let G be an H-colored graph. Then*

$$\#\mathsf{cf\text{-}Hom}(H, X \to G) = \#\mathsf{Aut}(H, X) \cdot \#\mathsf{cp\text{-}Hom}(H, X \to G),$$

where $\mathsf{Aut}(H, X) := \{b : X \to X \mid \exists h \in \mathsf{Aut}(H) : h|_X = b\}$.

Proof. We define an equivalence relation $\sim$ on the set $\mathsf{cf\text{-}Hom}(H, X \to G)$ as follows: Two mappings $a, a' \in \mathsf{cf\text{-}Hom}(H, X \to G)$ are equivalent, written $a \sim a'$, if and only if their image is equal, that is, $a(X) = a'(X)$ holds. We denote the equivalence class of a with $[\![a]\!]$.

To show "$\geq$", let $a \in \mathsf{cp\text{-}Hom}(H, X \to G)$ be a partial color-prescribed homomorphism. We show that $[\![a]\!]$ contains at least $\#\mathsf{Aut}(H, X)$ elements, exactly one of which is color-prescribed. Indeed, composing a with a bijection $b \in \mathsf{Aut}(H, X)$ yields distinct functions $a \circ b$, each of which has the same image as a and thus is an element of $[\![a]\!]$. Moreover, each $a \circ b$ can be extended to a colorful homomorphism by composing the assumed automorphism extension of b with the assumed colorful homomorphism extension for a. Finally, $a \circ b$ is color-prescribed only if b is the identity function. Thus each color-prescribed a leads to at least $\#\mathsf{Aut}(H, X)$ distinct colorful $a \circ b$, which proves "$\geq$".

For the backward direction "$\leq$", it suffices to prove that every colorful partial homomorphism $a \in \mathsf{cf\text{-}Hom}(H, X \to G)$ has some partial automorphism $b \in \mathsf{Aut}(H, X)$ such that $a \circ b \in \mathsf{cp\text{-}Hom}(H, X \to G)$ holds. Let h denote the assumed colorful extension of a. Using Fact 6.11, h induces a canonical automorphism $\pi \in \mathsf{Aut}(H)$ which maps X to X. Thus the function b with $b = \pi^{-1}|_X$ is a member of $\mathsf{Aut}(H, X)$. Moreover, by definition of π, the mapping $a \circ b$ is a partial color-prescribed homomorphism. ∎

It remains to reduce colorful homomorphisms to uncolored homomorphisms. We first observe that for minimal queries (H, X), the property $c(a(X)) = X$ already implies the existence of a colorful extension of a.

Observation 6.13. *Let (H, X) be a minimal graphical conjunctive query and let G be an H-colored graph with coloring c. Furthermore let a be a function from X to $V(G)$ satisfying $c(a(X)) = X$. If $h \in \mathsf{Hom}(H \to G)$ is a homomorphism that extends a, then h is colorful.*

Proof. By Fact 6.11, h induces the canonical endomorphism $\pi : v \mapsto c(h(v))$. As h extends a and $c(a(X)) = X$ holds, the endomorphism π bijectively maps X to X. Therefore, by Lemma 6.9, π is an automorphism, which implies that h is colorful. ∎

We proceed with the reduction to uncolored homomorphisms. Again, the following lemma strengthens the corresponding version for graph homomorphisms (see Lemma 2.52). Our proof follows the strategy of multivariate polynomial interpolation.

Lemma 6.14. *Let (H, X) be a minimal graphical conjunctive query. Then there is a deterministic algorithm $\mathbb{A}$ with oracle access to $\#\mathsf{Hom}(H, X \to \star)$ that computes $\#\mathsf{cf\text{-}Hom}(H, X \to \star)$. Furthermore the running time of $\mathbb{A}$ is bounded by $O(f(|H, X|) \cdot n^c)$ for some computable function f and some constant c independent of (H, X).*

Proof. We start by providing an intuition. Let $k = |V(H)|$, $\ell = |X|$ and assume that the vertices of H are the integers $1, \dots, k$ from which the first ℓ are in X. Further, let an H-colored graph G with coloring c be given. For every color $i \in [k]$, we clone (including incident edges) all vertices with color i precisely $z_i - 1$ times for some positive integer z_i. We denote the resulting graph as $G^{\vec{z}}$, which is still H-colored.

Next, the numbers z_i for $i \in [k]$ are interpreted as formal variables and it will turn out that $\#\mathsf{Hom}(H, X \to G^{\vec{z}})$ is a polynomial in $\mathbb{Q}[z_1, \dots, z_k]$. Additionally, the coefficient of $\Pi_{i=1}^{\ell} z_i$ is the number of assignments a from X to $V(G)$ such that $c(a(X)) = X$ and that a can be extended to a homomorphism.

Applying Lemma 6.13 we obtain that those homomorphisms are indeed colorful. We will be able to compute the coefficient by multivariate interpolation. Note that the evaluation of the polynomial in $\vec{z}$ can be done by querying the oracle for $G^{\vec{z}}$.

Formally, we define an equivalence relation on $\mathsf{Hom}(H, X \to G^{\vec{z}})$ as follows. Two assignments $a_{\vec{z}}$ and $a'_{\vec{z}}$ are equivalent if and only if for every $x \in X$ it holds that $a_{\vec{z}}(x)$ and $a'_{\vec{z}}(x)$ are clones of the same vertex. Note that every equivalence class corresponds to precisely one mapping $a \in \mathsf{Hom}(H, X \to G)$ and we write $[\![a]\!]_{\vec{z}}$ for that class.

Next observe that every $a \in \mathsf{Hom}(H, X \to G)$ induces a color-vector

$$c_a = (c(a(1), \dots, c(a(\ell))) \in [k]^\ell ,$$

which allows us to express the size of $[\![a]\!]_{\vec{z}}$ as $\Pi_{i=1}^{\ell} z_{c_a(i)}$. This yields the following polynomial for $\#\mathsf{Hom}(H, X \to G^{\vec{z}})$:

$$\#\mathsf{Hom}(H, X \to G^{\vec{z}}) = \sum_{a \in \mathsf{Hom}(H, X \to G)} \#[\![a]\!]_{\vec{z}} = \sum_{v \in [k]^\ell} \sum_{\substack{a \in \mathsf{Hom}(H, X \to G) \\ c_a = v}} \prod_{i=1}^{\ell} z_{c_a(i)}$$

Finally it can be verified easily that the coefficient of $\Pi_{i=1}^{\ell} z_i$ is indeed the number of assignments a from X to $V(G)$ such that $c(a(X)) = X$ and that a can be extended to a homomorphism h. Note that h is colorful by minimality of H and Lemma 6.13. In other words, the coefficient of $\Pi_{i=1}^{\ell} z_i$ is precisely $\#\mathsf{cf}\text{-}\mathsf{Hom}(H, X \to G)$.

As we can evaluate the polynomial for every vector $\vec{z} \in \mathbb{N}_{>0}^k$ the coefficient can be computed by Theorem 2.28. ■

6.3 Classifying Graphical Conjunctive Queries

In this section, we classify the complexity of counting homomorphisms for classes of graphical conjunctive queries. From Chapter 6.3.4 on we will also consider the color-prescribed variant which yields significantly more pleasant proofs as Lemma 6.10 allows us to reduce from minors of conjunctive queries in this case. Using the observations in Chapter 6.2.1, the hardness results we discuss for color-prescribed homomorphisms carry over to the uncolored situation and yield our refined complexity classification for the case of graphs (Theorem 6.6).

The first five subsections correspond to the five cases in the Complexity Pentachotomy. In each case, we define the precise parameters that we need in order to classify the complexity of $\#\textsc{Hom}(\Delta)$, and we also give an example class of queries that exhibits that complexity. All five example classes along with their structural properties are depicted in Figure 6.3.

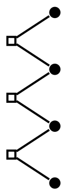
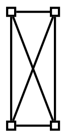
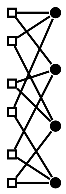
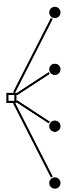
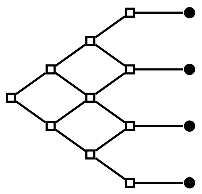
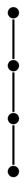
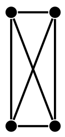
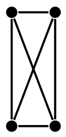
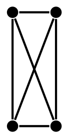
	Δ_{poly}	$\Delta_{\mathsf{W[1]}}$	$\Delta_{\#\mathsf{W[1]}}$	$\Delta_{\#\mathsf{W[2]}}$	$\Delta_{\#\mathsf{A[2]}}$
Query					
contract		$\emptyset$			
tw	$O(1)$	∞	∞	$O(1)$	∞
tw(contract)	$O(1)$	$O(1)$	∞	∞	∞
dss	$O(1)$	$O(1)$	$O(1)$	∞	∞
lmn	$O(1)$	$O(1)$	$O(1)$	$O(1)$	∞
Complexity	P	W[1]-eq.	#W[1]-eq.	#W[2]-hard	#A[2]-eq.†

Figure 6.3: Illustration of the five typical classes of conjunctive queries that we discuss in Chapter 6.3. Depicted is the query (H, X) for $k = 4$, where free variables (i.e., vertices in X) are drawn as solid discs and quantified variables (i.e., vertices in $V(G) \setminus X$) are drawn as hollow squares. We also display the contract (see Definition 6.15) of each query for $k = 4$. We write $O(1)$ whenever a parameter is bounded by a constant in the entire query class, and ∞ whenever it is unbounded.

†The observant reader might notice that the queries in $\Delta_{\#\mathsf{A[2]}}$ are not minimal. For this reason, the #A[2]-equivalence in the last column only holds for classes of minimal queries that contain $\Delta_{\#\mathsf{A[2]}}$ as minors, or for the color-precsribed variant, that is for #cp-Hom($\Delta_{\#\mathsf{A[2]}}$).

In the sixth subsection, we are then in position to restate the full classification theorem in an explicit fashion, relying on the structural parameters defined in Subsections 1–5, The first three subsections should be considered a review of previous work [58, 31], which is necessary to formally state our techniques and results.

6.3.1 Query Classes that are Polynomial-time

Which classes Δ of graphical conjunctive queries yield polynomial-time algorithms for #Hom(Δ)? Chen, Durand and Mengel [58, 31] proved that the problem is polynomial-time computable if all graphs in Δ as well as their *contracts* have at most a constant treewidth.

Definition 6.15 (Contract). The *contract* of a conjunctive query (H, X) is a graph on the vertex set X, obtained by adding an edge between two vertices u and v in X if $\{uv\}$ is an edge of H or if there exists a connected component C in $H \setminus X$ that is adjacent to both u and v. Given a class Δ of conjunctive queries, we write $\mathsf{contract}(\Delta)$ for the set of all of its contracts.

The following example class Δ_{poly} of queries is satisfied by the k-tuples

$$v_1, v_3, \ldots, v_{2k-1}$$

of vertices in the input graph G for which there exists an extension

$$v_2, v_4, \ldots, v_{2k-2}\,,$$

such that $v_1, \ldots, v_{2k-1}$ is a walk in G. $\Delta_{\mathsf{poly}} = \{\psi_k \mid k \in \mathbb{N}\}$, where

$$\psi_k := x_1 \ldots x_k \; \exists y_1 \ldots \exists y_{k-1} : \bigwedge_{1 \le i < k} E x_i y_i \wedge E y_i x_{i+1}\,. \tag{6.6}$$

Since these queries and their contracts are just paths (cf. Figure 6.3), their treewidth is bounded by a constant. Thus the problem $\#\text{HOM}(\Delta_{\mathsf{poly}})$ is polynomial-time computable by the complexity trichotomy of Chen, Mengel and Durand [58, 31]. This can be seen more directly using dynamic programming, or by considering the square A^2 of the adjacency matrix of G and replacing each positive entry by a 1 to obtain B – then the sum of all entries in B^k is the desired number.

Formally, the trichotomy theorem of [58, 31] is as follows:

Theorem 6.16 ([58, 31]). *Let Δ be a set of minimal conjunctive queries.*

(1) If the treewidth of the formulas in Δ and their contracts, is bounded then $\#\text{HOM}(\Delta)$ is solvable in polynomial time.

(2) If the treewidth of the formulas is unbounded but the treewidth of the contracts is bounded, then $\#\text{HOM}(\Delta)$ is $\mathsf{W}[1]$-equivalent.

(3) If the treewidth of the contracts is unbounded, then $\#\text{HOM}(\Delta)$ is hard for $\#\mathsf{W}[1]$.

6.3.2 Query Classes that are W[1]-equivalent

As it turns out, the situation in which the treewidth of the queries and their cores is bounded appears to be the only one that is polynomial-time computable: If the treewidth of Δ or $\mathsf{contract}(\Delta)$ is unbounded, then Theorem 6.16 implies that $\#\text{HOM}(\Delta)$ is not polynomial-time computable, unless all $\mathsf{W}[1]$-easy problems are fixed-parameter tractable and hence ETH fails. More precisely, when the treewidth of $\mathsf{contract}(\Delta)$ is bounded but the treewidth of Δ is unbounded, then the problem is $\mathsf{W}[1]$-equivalent. To exemplify this latter situation further, note that the $\mathsf{W}[1]$-equivalent problem Clique is a special case: The following query ψ_k in the class $\Delta_{\mathsf{W}[1]}$ is satisfiable (i.e., has the empty tuple as a satisfying assignment) if and only if the input graph has a clique of size k. Formally $\Delta_{\mathsf{W}[1]} = \{\psi_k \mid k \in \mathbb{N}\}$, where

$$\psi_k := \exists y_1 \ldots \exists y_k : \bigwedge_{1\le i<j\le k} E y_i y_j \,. \tag{6.7}$$

Indeed, the contract of each query in $\Delta_{\mathsf{W}[1]}$ is the empty graph, but the treewidth of the k-th query is equal to $k-1$, so $\textsc{Hom}(\Delta_{\mathsf{W}[1]})$ is $\mathsf{W}[1]$-equivalent by Theorem 6.16.

6.3.3 Query Classes that are #W[1]-equivalent

If the treewidth of $\mathsf{contract}(\Delta)$ is unbounded, then $\#\textsc{Hom}(\Delta)$ is $\#\mathsf{W}[1]$-hard by Theorem 6.16. We now define the *dominating star size*, a structural parameter with the property that, if all elements of Δ have bounded dominating star size, then $\#\textsc{Hom}(\Delta)$ is $\#\mathsf{W}[1]$-easy.

Definition 6.17 (Dominating star size). Let (H, X) be a conjunctive query and let $Y_1, \ldots, Y_\ell$ be the connected components of the induced subgraph $H[V(H) \setminus X]$ given by the quantified variables. Further, let k_i be the number of vertices $x \in X$ for which there exists a vertex $y \in Y_i$ that is adjacent to x. The *dominating star size* of (H, X) is defined via

$$\mathsf{dss}(H, X) = \max\{k_i \mid i \in \ell\} \,.$$

This notion is identical to the notion of *strict star size*, which was used by Chen and Mengel [31] in an intermediate step of their $\#\mathsf{W}[1]$-hardness proof.

Before we prove that bounded dss implies $\#\mathsf{W}[1]$-easiness, we first give an example query class $\Delta_{\#\mathsf{W}[1]}$ that fits into this situation. The query ψ_k contains as satisfying assignments exactly those tuples $v_1, \ldots, v_k$ of vertices in G such that there is a length-2 walk $v_i w_{ij} v_j$ in G for any distinct i, j. Formally, $\Delta_{\#\mathsf{W}[1]} = \{\psi_k \mid k \in \mathbb{N}\}$, where

$$\psi_k := x_1 \ldots x_k : \bigwedge_{1\le i<j\le k} \exists y_{ij} : E x_i y_{ij} \wedge E y_{ij} x_j \,. \tag{6.8}$$

Note that the graphical representation of ψ_k corresponds to a subdivided k-clique (cf. Figure 6.3), and its contract is a k-clique. Thus the queries of $\Delta_{\#\mathsf{W}[1]}$ and their contracts have unbounded treewidth. However, the dominating star size is equal to 2 because each connected component of the graph $H[V(G) \setminus X]$ consists of a variable y_{ij} which has two neighbors.

The $\#\mathsf{W}[1]$-hardness of $\#\textsc{Hom}(\Delta_{\#\mathsf{W}[1]})$ as claimed by Theorem 6.16 can be proved using a straightforward reduction from counting multicolored cliques of size k, where each edge is subdivided once. Conversely, we establish that $\#\textsc{Hom}(\Delta_{\#\mathsf{W}[1]})$ is $\#\mathsf{W}[1]$-easy by reducing it *to* counting cliques. We prove the special case of $\Delta_{\#\mathsf{W}[1]}$ here for illustration and then sketch the proof of the general result when the dominating star size is bounded.

Lemma 6.18. *$\#\textsc{Hom}(\Delta_{\#\mathsf{W}[1]})$ is $\#\mathsf{W}[1]$-easy.*

Proof. Given ψ_k for some $k \in \mathbb{N}$ and a graph G, we wish to compute $\#\mathsf{Hom}(\psi_k \to G)$. We reduce to the problem of counting cliques. First, we construct a graph G' from G as follows. The vertex set of G' consists of k copies of the vertex set of G. We add an edge between two vertices u and v in G' if and only if they are contained in different copies and if there exists a vertex z such that $\{u, z\}$ and $\{z, v\}$ are edges in G. Now it can easily be observed that $\#\mathsf{Hom}(\psi_k \to G)$ equals $k!$ times the number of cliques of size k in G'. This concludes the proof. ∎

Important in this proof is the preprocessing phase, where for every pair of vertices u and v we check if they have a common neighbor. After that, we expressed the problem as a homomorphism counting problem without quantified variables. Indeed, whenever the dominating star size is bounded, the preprocessing works and allows us to get rid of the quantified variables. The remainder of the reduction to a $\#\mathsf{W}[1]$-easy problem follows from the fact that counting answers to model-checking problems without quantified variables is $\#\mathsf{W}[1]$-equivalent by Theorem 2.59.

Theorem 6.19. *Let Δ be a class of graphical conjunctive queries such that the dominating star size of queries in Δ is bounded. Then $\#\textsc{Hom}(\Delta)$ is $\#\mathsf{W}[1]$-easy.*

We give a sketch here, the formal proof is deferred to the general case of structures in Chapter 6.5.3.

Proof (Sketch). Let $c \in \mathbb{N}$ be the maximum dominating star size among all queries in Δ. Let $\delta \in \Delta$ be a conjunctive query with free variables X and quantified variables Y. Furthermore let (H, X) be the associated graph of δ. Recall that $H[Y]$ and $H[X]$ are the induced subgraphs of H that only contain vertices in Y and in X, respectively. Furthermore, we let $Y_1, \ldots, Y_\ell$ be the connected components of $H[Y]$. Given a graph G, we wish to compute $\#\mathsf{Hom}(\delta \to G)$. Since $\mathsf{dss}(H, X) \leq c$, the number of vertices in X that are adjacent to a vertex in Y_i in H is bounded by c.

This allows us to perform the following preprocessing: For every tuple $\vec{v} = (v_1, \ldots, v_c)$ of vertices in G and for every $i \in [\ell]$, we check whether $\vec{v}$ is a candidate for the image of the neighbors of Y_i in an answer to δ. Note that these checks can be done using an oracle for a (#)$\mathsf{W}[1]$-equivalent problem as they can equivalently be expressed as a (decision version of a) model checking problem where all variables are existentially quantified, which is known to be $\mathsf{W}[1]$-easy (see e.g. Theorem 7.22 in [65]).

After performing all of those checks — at most $\ell \cdot n^c$ many — we need to count the number of homomorphisms from $H[X]$ to G that additionally are consistent with the checks. This final step can be expressed as a counting model checking problem such that every variable is free, more precisely, as an instance of $\#\text{P-MC}(\Pi_0)$ which is known to be $\#\mathsf{W}[1]$-easy by Theorem 2.59. ∎

Remark 6.20. For each fixed conjunctive query (H, X), we can use the preprocessing of Theorem 6.19 to obtain a deterministic algorithm for computing $\#\mathsf{Hom}(H, X \to \star)$: Each oracle query is answered by a subroutine that uses standard dynamic programming over the tree decompositions of $H[Y]$ and the contract of (H, X) (see Theorem 2.35 and Remark 2.42). The overall running time of the algorithm is bounded by

$$O\left(n^{\mathsf{dss}(H,X)+\mathsf{tw}(H[Y])+1} + n^{\mathsf{tw}(\mathsf{contract}(H,X))+1}\right).$$

6.3.4 Query Classes that are #W[2]-hard

We have seen that $\#\text{Hom}(\Delta)$ is $\#\mathsf{W}[1]$-easy if the dominating star size of Δ is bounded. We now show that the dominating star size is the right parameter for this complexity demarcation, since if it is unbounded for Δ, then we show the problem to be $\#\mathsf{W}[2]$-hard. To this end, we will from now on consider its color-prescribed variant. Recall that the problem $\#\text{cp-Hom}(\Delta)$ is given $\delta \in \Delta$ and a δ-colored graph G and the task is to compute

$$\#\mathsf{cp\text{-}Hom}(\delta \to G).$$

It is parameterized by the size of δ.

As an example, consider the queries ψ_k in $\Delta_{\#\mathsf{W}[2]}$, which have as satisfying assignments exactly the tuples

$$v_1, \ldots, v_k$$

of vertices in G whose neighborhood contains at least one common vertex. Formally, $\Delta_{\#\mathsf{W}[2]} = \{\psi_k \mid k \in \mathbb{N}\}$, where

$$\psi_k := x_1 \ldots x_k \exists y : \bigwedge_{1 \le i \le k} E x_i y. \tag{6.9}$$

Note that $\mathsf{dss}(\psi_k) = k$ holds because the only quantified variable y has k neighbors. Thus $\Delta_{\#\mathsf{W}[2]}$ has unbounded dominating star size. Moreover, the negated formula $\neg\psi_k$ on the complement graph $\overline{G}$ has exactly the dominating sets (or rather, tuples) of size at most k as its satisfying assignments. Since the counting problem #CP-HOM(Δ) allows for this negation by subtracting the number of color-prescribed solutions of ψ_k from the number of all possible color-prescribed tuples, it is clear that #CP-HOM(Δ) is indeed #W[2]-hard. Using the same observation, it is also clear that counting solutions of ψ_k cannot be done in time $O(n^{k-\varepsilon})$ for any $\varepsilon > 0$ unless SETH is false. This implies the following lemma.

Lemma 6.21. #CP-HOM($\Delta_{\#\mathsf{W}[2]}$) *is* #W[2]*-hard. Furthermore, for every* $k \geq 3$*, counting answers to* ψ_k *cannot be done in time* $O(n^{k-\varepsilon})$ *for any* $\varepsilon > 0$ *unless SETH fails.*

Proof. We construct a reduction from the problem #DOMSET of counting dominating sets of size k, which is known to be #W[2]-hard when parameterized by k and which cannot be solved in time $O(n^{k-\varepsilon})$ for any $\varepsilon > 0$ assuming SETH holds by Theorem 2.21.[6] Intuitively, the proof exploits that the set of solutions to ψ_k is in some sense the complement of the set of all k-dominating sets and that the ability to count solutions allows us to compute the cardinality of the complementary set. Let $k \in \mathbb{N}$ and let G be a graph. It will be convenient to relabel the quantified variable in ψ_k with 0 and the k free variables with $1, \dots, k$. Recall that a subset S of vertices dominates a graph G if every vertex in $V(G) \setminus S$ is adjacent a vertex in S. We first show how to compute the cardinality of the following set using an oracle for $\#\mathsf{cp\text{-}Hom}(\psi_k \to \star)$:

$$\mathsf{Dom}_k(G) := \{a : [k] \to V(G) \mid \mathsf{im}(a) \text{ dominates } G\}\,.$$

We assume a given graph G to be not complete as otherwise $\mathsf{Dom}_k(G)$ can be computed trivially. Now a ψ_k-colored graph G' is constructed from G as follows. First, we take $k+1$ copies $V^0, \dots, V^k$ of the vertex set of G and color V^0 with the quantified variable 0 and V^i with the free variable i for each $i \in [k]$. Finally, for every $i \in [k]$, we add an edge between a pair of vertices $u \in V^0$ and $v \in V^i$ if and only if the primal vertices of u and v are *not* adjacent in G. Observe that G' is indeed ψ_k-colored as G is not a complete graph. Now let F be the set of all assignments a from $[k]$ to $V(G')$ such that for all $i \in [k]$ the vertex $a(i)$ is colored with i, i.e., contained in V^i. Then we have that $\mathsf{cp\text{-}Hom}(\psi_k \to G') \subseteq F$ and, in particular,

$$\begin{aligned}
&F \setminus \mathsf{cp\text{-}Hom}(\psi_k \to G') \\
&= F \ \setminus \ \{a : [k] \to V(G') \mid a(i) \in V^i \wedge \exists y \in V^0 : \{a(i), y\} \in E(G')\} \\
&= \{a : [k] \to V(G') \mid a(i) \in V^i \wedge \forall y \in V^0 : \{a(i), y\} \notin E(G')\}
\end{aligned}$$

Observe that by construction of G', the cardinality of the latter set is equal to $\#\mathsf{Dom}_k(G)$. As $\#F = \#V(G)^k$ we hence obtain

$$\#V(G)^k - \#\mathsf{cp\text{-}Hom}(\psi_k \to G') = \#\mathsf{Dom}_k(G)\,.$$

[6]See also [109] for the case of a fixed $k \geq 3$.

Now, given a graph G and $j \in \mathbb{N}$, we define G^j to be the graph obtained from G by adding j isolated vertices. Furthermore we let $\mathsf{Surj}(i,j)$ be the number of surjections from $[i]$ to $[j]$. Then we claim that

$$\#\mathsf{Dom}_k(G^j) = \sum_{i=1}^{k} \binom{k}{i} \cdot \mathsf{Surj}(i,j) \cdot \#\mathsf{Dom}_{k-i}(G)\,. \tag{6.10}$$

To see this we observe that every isolated vertex has to be in the image of every $a \in \mathsf{Dom}_k(G^j)$. Hence we can partition the elements in $\mathsf{Dom}_k(G^j)$ by the number of elements in $[k]$ that are mapped to the isolated vertices. Let i be this number. Then there are $\binom{k}{i}$ possibilities to choose these elements and $\mathsf{Surj}(i,j)$ to map them to the isolated vertices. Finally, the remaining $k-i$ elements have to be mapped to $V(G)$ such that their image dominates G. We observe that (6.10) yields a system of linear equations such that the corresponding matrix is triangular if proper values for j are chosen. Hence we can compute all numbers $\#\mathsf{Dom}_\ell(G)$ for $\ell \le k$. Finally we show how to use these numbers to compute the numbers $D_1,\dots,D_k$ of dominating sets of size $i = 1,\dots,k$ in G. We proceed inductively. If $i = 1$ we have that $D_1 = \#\mathsf{Dom}_1(G)$. Otherwise let $\ell \le k$ and assume that $D_1,\dots,D_{\ell-1}$ have be computed so far. It can easily be seen that

$$\#\mathsf{Dom}_\ell(G) = \sum_{i=1}^{\ell} D_i \cdot \mathsf{Surj}(\ell,i)\,,$$

and therefore

$$D_\ell = \mathsf{Surj}(\ell,\ell)^{-1} \cdot \left(\#\mathsf{Dom}_k(G) - \sum_{i=1}^{\ell-1} D_i \cdot \mathsf{Surj}(\ell,i)\right)\,.$$

The above steps constitute a tight reduction from counting dominating sets of size k to counting solutions to $\#\mathsf{cp}\text{-}\mathsf{Hom}(\psi_k \to \star)$ which implies both, the lower bound under SETH and $\#\mathsf{W}[2]$-hardness of $\#\textsc{cp-Hom}(\Delta_{\#\mathsf{W}[2]})$. ∎

The class $\Delta_{\#\mathsf{W}[2]}$ is not only an example of a class for which $\textsc{Hom}(\Delta_{\#\mathsf{W}[2]})$ is $\#\mathsf{W}[2]$-hard, but it is the minimal one. Indeed, every class Δ of unbounded dominating star size contains arbitrarily large elements of $\Delta_{\#\mathsf{W}[2]}$ as a minor, and we have already seen that this implies that $\textsc{Hom}(\Delta_{\#\mathsf{W}[2]})$ reduces to $\textsc{Hom}(\Delta)$. Using the fact that counting color-prescribed answers to a conjunctive query is at least as hard as counting color-prescribed answers for any minor of the query (Lemma 6.10), we are now able to prove the following theorem.

Theorem 6.22. *Let Δ be a recursively enumerable class of conjunctive queries with unbounded dominating star size. Then $\#\textsc{cp-Hom}(\Delta)$ is $\#\mathsf{W}[2]$-hard, and given a formula δ with $\mathsf{dss}(\delta) \ge 3$, computing $\#\mathsf{cp}\text{-}\mathsf{Hom}(\delta \to \star)$ cannot be done in time $O(n^{\mathsf{dss}(\delta)-\varepsilon})$ for any $\varepsilon > 0$ unless SETH fails.*

Proof. Assume that we are given such a class Δ. As the dominating star size of Δ is unbounded, we have that for every k, there exists $\delta_k \in \Delta$ with dominating star size $\geq k$ and hence ψ_k is a minor of δ_k. Therefore we have that the set $\Delta_{\#\mathsf{W}[2]}$ is a set of minors of Δ. By Lemma 6.21 and Lemma 6.10 the claim of the theorem follows. ∎

6.3.5 Query Classes that are #A[2]-equivalent

Recall that the parameterized complexity class $\#\mathsf{A}[2]$ is defined via the model checking problem of universally quantified first-order formulas, and it is not known to be equal to $\#\mathsf{W}[2]$. By the same observation as in the preceding subsection, it follows that $\#\textsc{Hom}(\Delta)$ and $\#\textsc{cp-Hom}(\Delta)$ are $\#\mathsf{A}[2]$-easy, which is made formal in Appendix C. We now introduce the structural parameter *linked matching number* for conjunctive queries that, if unbounded for Δ, leads to $\#\mathsf{A}[2]$-equivalence. To define the parameter, we use the notion of a node-well-linked set.

Definition 6.23 (Node-well-linked). Let G be a graph. A set $S \subseteq V(G)$ is called *node-well-linked* if, for every two disjoint and equal-sized subsets A, B of S, there are $|A|$ vertex disjoint paths in G that connect the vertices in A with the vertices in B.

Node-well-linked sets play a central role in the theory of graph minors, particularly in the proof of Chekuri, Chuzhoy and Tan [29, 36, 37] for the Excluded-Grid-Theorem (see Theorem 2.6 and Theorem 2.7). Indeed, if large node-well-linked sets exist in a graph, then its treewidth is large and it contains a large grid as a minor. We now introduce a structural parameter for conjunctive queries that measures the size of the largest set that is node-well-linked in the quantified variables and has a saturating matching to the free variables. Furthermore, we say that set of vertices S of a graph G is *ℓ-connected* if, for every pair A, B of disjoint size-ℓ subsets of S, there are ℓ vertex-disjoint paths in G connecting A and B. A *separation* of a graph is an ordered pair (A, B) of vertex subsets of G such that $V(A) \cup V(B) = V(G)$ and there are no edges between $A \setminus B$ and $B \setminus A$.

Definition 6.24 (Linked matching number). Let (H, X) be a graphical conjunctive query, let Y be the set $V(H) \setminus X$ of quantified variables, and let M be a matching from X to Y. The matching M is called *linked* if the set $V(M) \cap Y$ is node-well-linked in the graph $H[Y]$. The *linked matching number* lmn of (H, X) is the size of the largest linked matching of H.

We prove later that queries with a large linked matching number not only have large treewidth, but they also contain a large *grate* as a minor. Informally, a grate is just half of a grid that lives in the quantified variables and is cut along its diagonal, and the diagonal has a saturating matching to the free variables (the upper right corner of Figure 6.3 depicts the 4-grate).

Definition 6.25. For a positive integer k, the *k-grate* is the graphical conjunctive query whose k free variables are x^i_j for $i, j \geq 0$ with $i + j = k - 1$, and whose quantified variables are y^i_j for $i, j \geq 0$ with $i + j \leq k - 1$. The edges between free and quantified variables are $x^i_j y^i_j$ for $i + j = k - 1$. The edges on the quantified variables are $y^i_j y^i_{j+1}$ and $y^i_j y^{i+1}_j$. Let Grates be the set of all grates.

We now sketch a proof that #cp-Hom(Grates) is #A[2]-hard. The full proof requires lifting a rather technical normalization theorem for A[2] to #A[2] which is done in Appendix C. In particular, this normalization implies that the general problem of counting answers to conjunctive queries in graphs is #A[2]-equivalent.

Lemma 6.26. *The problem* #cp-Hom(Grates) *is* #A[2]*-hard.*

Proof (Sketch). The construction is similar to the reduction in Lemma 2.45. We will reduce from #cp-Hom(Γ) where Γ contains the queries

$$\gamma_k := x_1 \dots x_k \exists y_1 \dots \exists y_k : \bigwedge_{i=1}^{k} E x_i y_i \wedge \bigwedge_{1 \leq i < j \leq k} E y_i y_j \tag{6.11}$$

for all $k \in \mathbb{N}$. #cp-Hom(Γ) is shown to be #A[2]-hard in Appendix C in the context of the normalization result. Intuitively, answers to γ_k are vertex sets of size at most k that can be perfectly matched to a clique.

Now let ω_k be the k-grate. Roughly speaking, given a γ_k-colored graph G for which we want to compute $\#\mathsf{cp\text{-}Hom}(\gamma_k \to G)$, we just need to modify the part of G that is colored with quantified variables. To this end recall that in case of γ_k, the free variables have to be connected to a clique of quantified variables by a matching and in case of ω_k, the free variables have to be connected to vertices on the diagonal of a half-grid by a matching. Now, given G, we obtain a new graph G' by first deleting all edges between vertices that are colored with quantified variables, and then adding blocks of vertices that correspond to the former edges in a half-grid like manner. Then, given two vertices a and a' corresponding to the former edges $\{u, v\}$ and $\{u', v'\}$ we add an edge between a and a' if and only if either $u = u'$ and the blocks of a and a' are adjacent horizontally or $v = v'$ and the blocks of a and a' are adjacent vertically. We encourage the reader to verify the correctness of the construction in the case $k = 3$ using Figure 6.4. ∎

Next we show that every class Δ with unbounded linked matching number contains arbitrarily large grate minors and point out that this constitutes a generalization of the Excluded-Grid-Theorem (Theorem 2.6). Due to the hardness of #cp-Hom(Grates) (Lemma 6.26) and using the fact that the homomorphism counting problem is "minor-closed" (Lemma 6.10), this yields the #A[2]-hardness of #cp-Hom(Δ).

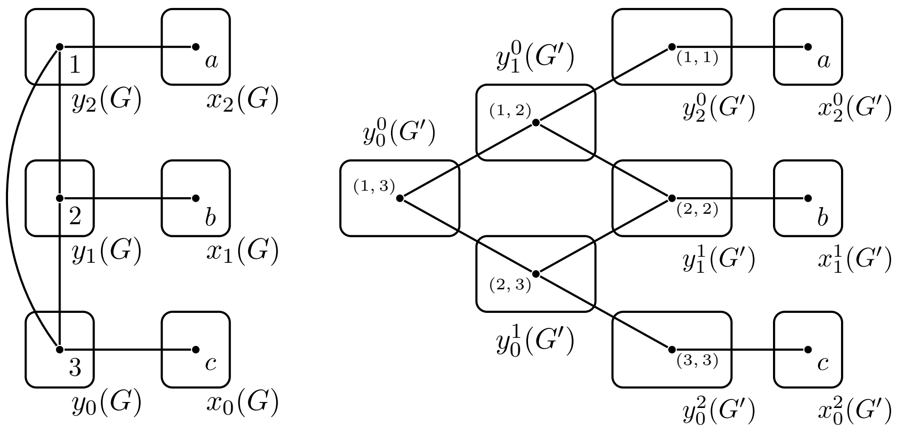


Figure 6.4: Illustration of the construction of G' for $k = 3$. The graph G (*left*) is γ_3-colored and the mapping $m = \{x_2 \mapsto a, x_1 \mapsto b, x_0 \mapsto c\}$ is contained in $\mathsf{cp\text{-}Hom}(\gamma_3 \to G)$ as a, b, c are connected to $1, 2, 3$ by a matching and $1, 2, 3$ form a clique. The graph G' (*right*) is ω_3-colored and m corresponds to the mapping $m' = \{x_2^0 \mapsto a, x_1^1 \mapsto b, x_0^2 \mapsto c\}$. Now m' is contained in $\mathsf{cp\text{-}Hom}(\omega_3 \to G')$ as a, b, c are connected to $(1,1), (2,2), (3,3)$ by a matching and $(1,1), (2,2), (3,3)$ are the diagonal of a half-grid. Note that the latter is only the case as there are vertices $(1,2), (2,3)$ and $(1,3)$ that correspond to the edges of the clique in G.

We use the work of Marx, Seymour and Wollan [97] as well as of Diestel et al. [51] for an easy proof of the following theorem. Recall that the $(g \times g)$-grid $\boxplus_g$ has vertices $v_{i,j}$ for $i, j \in \{1, \dots, g\}$ and edges $v_{i,j}v_{i+1,j}$ for $i < g$ and $v_{i,j}v_{i,j+1}$ for $j < g$. Recall further from Definition 2.3 that a minor mapping η from a graph H to a graph G is a function mapping vertices v in $V(H)$ to sets $\eta(v) \subseteq V(G)$ such that the following constraints are satisfied:

- For every $v \in V(H)$ the graph $G[\eta(v)]$ is nonempty and connected.
- For all $u, v \in V(H)$ with $u \neq v$ the sets $\eta(u)$ and $\eta(v)$ are disjoint.
- For all edges $\{u, v\} \in E(H)$ there exist $u' \in \eta(u)$ and $v' \in \eta(v)$ such that $\{u', v'\} \in E(G)$.

Theorem 6.27. *For all integers $g > 0$ there exists $\kappa \geq 1$ such that the following is true. Let G be a graph and $X \subseteq V(G)$ be a node-well-linked set of size at least κ. Then there exists a minor mapping η from $\boxplus_g$ to G satisfying that for all $j \in [g]$ there exists $x \in X$ such that $x \in \eta(v_{1,j})$.*

Proof. Given a graph G, we write $\mathsf{Sep}(G)$ for the set of all separations of G. We apply Theorem 1.2 of [97] with $k = g$. By the theorem, there exists a number $K \in \mathbb{N}$ such that for any tangle[7] $\mathcal{T}$ of order at least K in G and any set $Z \subseteq V(G)$ with $|Z| = g$ the following is true.

[7]For the purpose of this proof we do not need the definition of a tangle. The interested reader is referred to e.g. Chapter 4 in [51].

Fact 6.28. *If there is no separation $(A, B) \in \mathcal{T}$ with $|V(A \cap B)| < g$ and $Z \subseteq V(A)$ then there is a minor mapping η from $\boxplus_g$ to G satisfying that for all $j \in [g]$ there exists $z \in Z$ such that $z \in \eta(v_{1,j})$.*

Now let $\ell := \max\{g, K\}$ and $\kappa := 3\ell$. Hence X is an node-well-linked set of size 3ℓ. In particular, X is an ℓ-connected set of size 3ℓ. Diestel et al. (see Chapter 4 in [51]) have shown that the following is a tangle of order $\ell \geq K$ in G:

$$\mathcal{T}[X] := \{(A, B) \mid (A, B) \in \mathsf{Sep}(G) \wedge |V(A \cap B)| < \ell \wedge |V(A) \cap X| \leq |V(B) \cap X|\}$$

Next let Z be any subset of X of size $g \leq \ell$ and assume for contradiction that there exists a separation $(A, B) \in \mathcal{T}[X]$ such that $|V(A \cap B)| < g$ and $Z \subseteq V(A)$. By the definition of $\mathcal{T}[X]$ we have that

$$|V(B) \cap X| \geq |V(A) \cap X| \geq |Z| = g\,.$$

Consequently there exists $Z' \in V(B) \cap X$ with $|Z'| = g$ and $Z \cap Z' = \emptyset$. As X is $\ell \geq g$-connected, there are g vertex-disjoint paths from Z to Z'. Therefore, by Menger's Theorem, $|V(A \cap B)| < g$ is false and we obtain a contradiction. We hence conclude the proof with the application of Fact 6.28 and the observation that Z was chosen to be a subset of X. ∎

Theorem 6.29 (Excluded-Grate-Theorem). *Let Δ be a class of graphical conjunctive queries. If the linked matching number of Δ is unbounded, then Δ contains arbitrarily large grates as minors.*

Proof. Let $g \in \mathbb{N}$ and let ω_g be the g-grate. We show that ω_g is a minor of some query in Δ. To this end, invoke Theorem 6.27 with g to obtain κ for which its statement is true. Now let $\delta \in \Delta$ be a query with linked matching number at least κ and let (H, X) be the query graph of δ. By assumption there exists a set S of at least κ many vertices in the quantified variables that is node-well-linked in the $H \setminus X$ and that is connected to X by a matching. By Theorem 6.27 there exists a minor mapping η from the $g \times g$-grid, such that for every vertex v in the first column of the grid, we have that $\eta(v)$ contains an element of S. The grid minor can now further be contracted to obtain a half-grid. Finally, we obtain the g-grate ω_g as a minor by deleting all vertices and edges in X and then all edges between X and $H \setminus Y$ except for the matching connecting S to X. ∎

Theorem 6.30. *Let Δ be a class of conjunctive queries with unbounded linked matching number. Then $\#\textsc{cp-Hom}(\Delta)$ is $\#\mathsf{A}[2]$-equivalent.*

Proof. Follows from the $\#\mathsf{A}[2]$-equivalence of $\#\textsc{cp-Hom}(\mathsf{Grates})$, given by Lemma 6.26, from the Excluded-Grate-Theorem (Theorem 6.29) as well as the minor reduction (Lemma 6.10). ∎

6.3.6 A Pentachotomy-Theorem

We are now in position to state the main result of this section, the full classification for counting answers to conjunctive queries. Note that Theorem 6.6 is subsumed by the full classification in the case of graphs. The general version, that is, the case of arbitrary logical signatures with bounded arity, is proved in Chapter 6.5.3.

Theorem 6.31. *Let Δ be a recursively enumerable class of minimal conjunctive queries.*

1. *If the treewidth of Δ and $\mathsf{contract}(\Delta)$ is bounded, then $\#\textsc{Hom}(\Delta)$ can be computed in polynomial time.*

2. *If the treewidth of Δ is unbounded and the treewidth of $\mathsf{contract}(\Delta)$ is bounded, then $\#\textsc{Hom}(\Delta)$ is $\mathsf{W}[1]$-equivalent.*

3. *If the treewidth of $\mathsf{contract}(\Delta)$ is unbounded and the dominating star size of Δ is bounded, then $\#\textsc{Hom}(\Delta)$ is $\#\mathsf{W}[1]$-equivalent.*

4. *If the dominating star size of Δ is unbounded, then $\#\textsc{Hom}(\Delta)$ is $\#\mathsf{W}[2]$-hard. Moreover, for any fixed query δ with $\mathsf{dss}(\delta) \geq 3$, the problem $\#\mathsf{Hom}(\delta \to \star)$ cannot be computed in time $O(n^{\mathsf{dss}(\delta)-\varepsilon})$ for any $\varepsilon > 0$ unless SETH fails.*

5. *If the linked matching number of Δ is unbounded, then $\#\textsc{Hom}(\Delta)$ is $\#\mathsf{A}[2]$-equivalent.*

Proof. The first two claims and the $\#\mathsf{W}[1]$-hardness in the third follow from Theorem 6.16. The $\#\mathsf{W}[1]$-easiness in the third claim follows from Theorem 6.19 and the fact that $\#\textsc{Hom}(\Delta)$ reduces to $\#\textsc{cp-Hom}(\Delta)$ as shown in the context of the normalization theorem in Appendix C. The fourth and fifth claim follow from Theorem 6.22 and Theorem 6.30, respectively, as well as from the fact that $\#\textsc{cp-Hom}(\Delta)$ reduces to $\#\textsc{Hom}(\Delta)$ as shown in Chapter 6.2.1. ■

Our classification leaves open the question whether every class Δ that has a bounded linked matching number is in fact $\#\mathsf{W}[2]$-easy; this question is related to some exotic parameterized complexity classes between $\#\mathsf{W}[2]$ and $\#\mathsf{A}[2]$.[8]

[8]The interested reader is encouraged to make themself familiar with the class $\mathsf{W}^{\mathsf{func}}[2]$ (see Chapt. 8 in [65]) and to observe that strengthening the classification as suggested would imply $\#\mathsf{W}[2] = \#\mathsf{W}^{\mathsf{func}}[2]$ or $\#\mathsf{A}[2] = \#\mathsf{W}^{\mathsf{func}}[2]$.

6.4 Quantum Queries

In this section, we extend our results to disjunctions of conjunctive queries, and to conjunctive queries with inequality constraints and negations on the free variables. We show in this section that both of these extensions are captured by considering abstract linear combinations of conjunctive queries. To this end, we first adapt the notion of quantum graphs to the setting of graphical conjunctive queries.

Definition 6.32. A *quantum query* Q is a formal linear combination of a finite number of graphical conjunctive queries. We write

$$Q = \sum_{i=1}^{\ell} \lambda_i \cdot (H_i, X_i)\,, \tag{6.12}$$

where all λ_i are non-zero rational numbers. The *support* of Q is the set $\mathsf{supp}(Q) = \{(H_i, X_i) \mid i \in \{1, \dots, \ell\}\}$. The number of homomorphisms extends to quantum graphs linearly, i.e., if Q is a quantum query as above and G is a simple graph, then we define

$$\#\mathsf{Hom}(Q \to G) = \sum_{i=1}^{\ell} \lambda_i \cdot \#\mathsf{Hom}(H_i, X_i \to G)\,. \tag{6.13}$$

In the subsequent sections we are going to collect for equivalent queries in a quantum query and hence consider the support to be a set of pairwise non-equivalent and minimal conjunctive queries. The structural parameters discussed in Chapter 6.3 then extend to quantum queries by taking the maximum over all queries in the support.

Definition 6.33. Let (H, X) and $(\hat{H}, \hat{X})$ be graphical conjunctive queries.

1. (H, X) *maps surjectively to* $(\hat{H}, \hat{X})$, written $(H, X) \geq (\hat{H}, \hat{X})$, if there is a surjective function $s : X \to \hat{X}$ that extends to a homomorphism, that is, which satisfies $s \in \mathsf{Hom}(H, X \to \hat{H})$. Let $\mathsf{Surj}(H, X \to \hat{H}, \hat{X})$ be the set of all surjective mappings $s : X \to \hat{X}$ that can be extended to a homomorphism from H to $\hat{H}$.

2. If $(H, X) \geq (\hat{H}, \hat{X}) \geq (H, X)$ holds, then we adopt the notation of Chen and Mengel [32] and say that the two queries are *renaming equivalent*. Moreover, (H, X) is a *minimal representative* if it has a lexicographically smallest pair $(|V(H)|, |E(H)|)$ among all queries that are renaming equivalent to (H, X). For each equivalence class, we arbitrarily fix one minimal representative. If (H, X) is the selected minimal representative of its class, we also call it *renaming minimal*.

It is clear that $\geq$ defines a partial order and so this notion of equivalence is indeed an equivalence relation. If $X = \hat{X} = \emptyset$ holds, then the notion specializes to homomorphic equivalence,[9] whereas for $X = V(H)$ and $\hat{X} = V(\hat{H})$, it specializes to isomorphism. It will turn out that renaming equivalence is identical to equivalence of conjunctive queries as introduced in Chapter 6.2.1. In what follows we will therefore omit "renaming" and only speak of equivalent and minimal queries. The next lemma generalizes the fact that all homomorphic cores of a graph are isomorphic.

Lemma 6.34. *If two minimal graphical conjunctive queries are equivalent, then they are isomorphic.*

Proof. Let (H, X) and $(\hat{H}, \hat{X})$ be minimal graphical conjunctive queries that are equivalent. By equivalence, we get bijective functions $s : X \to \hat{X}$ and $\hat{s} : \hat{X} \to X$ that can be extended to homomorphisms h and $\hat{h}$, respectively. Let F be the subgraph of $\hat{H}$ that is the image of h, that is we have $V(F) = h(V(H))$ and $E(F) = h(E(H))$. We claim that $F = \hat{H}$ must hold by minimality. Indeed, when $\hat{h}$ is restricted to the vertices of F, it must remain a homomorphism that extends $\hat{s}$, and so (H, X) and $(F, \hat{X})$ are equivalent. Minimality implies $|V(H)| = |V(F)|$ and $|E(H)| = |E(F)|$, and so h must have every vertex and edge of $\hat{H}$ in its image. Thus h is in fact an isomorphism between H and $\hat{H}$, which is what we claimed. ∎

We can easily express the number of all partial homomorphisms as a linear combination of the number of partial surjective homomorphisms.

Lemma 6.35. *For all graphical queries (H, X) and all graphs G, we have the following identity:*

$$\#\mathsf{Hom}(H, X \to G) = \sum_{Z \subseteq V(G)} \#\mathsf{Surj}(H, X \to G, Z)\,.$$

Proof. Every element $a \in \mathsf{Hom}(H, X \to G)$ has a unique set

$$Z = a(X) \subseteq V(G)\,,$$

such that a is surjective on Z. Thus the sets $\mathsf{Surj}(H, X \to G, Z)$ are disjoint for distinct Z, and their union is $\mathsf{Hom}(H, X \to G)$, so the claimed identity follows. ∎

In the following lemma, we show that the functions $\#\mathsf{Hom}(H, X \to \star)$ are linearly independent for all minimal conjunctive queries (H, X). It was proved implicitly by Chen and Mengel [32]. The following, explicit proof is due to Holger Dell;[10] we include it only for completeness.

[9]Two graphs F and H are homomorphically equivalent if there exists a homomorphism from F to H and a homomorphism from H to F.

[10]See also Lemma 34 in [49].

Lemma 6.36. *Let $k \geq 0$ and let $\mathcal{M}$ be the (finite) set of all minimal graphical conjunctive queries with at most k vertices, and let $\mathcal{G}$ be the finite set of all (unlabeled) simple graphs with at most k^k vertices.*

(1) Let L be the $(\mathcal{M} \times \mathcal{M})$-matrix with

$$L[(H,X),(\hat{H},\hat{X})] = \#\mathsf{Surj}(H,X \to \hat{H},\hat{X})\,.$$

If we linearly sort $\mathcal{M}$ consistent with the partial order "$\leq$", then L is a lower-triangular matrix whose diagonal entries are positive integers.

(2) Let B be the $(\mathcal{M} \times \mathcal{G})$-matrix where $B[(\hat{H},\hat{X}),G]$ is the number of sets $Z \subseteq V(G)$ such that $(\hat{H},\hat{X})$ and (G,Z) are equivalent. Then B has full rank.

(3) Let A be the $(\mathcal{M} \times \mathcal{G})$-matrix with

$$A[(H,X),G] = \#\mathsf{Hom}(H,X \to G)\,.$$

Then $A = LB$ holds and thus A has full rank.

Proof. First we discuss how to sort the elements of $\mathcal{M}$. Since $\mathcal{M}$ only contains minimal queries, any two distinct elements of $\mathcal{M}$ are not equivalent, and thus $(H,X) \not\geq (\hat{H},\hat{X})$ or $(H,X) \not\leq (\hat{H},\hat{X})$ holds. Thus we can linearly order $\mathcal{M}$ in such a way, that $(H,X) \not\geq (\hat{H},\hat{X})$ holds whenever (H,X) occurs before $(\hat{H},\hat{X})$ in the order, and this is the order we choose.

(1). The identity function $s : X \to X$ is clearly an element of

$$\mathsf{Surj}(H,X \to H,X)\,,$$

so all diagonal entries of L are positive integers. Now let (H,X) and $(\hat{H},\hat{X})$ be distinct elements of $\mathcal{M}$ such that (H,X) occurs before $(\hat{H},\hat{X})$ in the linear order and so $(H,X) \not\geq (\hat{H},\hat{X})$ holds. By definition, this means that no surjective function $X \to \hat{X}$ extends to a homomorphism from H to $\hat{H}$, which implies $L[(H,X),(\hat{H},\hat{X})] = 0$. Thus, L is lower-triangular.

(2). The proof is similar to the proof of Lemma 6.14. To prove the claim, we show that each $(\hat{H},\hat{X}) \in \mathcal{M}$ has a linear combination of the columns $\sum_z \lambda_z \cdot B[\star, G^z] = 1$ such that $\sum_z \lambda_z \cdot B[(H,X),G^z] \neq 0$ holds if and only if $(H,X) = (\hat{H},\hat{X})$.

For each $(\hat{H},\hat{X})$ and each $z \in \{1,\ldots,k\}^{\hat{X}}$, we construct graphs G^z as follows: Start from $G := \hat{H}$ and clone each vertex $v \in \hat{X}$ exactly $z_v - 1$ times (i.e. replace it with an independent set of size z_v where each vertex has the same neighborhood as v). Note that G^z is $\hat{H}$-colored, and let

$$c \in \mathsf{Hom}(G^z \to \hat{H})$$

be the coloring. Now recall that $B[(H,X),G^z]$ counts the sets $Z' \subseteq V(G^z)$ such that (G^z,Z') and (H,X) are equivalent. Clearly $|Z'| = |X|$ must hold for this to be the case.
We call Z' *proper* if $c(Z') = \hat{X}$ holds, and *improper* otherwise. Moreover, we say that Z' is H-equivalent if (H,X) and (G^z,Z) are equivalent. We have:

$$B[(H,X),G^z] = \#\{\text{proper } H\text{-equivalent } Z'\} + \#\{\text{improper } H\text{-equivalent } Z'\}\,.$$

If Z' is proper, then (G^z,Z') is equivalent to $(\hat{H},\hat{X})$ by construction of G^z. Thus if a proper H-equivalent Z' exists, then $(\hat{H},\hat{X}) = (H,X)$ holds and the number of proper H-equivalent Z' in G^z is equal to

$$\prod_{v \in \hat{X}} z_v\,.$$

On the other hand, if $(\hat{H},\hat{X}) \neq (H,X)$, then the number of proper H-equivalent Z' is equal to zero. In any case, the number of improper H-equivalent Z' is a polynomial in the z_v variables which however does not contain the monomial $\prod_{v\in\hat{X}} z_v$. By multivariate Lagrange interpolation, there is a linear combination

$$\sum_z \lambda_z B[(H,X),G^z]$$

which is equal to the coefficient of the monomial $\prod_{v\in\hat{X}} z_v$. This monomial is zero if and only if $(H,X) \neq (\tilde{H},\tilde{X})$.

(3). The fact that $A = LB$ holds follows directly from Lemma 6.35 by collecting terms for equivalent (G,Z). Since L is invertible and B has full rank, this also implies that A has full rank. ∎

We point out, that Lemma 6.36 implies that renaming equivalence of two conjunctive queries is an explicit notion for equivalence. Note that the following was also shown by Chen and Mengel [32] with a more complicated proof.

Corollary 6.37. *Two conjunctive queries are renaming equivalent if and only if they are equivalent.*

Proof. The forward implication is immediate and the reverse follows from the third item of Lemma 6.36. To see this, we observe that the full rank of A certainly implies that its row vectors are pairwise different. ∎

6.4.1 Complexity Monotonicity revisited

We will now lift complexity monotonicity from quantum graphs (Theorem 3.13) to the more general case of quantum queries. The proof is completely analogous.

Lemma 6.38 (Complexity monotonicity, implicit in [32]). *Let Q be a quantum query. There is an oracle algorithm $\mathbb{A}$ that is given G as input and oracle access to the $\#\mathsf{Hom}(Q \to \star)$, and computes $\#\mathsf{Hom}(H, X \to G)$ for all $(H, X) \in \mathsf{supp}(Q)$ in time $t(|Q|) \cdot n$, where $n = |V(G)|$ and t is a computable function. Furthermore, every oracle query $\#\mathsf{Hom}(Q \to G')$ satisfies $|V(G')| \leq t(|Q|) \cdot n$.*

Proof. Let k be the largest number of vertices among the graphs in the support of Q and let $\mathcal{M}$ be the set from Lemma 6.36. Recall that the tensor product of two graphs is denoted by $G \times F$ and note that, similarly to Fact 3.11, we have that

$$\mathsf{Hom}(H, X \to G \times F) = \mathsf{Hom}(H, X \to G) \cdot \mathsf{Hom}(H, X \to F)\,.$$

Let $Q = \sum_{H \in \mathcal{M}} \lambda_H H$ and write $x_H = \lambda_H \cdot \mathsf{Hom}(H, X \to G)$. Moreover, set $b_F = \mathsf{Hom}(Q \to G \times F)$ and let A be the matrix from Lemma 6.36. Then we have $x^T A = b$. To compute the vector b, we simply query the oracle, and the queries have the required size bound. The matrix A and in fact its inverse A^{-1} can be hard-wired into the algorithm. Then $x^T = bA^{-1}$ holds, and we can compute the values x_H/λ_H for H in the support of Q in the time required. ∎

Theorem 6.39. *Let Δ be a recursively enumerable class of quantum queries and let $\hat{\Delta}$ be the set of all minimal conjunctive queries that are contained in the support of some query in Δ.*

1. *If the treewidth of $\hat{\Delta}$ and $\mathsf{contract}(\hat{\Delta})$ is bounded, then $\#\mathrm{HOM}(\Delta)$ is fixed-parameter tractable.*

2. *If the treewidth of $\hat{\Delta}$ is unbounded and the treewidth of $\mathsf{contract}(\hat{\Delta})$ is bounded, then $\#\mathrm{HOM}(\Delta)$ is $\mathsf{W}[1]$-equivalent.*

3. *If the treewidth of $\mathsf{contract}(\hat{\Delta})$ is unbounded and the dominating star size of $\hat{\Delta}$ is bounded, then $\#\mathrm{HOM}(\Delta)$ is $\#\mathsf{W}[1]$-equivalent.*

4. *If the dominating star size of $\hat{\Delta}$ is unbounded, then $\#\mathrm{HOM}(\Delta)$ is $\#\mathsf{W}[2]$-hard. Moreover, for any fixed quantum query δ with $\mathsf{dss}(\delta) \geq 3$, the problem $\#\mathsf{Hom}(\delta \to \star)$ cannot be computed in time $O(n^{\mathsf{dss}(\delta)-\varepsilon})$ for any $\varepsilon > 0$ unless SETH fails.*

5. *If the linked matching number of $\hat{\Delta}$ is unbounded, then $\#\mathrm{HOM}(\Delta)$ is $\#\mathsf{A}[2]$-equivalent.*

Proof. Follows from the classification for conjunctive queries (Theorem 6.31) and the complexity monotonicity property. ■

We remark that, in case of graphs, the classification for quantum queries implies both, Theorem 6.5 and Theorem 6.7, if we can express existential and universal positive queries with inequalities and non-monotone constraints over the free variables as quantum queries. This is proved in the subsequent Chapters 6.4.2-6.4.4. Again the general version for arbitrary logical signatures with bounded arity is deferred to Chapter 6.5.3.

6.4.2 Conjunctive Queries with Inequalities

In what follows, we will generalize Theorem 6.31 to conjunctive queries that may contain inequalities over free variables[11]. In particular we will show that the support of the resulting quantum query can be given explicitly. Answers to conjunctive queries with inequalities are modeled via partially injective homomorphisms. We remark that the subsequent notions and proofs are *completely similar* to our treatment of partially injective homomorphisms in Chapter 4.1. For this reason we will keep this section concise and refer the reader to the detailed exposition in Chapter 4.1 for the entirety of the technicalities.

Definition 6.40. A *conjunctive query with inequalities over the free variables* is a triple (H, I, X) where (H, X) is a conjunctive query and I is an irreflexive and symmetric relation $I \subseteq X^2$. We say that I is a set of inequalities and write $\{x, x'\} \in I$ whenever $(x, x') \in I$.

Given a graph G and a conjunctive query with inequalities (H, I, X), an assignment $a : X \to V(G)$ is an answer to (H, I, X) in G if and only if a is an answer to (H, X) and, additionally, for every inequality $\{x, x'\} \in I$, it holds that $a(x) \neq a(x')$. Formally, we define the set of answers to (H, I, X) in terms of partially injective partial homomorphisms

$$\mathsf{PartInj}(H, I, X \to G) := \{a \in \mathsf{Hom}(H, X \to G) \mid \forall (x, x') \in I : a(x) \neq a(x')\}\,.$$

Given a conjunctive query (H, X) and a set $\sigma \subseteq I$ the *quotient query* $(H/\sigma, X/\sigma)$ is obtained by identifying every pair of vertices x and $\hat{x}$ as a single vertex for every inequality $(x, \hat{x}) \in \sigma$. Self-loops are kept and multiple edges are deleted. We point out that it is possible that the contraction of all pairs in σ might also contract vertices x and $\hat{x}$ that are not contained in σ. Consider for example $\sigma = \{\{x_1, x_2\}, \{x_2, x_3\}\}$. Then contracting $\{x_1, x_2\}$ and $\{x_2, x_3\}$ will also contract x_1 and x_3.

[11]At the end of this subsection, we argue why a similar result which also takes inequalities into account that may contain quantified variables, would require to solve a long standing open problem in parameterized complexity theory.

Theorem 6.41. *Let $\chi = (H, I, X)$ be a conjunctive query with inequalities over the free variables. Then there exists a quantum query $Q[\chi]$ such that*

$$\#\mathsf{PartInj}(H, I, X \to \star) = \#\mathsf{Hom}(Q[\chi] \to \star)\,.$$

Furthermore, the mapping $\chi \mapsto Q[\chi]$ is computable and the support of $Q[\chi]$ is, up to equivalence, the set of all contracted queries $(H/\sigma, X/\sigma)$ without self-loops where σ is a subset of I.

In other words, given $\chi = (H, I, X)$, we can contract arbitrary variables in (H, X) that are connected by an inequality in I. Then Theorem 6.41 guarantees that a minimal equivalent of the resulting query is contained in the support of the quantum query $Q[\chi]$.

We remark that a general theorem in the above form that also includes inequalities over quantified variables remains elusive, as this would require to completely understand the subgraph decision problem, which is one of the most famous open problems in parameterized complexity (see e.g [57, Chapt. 33.1]). In terms of conjunctive queries with inequalities, the subgraph decision problem can be formulated by a query without free variables and with all inequalities over the quantified variables. Then the empty assignment is in the set of solutions if and only if there is an injective homomorphism, that is, a subgraph embedding from the quantified variables to the host graph.

Proof (of Theorem 6.41). Let G be a graph and let (H, X, I) be a conjunctive query with inequalities. In particular, as the inequalities are only over the free variables, we can use Möbius inversion similarly as in Chapter 4.1 and obtain that

$$\#\mathsf{PartInj}(H, X, I \to G) = \sum_{\rho \in L} \mu(\emptyset, \rho) \cdot \#\mathsf{Hom}(H/\rho, X/\rho \to G)\,,$$

where μ is the Möbius function of the lattice of flats L of the graphic matroid $M(X, I)$. Now let $[\![H_1, X_1]\!], \dots, [\![H_k, X_k]\!]$ be the equivalence classes of the set

$$\{(H/\rho, X/\rho) \mid \rho \in L \ \wedge \ H/\rho \text{ contains no self-loops}\}$$

with minimal representatives. Then, we can define the desired quantum query to be

$$Q[\chi] := \sum_{i=1}^{k} \lambda_i \cdot (H_i, X_i) \qquad \text{where} \qquad \lambda_i = \sum_{\substack{\rho \in L \\ (H/\rho, X/\rho) \ \sim \ (H_i, X_i)}} \mu(\emptyset, \rho)\,.$$

It remains to show that for all $i \in [k]$ we have that $\lambda_i \neq 0$.

To this end, we observe that

$$(H/\rho, X/\rho) \sim (H/\sigma, X/\sigma) \ \Rightarrow\ \#X/\rho = \#X/\sigma\,,$$

and hence that ρ and σ have the same number of blocks with respect to the graphic matroid $M(X,I)$. Therefore, $\mathsf{rk}(\rho) = \mathsf{rk}(\sigma) =: r_i$ and therefore, by Rota's NBC Theorem (Theorem 2.26), we have that $\mathsf{sign}(\lambda_i) = (-1)^{r_i}$. Consequently, $\lambda_i \neq 0$.

Finally, we have that for every *subset* σ of I, there exists a flat ρ of $M(X,I)$ such that $(H/\rho, X/\rho) = (H/\sigma, X/\sigma)$. In particular it can be observed that in this case ρ is the closure $\mathsf{cl}(\sigma)$ of σ. This concludes the proof. ∎

6.4.3 Positive Formulas with Inequalities

In this subsection we will lift the classification once more, namely to existential and universal positive formulas with inequalities over the free variables. Our goal is hence to find quantum queries Q that allow us to express the number of solutions to the more general queries as $\#\mathsf{Hom}(Q \to \star)$. Recall that an existential first-order formula is of the form

$$\psi = x_1 \dots x_k \exists y_1 \dots \forall y_\ell : \psi'\,,$$

and that a universal first-order formula is of the form

$$\theta = x_1 \dots x_k \forall y_1 \dots \forall y_\ell : \theta'\,,$$

where both, ψ' and θ' are quantifier-free. An existential first-order formula is called *existential positive* if additionally ψ' does not contain negations. Similarly, a universal first-order formula is called *universal positive* if θ' does not contain negations. We write Σ_1^+ and Π_1^+ for the sets of all existential and universal positive formulas, respectively. The following is due to Chen and Mengel — we state their result in terms of quantum queries.

Theorem 6.42 ([32]). *Let $\psi \in \Sigma_1^+$ be an existential positive formula. Then there exists a quantum query $Q[\psi]$ such that for every graph G it holds that $\#\psi(G) = \#\mathsf{Hom}(Q[\psi] \to G)$. Furthermore, the mapping $\psi \mapsto Q[\psi]$ is computable.*

This result can easily be extended to universal positive queries.

Corollary 6.43. *Let $\theta \in \Pi_1^+$ be a universal positive formula. Then there exists a quantum query $Q[\theta]$ such that for every graph G it holds that*

$$\#\theta(G) = \#\mathsf{Hom}(Q[\theta] \to \overline{G})\,,$$

where $\overline{G}$ is the complement graph of G. Furthermore, the mapping $\theta \mapsto Q[\theta]$ is computable.

Proof. Let $\theta = x_1, \dots x_k \forall y_1, \dots y_\ell : \theta'$. Without loss of generality we can assume that $\theta' = \bigvee_{i=1}^{m_1} \bigwedge_{j=1}^{m_2} Ev_iv_j$. For every graph G with n vertices, we have that

$$\#\theta(G) = n^k - \#\{a : X \to V(G) \mid \exists h : X \cup Y \to V(G) : h|_X = a \wedge G \not\models_h \theta'\},$$

i.e., we can count all assignments a that can be extended to an assignment h that does not satisfy θ' and subtract this number from the number n^k of all assignments from X to $V(G)$. Now, using DeMorgan's Law, it holds that

$$G \not\models_h \theta' \Leftrightarrow \overline{G} \models_h \overline{\theta'},$$

where

$$\overline{\theta'} = \bigwedge_{i=1}^{m_1} \bigvee_{j=1}^{m_2} Ev_iv_j.$$

Finally, we let

$$\overline{\theta} := x_1, \dots x_k \exists y_1, \dots y_\ell : \overline{\theta'},$$

and obtain

$$\begin{aligned} \#\theta(G) &= n^k - \#\{a : X \to V(G) \mid \exists h : X \cup Y \to V(G) : h|_X = a \wedge G \not\models_h \theta'\} \\ &= n^k - \#\overline{\theta}(\overline{G}) \end{aligned}$$

Since $\overline{\theta}$ is an existential positive formula, we can apply Theorem 6.42 to compute $Q[\overline{\theta}]$. Furthermore, it holds that $\#\mathsf{Hom}(\mathsf{IS}_k \to \overline{G}) = n^k$, where IS_k is the independent set of size k, that is, the graph consisting of vertices $[k]$ without edges. We conclude by setting

$$Q[\theta] := (\mathsf{IS}_k, [k]) - Q[\overline{\theta}],$$

and collecting for equivalent conjunctive queries in the support; note that the query $(\mathsf{IS}_k, [k])$ might a priori be contained in the support of $Q[\overline{\theta}]$. ∎

Remark 6.44 (Isolated vertices). Observe that the graph IS_k does not correspond directly to a conjunctive query; the same holds true for every graph with isolated vertices. For this reason, we allow that conjunctive queries may be equipped with additional free variables. We write

$$\varphi = x_1, \dots, x_k : \psi$$

to equip a conjunctive query ψ with *new* free variables $x_1, \dots, x_k$. Then we have that $\#\varphi(G) = |V(G)|^k \cdot \#\psi(G)$.

Now consider (existential or universal) positive formulas φ that are equipped with a set I of inequalities between free variables. Similarly to conjunctive queries with inequalities, we define

$$\varphi_I(G) := \{\beta \in \varphi(G) \mid \forall \{x, x'\} \in I : \beta(x) \neq \beta(x')\}.$$

Furthermore, we extend the notion of a quotient query to positive formulas as well. To this end, let J be a subset of the inequalities I. We define φ/J as the formula obtained from φ by renaming every variable x' with x whenever we have that $\{x,x'\} \in J$. We observe that

$$\#\varphi/J(G) = \#\{\beta \in \varphi(G) \mid \forall \{x,x'\} \in J : \beta(x) = \beta(x')\}\,. \tag{6.14}$$

We will now use the principle of inclusion and exclusion to express $\#\varphi_I(\star)$ as a linear combination of terms $\#\varphi/J(\star)$ for $J \subseteq I$. We point out that it is also possible to prove the subsequent theorem with Möbius inversion over the lattice of flats of the matroid induced by I as in case of conjunctive queries with inequalities. However, the terms $\#\varphi/J(\star)$ will be expressed by the quantum queries given by Theorem 6.42 and Corollary 6.43, and the support of those quantum queries cannot be given explicitly. For this reason there is no advantage in relying on the NBC Theorem to prove that the coefficients of the first transformation

$$\#\varphi_I(\star) = \sum_J v_J \cdot \#\varphi/J(\star)$$

do not cancel, as the cancellation might occur in the second transformation

$$\#\varphi/J(\star) = \#\mathsf{Hom}(Q[\varphi/J] \to \star)\,.$$

Consequently, we will not be able to explicitly give the support of the quantum graph $Q[\varphi, I]$ that satisfies

$$\#\mathsf{Hom}(Q[\varphi, I] \to \star) = \#\varphi_I(\star)\,,$$

regardless on whether we invoke inclusion-exclusion or Möbius inversion.[12] We decided for the former as it allows for a cleaner proof.

Theorem 6.45. *Let φ be an existential or universal positive formula and let I be a set of inequalities I over the free variables of φ. Then there exists a quantum query $Q[\varphi, I]$ such that for every graph G it holds that*

$$\#\varphi_I(G) = \#\mathsf{Hom}(Q[\varphi, I] \to G)\,.$$

Furthermore, the mapping $(\varphi, I) \mapsto Q[\varphi, I]$ is computable.

[12]Indeed, using the boolean expansion formula for the Möbius function over lattices of flats [141, Proposition 7.1.4], it can be shown that both principles are essentially equivalent.

Proof. We have that for all graphs G

$$\begin{aligned}
\#\varphi_I(G) &= \#\{\beta \in \varphi(G) \mid \forall\{x,x'\} \in I : \beta(x) \neq \beta(x')\} \\
&= \#\left(\varphi(G) \setminus \bigcup_{\{x,x'\}\in I} \{\beta \in \varphi(G) \mid \beta(x) = \beta(x')\}\right) \\
&= \sum_{J\subseteq I} (-1)^{\#J} \cdot \#\{\beta \in \varphi(G) \mid \forall\{x,x'\} \in J : \beta(x) = \beta(x')\} \\
&= \sum_{J\subseteq I} (-1)^{\#J} \cdot \#\varphi/J(G)\,,
\end{aligned}$$

where the third equality is inclusion-exclusion (Theorem 2.27) and the last equality is (6.14). Now if φ is existential positive, then so is φ/J for all $J \subseteq I$. We hence use Theorem 6.42 and collect for equivalent conjunctive queries; if φ is universal positive we proceed similarly with Corollary 6.43. ∎

6.4.4 Non-monotone Constraints over Free Variables

Last but not least we will lift the classification theorem to existential and universal positive queries with inequalities over the free variables that additionally may contain non-monotone constraints of the form $\neg Ex\hat{x}$ over free variables. The idea is quite simple: Just perform inclusion-exclusion over the non-monotone constraints. Unfortunately, this requires us to circumvent the following two cumbersome technicalities: First, it is possible that some of the free variables *only* occur in non-monotone constraints. This issue will be dealt with by using Remark 6.44. Second, we have to guarantee that a transformation of existential or universal positive queries with non-monotone constraints over free variables to a linear combination of conjunctive queries does not create queries that contain non-monotone constraints over quantified variables. This latter issue will be dealt with by taking a closer into the proof of Theorem 6.42 by Chen and Mengel [32].

We start by considering conjunctive queries with non-monotone constraints over the free variables. For technical reasons (see Remark 6.44), given a formula φ, a set J of atoms containing only variables that are new or free in φ and a set $\mathcal{V} = \{v_1, \dots, v_k\}$ disjoint from all variables in φ and J, we define

$$[\varphi \wedge J]^{\mathcal{V}} := v_1, \dots, v_k : \varphi \wedge \bigwedge_{a\in J} a\,.$$

We write $[\varphi \wedge J]$ if $\mathcal{V}$ is empty and $[\varphi]^{\mathcal{V}}$ if J is empty. Now let G be a graph and let φ be a conjunctive query with non-monotone constraints

$$\neg S = \{\neg s_1, \neg s_2, \dots, \neg s_\ell\} \tag{6.15}$$

where each s_i is an atom $Ex_i\hat{x}_i$ for some free variables x_i and x_i'.

As φ is a conjunctive query, there is a subquery δ without non-monotone constraints such that $\varphi = [\delta \wedge \neg S]$. Now observe that

$$\varphi(G) = [\delta \wedge \neg S](G) = \left\{ a \in [\delta]^{\mathcal{V}(\neg S \setminus \delta)}(G) \;\middle|\; \bigwedge_{i=1}^{\ell} \{a(x_i), a(\hat{x}_i)\} \notin E(G) \right\}, \tag{6.16}$$

where $\mathcal{V}(\neg S \setminus \delta)$ is the set of variables occurring only in $\neg S$ and not δ. In other words, (6.16) states that the assignments satisfying φ are precisely those assignments that satisfy δ and all non-monotone constraints in $\neg S$. As, however, it might be possible that there are free variables in φ that only occur in the non-monotone part $\neg S$, we have to extend δ by those variables.

Again, we will use the principle of inclusion and exclusion to first get rid of the non-monotone constraints and then build up on the prior transformations to quantum queries.

Lemma 6.46. *Let $\varphi = [\delta \wedge \neg S]$ be a conjunctive query with non-monotone constraints $\neg S$ as given by* (6.15)*. Then we have that*

$$\#\varphi(\star) = \sum_{J \subseteq S} (-1)^{\#J} \cdot \#[\delta \wedge J]^{\mathcal{V}(S,\delta,J)}(\star),$$

where $S := \{s_1, s_2, \dots, s_\ell\}$ and $\mathcal{V}(S,\delta,J)$ is the set of all variables occurring in S but neither in δ nor in J.

Proof. Let G be a graph. Using inclusion-exclusion (Theorem 2.27), we obtain that

$$\begin{aligned}
\#\varphi(G) &= \#[\delta \wedge \neg S](G) \\
&= \#\left\{ a \in [\delta]^{\mathcal{V}(\neg S \setminus \delta)}(G) \;\middle|\; \bigwedge_{i=1}^{\ell} \{a(x_i), a(\hat{x}_i)\} \notin E(G) \right\} \\
&= \#[\delta]^{\mathcal{V}(\neg S \setminus \delta)}(G) - \#\left\{ a \in [\delta]^{\mathcal{V}(\neg S \setminus \delta)}(G) \;\middle|\; \bigvee_{(Ex\hat{x}) \in S} \{a(x), a(\hat{x})\} \in E(G) \right\} \\
&= \sum_{J \subseteq S} (-1)^{\#J} \cdot \#\left\{ a \in [\delta]^{\mathcal{V}(\neg S \setminus \delta)}(G) \;\middle|\; \bigwedge_{(Ex\hat{x}) \in J} \{a(x), a(\hat{x})\} \in E(G) \right\} \\
&= \sum_{J \subseteq S} (-1)^{\#J} \cdot \#[\delta \wedge J]^{\mathcal{V}(S,\delta,J)}(G).
\end{aligned}$$

■

Our next goal is to generalize to existential and universal positive formulas with inequalities and non-monotone constraints over the free variables. To this end, we wish to invoke Theorem 6.45. However, the statement of the latter theorem does formally not apply to formulas with non-monotone constraints.

To circumvent this issue, we will just add a the relation symbol $\overline{E}$ to the signature of graphs — we will argue in Chapter 6.5 that all results for the signature of graphs readily extend to arbitrary signatures of bounded arity.

Now let ψ be an existential or universal positive formula over the signature of graphs with non-monotone constraints of the form $\neg Ex\hat{x}$. The formula $\psi_{\uparrow}$ is obtained from ψ by substituting every atom $\neg Ex\hat{x}$ by $\overline{E}x\hat{x}$, where $\overline{E}$ is a new relation symbol of arity 2. Consequently, the signature of $\psi_{\uparrow}$ is $\tau = (E, \overline{E})$. Similarly, given a graph G, that is, a structure over the signature (E), we let $G_{\uparrow}$ be the following structure over signature τ: The vertices $V(G_{\uparrow})$ of $G_{\uparrow}$ are precisely the vertices of G and a pair $(x, \hat{x})$ is in $E(G_{\uparrow})$ if and only if $\{x, \hat{x}\}$ is an edge of G. Furthermore, a pair $(x, \hat{x})$ is in $\overline{E}(G_{\uparrow})$ if and only if $\{x, \hat{x}\}$ is not an edge of G.

The operation $\downarrow$ is defined analogously: Given an existential or universal positive formula φ over the signature τ, we obtain the formula $\varphi_{\downarrow}$ from φ by substituting every atom $\overline{E}x\hat{x}$ by a non-monotone constraint $\neg Ex\hat{x}$. Consequently, the signature of $\varphi_{\downarrow}$ is (E). Similarly, given a structure G over signature τ, we obtain a graph $G_{\downarrow}$ without self-loops from G by taking the same set of vertices and adding an edge $\{x, \hat{x}\}$ to $E(G_{\downarrow})$ if and only if $x \neq \hat{x}$ and $(x, \hat{x}) \in E(G)$. The following is immediate.

Fact 6.47. *We have that*

(1) $\psi_I(G) = \psi_{\uparrow I}(G_{\uparrow})$ for every graph G without self-loops, existential or universal positive formula ψ over the signature of graphs, and set of inequalities I over the free variables of ψ,

(2) $\varphi_I(G) = \varphi_{\downarrow I}(G_{\downarrow})$ for every structure G and existential or universal positive formula φ over the signature $(E, \overline{E})$, and for every set of inequalities I over the free variables of φ,

(3) and $G_{\uparrow\downarrow} = G$ for every graph G without self-loops.

Theorem 6.48. *Let ψ be an existential or universal positive formula with non-monotone constraints over the free variables and let I be a set of inequalities over the free variables of ψ. There exists a quantum query $Q[\psi, I]$ satisfying that*

$$\#\psi_I(\star) = \#\mathsf{Hom}(Q[\psi, I] \to \star)\,.$$

Furthermore, the mapping $(\psi, I) \mapsto Q[\psi, I]$ is computable.

Proof. We have that for every graph G

$$\begin{aligned}
\#\psi_I(G) &= \#\psi_{\uparrow I}(G_\uparrow) && (6.17)\\
&= \#\mathsf{Hom}(Q[\psi_\uparrow, I] \to G_\uparrow) && (6.18)\\
&= \sum_{\varphi\in\mathsf{supp}(Q[\psi_\uparrow,I])} \lambda_\varphi \cdot \#\varphi(G_\uparrow) && (6.19)\\
&= \sum_{\varphi\in\mathsf{supp}(Q[\psi_\uparrow,I])} \lambda_\varphi \cdot \#\varphi_\downarrow(G)\,, && (6.20)
\end{aligned}$$

where (6.17) holds by Fact 6.47 and (6.18) holds by the generalized version of Theorem 6.45 that works for arbitrary signatures. Furthermore, (6.19) holds by definition of a quantum query and (6.20) is again due to Fact 6.47. Now consider the conjunctive queries $\varphi_\downarrow$: Those formulas might contain non-monotone constraints and we wish to get rid of them by invoking Lemma 6.46. However, this requires that the non-monotone constraints are only over free variables of $\varphi_\downarrow$. Equivalently, φ must satisfy that each atom $\overline{E}x\hat{x}$ in only over free variables. To this end, we observe that the quantum query $Q[\psi_\uparrow, I]$ in (6.18) is obtained in two steps; consult the proof of Theorem 6.45.

In the first step $\#\psi_{\uparrow I}(G_\uparrow)$ is transformed into a linear combination of quotient formulas $\#\psi_\uparrow/J$ for $J \subseteq I$. As, by assumption, all non-monotone constraints of ψ are over free variables, it hence holds that all atoms $\overline{E}x\hat{x}$ in $\#\psi_\uparrow/J$ are over free variables as well — recall that the quotient only contracts free variables.

Note that all quotient formulas $\#\psi_\uparrow/J$ are universal or existential positive. Now, in the second step, depending on whether ψ is existential or universal positive, the formulas $\#\psi_\uparrow/J$ are transformed to a linear combination of conjunctive queries by either Theorem 6.42 or Corollary 6.43. The latter does not change the free variables and also relies on Theorem 6.42. Consequently, we have to guarantee that the construction of the quantum queries $Q[\#\psi_\uparrow/J]$ yields as constituents only conjunctive queries satisfying that every atom $\overline{E}x\hat{x}$ is only over free variables. Now taking a look into the proof of Chen and Mengel [32, Section 4 and 5.3] reveals that they perform inclusion and exclusion over conjunctions of subformulas of $\psi_\uparrow/J$. Consequently, no atom $\overline{E}x\hat{x}$ such that either x or $\hat{x}$ is quantified, can be constructed. This allows us to continue from (6.20) by invoking Lemma 6.46:

$$\begin{aligned}
\#\psi_I(G) &= \sum_{\varphi\in\mathsf{supp}(Q[\psi_\uparrow,I])} \lambda_\varphi \cdot \#\varphi_\downarrow(G)\\
&= \sum_{\varphi\in\mathsf{supp}(Q[\psi_\uparrow,I])} \lambda_\varphi \cdot \sum_{J\subseteq S} (-1)^{\#J} \cdot \#[\delta \wedge J]^{\mathcal{V}(S,\delta,J)}(G)\,,
\end{aligned}$$

where $\varphi\downarrow = [\delta \wedge \neg S]$ and $\mathcal{V}(S, \delta, J)$ are as in Lemma 6.46. Finally, we collect for equivalent conjunctive queries and obtain the quantum query $Q[\psi, I]$. ∎

We are now able to proof Theorem 6.7 in case of the signature of graphs.

Proof (of Theorem 6.7). Given a family Φ of existential or universal positive formulas with non-monotone constraints and inequalities over the free variables, we let Δ be the set of the corresponding quantum queries as given by Theorem 6.48. Then the problems $\#\text{P-MC}(\Phi)$ and $\#\textsc{Hom}(\Delta)$ are equivalent. The claim follows hence by Theorem 6.39. ■

6.5 Generalization to Structures

In this section we are going to generalize all results that have been proved for graphs to logical structures. Similarly as to graphs, every conjunctive query φ over signature $\tau = (E_i)_{i\in[\ell]}$ with free variables X and quantified variables Y is associated with a pair (H, X) where H is a structure over the signature τ. Here, $V(\mathrm{H}) = X \cup Y$ and for every $i \in [\ell]$ we add a vector

$$(z_j)_{j\in[\mathsf{a}_i]} \in (X \cup Y)^{\mathsf{a}_i}$$

to $E_i(\mathrm{H})$ if and only if $E_i(z_j)_{j\in[\mathsf{a}_i]}$ is an atom of φ. Observe that for all structures G over τ it holds that $\varphi(\mathrm{G}) = \mathsf{Hom}(\mathrm{H}, X \to \mathrm{G})$. We adapt the notion for graphs and call (H, X) a graphical conjunctive query. The definitions of H-colored structures, color-prescribed and colorful homomorphism as well as of quantum queries transfers in the canonical way from graphs to structures.

Now let Δ be a set of first-order formulas. We say that Δ has *bounded arity* if there is a constant $C \in \mathbb{N}$ such that every signature of some formula in Δ has arity at most C. We assume all classes of formulas in this chapter to have bounded arity.

6.5.1 Reduction from the Gaifman Graph

In what follows we prove that counting color-preserving partial homomorphisms from a structure is at least as hard as counting color-preserving partial homomorphisms from its Gaifman graph.

Lemma 6.49. *Let (H, X) be a graphical conjunctive query of signature τ. Then there exists a deterministic algorithm $\mathbb{A}$ equipped with oracle access to $\#\mathsf{cp\text{-}Hom}(\mathrm{H}, X \to \star)$ that computes $\#\mathsf{cp\text{-}Hom}(\mathbb{G}(\mathrm{H}), X \to \star)$. Furthermore, $\mathbb{A}$ runs in time $O(f(\mathrm{H}) \cdot n^{\mathsf{a}(\tau)})$ for some computable function f.*

Proof. We will first provide the intuition behind the proof by considering the following restriction on H. We assume that $\tau = (E_1)$ and let $\mathsf{a} = \mathsf{a}_1$. Furthermore, we assume that H does not have any tuple in $E_1(\mathrm{H})$ that contains a multiple occurrence of the same vertex. Now given a $\mathbb{G}(\mathrm{H})$-colored graph G for which we want to compute $\#\mathsf{cp\text{-}Hom}(\mathbb{G}(\mathrm{H}), X \to G)$, we can construct an H-colored structure G' from G as follows. For every $\vec{u} = (u_1, \ldots, u_{\mathsf{a}}) \in E_1(\mathrm{H})$ we search all cliques in G of size a that are colored with $u_1, \ldots, u_{\mathsf{a}}$. Every clique $\vec{c} = (c_1, \ldots, c_{\mathsf{a}})$ satisfying that c_i has color u_i is then added to G as a tuple in E_1. If this is done for all $\vec{u} \in E_1(\mathrm{H})$ we delete all former edges of G. It is easy to see that the resulting structure G' is H-colored, except for the case that there was an $\vec{u} \in E_1(\mathrm{H})$ for which there was no corresponding clique in G. In this case, however, there is no color-preserving homomorphism from $\mathbb{G}(\mathrm{H})$ to G at all and we can just output 0. Otherwise we claim that

$$\mathsf{cp\text{-}Hom}(\mathbb{G}(\mathrm{H}), X \to G) = \mathsf{cp\text{-}Hom}(\mathrm{H}, X \to \mathrm{G}')\,. \tag{6.21}$$

For the first direction, let $a \in \mathsf{cp\text{-}Hom}(\mathbb{G}(\mathrm{H}), X \to G)$. Then there exists a homomorphism $h \in \mathsf{cp\text{-}Hom}(\mathbb{G}(\mathrm{H}) \to G)$ such that $h|_X = a$. We claim that h is contained in $\mathsf{cp\text{-}Hom}(\mathrm{H} \to \mathrm{G}')$ as well. To this end, let $\vec{u} = (u_1, \ldots, u_{\mathsf{a}})$ be a tuple of $E_1(\mathrm{H})$. By the definition of the Gaifman graph it holds that $\vec{u}$ is a clique in $\mathbb{G}(\mathrm{H})$ (recall that we assumed the absence of multiple occurrences). As $h \in \mathsf{cp\text{-}Hom}(\mathbb{G}(\mathrm{H}) \to G)$ it hence holds that $h(\vec{u})$ is a clique of size a in G with colors $u_1, \ldots, u_{\mathsf{a}}$. By the construction of G' we have that $h(\vec{u}) \in E_1(\mathrm{G}')$. Consequently $h \in \mathsf{cp\text{-}Hom}(\mathrm{H} \to \mathrm{G}')$ and $a \in \mathsf{cp\text{-}Hom}(\mathrm{H}, X \to \mathrm{G}')$.

For the backward direction, let $a \in \mathsf{cp\text{-}Hom}(\mathrm{H}, X \to \mathrm{G}')$. Then there exists a homomorphism $h \in \mathsf{cp\text{-}Hom}(\mathrm{H} \to \mathrm{G}')$ such that $h|_X = a$. We claim that h is contained in $\mathsf{cp\text{-}Hom}(\mathbb{G}(\mathrm{H}) \to G)$ as well. To see this, let $e = \{v, w\} \in E(\mathbb{G}(\mathrm{H}))$. By the definition of the Gaifman graph, there exists an edge $\vec{u} = \{u_1, \ldots, u_{\mathsf{a}}\}$ in $E_1(\mathrm{H})$ such that $v = u_i$ and $w = u_j$ for some $1 \leq i < j \leq \mathsf{a}$. As $h \in \mathsf{cp\text{-}Hom}(\mathrm{H} \to \mathrm{G}')$ it holds that $h(\vec{u}) \in E_1(\mathrm{G}')$. By the construction of G' we have that $h(\vec{u})$ is a clique in G, colored with $u_1, \ldots, u_{\mathsf{a}}$. In particular it holds that $\{h(u_i), h(u_j)\} \in E(G)$ and that $h(u_i)$ has color u_i and $h(u_j)$ has color u_j. As $v = u_i$ and $w = u_j$ we conclude that $h \in \mathsf{cp\text{-}Hom}(\mathbb{G}(\mathrm{H}) \to G)$ and hence $a \in \mathsf{cp\text{-}Hom}(\mathbb{G}(\mathrm{H}), X \to G)$.

This completes the reduction for the restricted case. We remark that the claimed running time bound follows from the fact that G' can be constructed in time $f(\mathrm{H}) \cdot n^{\mathsf{a}(\tau)}$ as we only need to search for cliques of size $\leq \mathsf{a}(\tau)$.

Let us now explain how to get rid of the restrictions. First, consider a tuple of H that contains multiple occurrences of a vertex, e.g.

$$\vec{u} = (u_1, u_2, u_1, u_1, u_3, u_2) \in E_1(\mathrm{H})\,.$$

We observe that there are exactly 3 different vertices: u_1, u_2 and u_3. Hence, when constructing G', we search for cliques of size 3, colored with u_1, u_2 and u_3. Then, for every clique (a, b, c) in G colored with (u_1, u_2, u_3) we add (a, b, a, a, c, b) to $E_1(\mathrm{G}')$.

Finally, we can assume that τ contains more than one relation symbol by employing the construction for every relation. It is easy to see that Equation 6.21 remains true for the unrestricted case if the construction is modified as explained above. ■

6.5.2 Equivalence of Conjunctive Queries

In this subsection we prove that every endomorphism of a minimal graphical conjunctive that bijectively maps the free variables to itself is already an automorphism. Recall that two conjunctive queries (H, X) and $(\hat{\mathrm{H}}, \hat{X})$ are called equivalent, we write $(\mathrm{H}, X) \sim (\hat{\mathrm{H}}, \hat{X})$, if $\#\mathsf{Hom}(\mathrm{H}, X \to \star)$ and $\#\mathsf{Hom}(\hat{\mathrm{H}}, \hat{X} \to \star)$ are the same functions and a query (H, X) is called minimal if it is a vertex-minimal element in its equivalence class.

Chen and Mengel provided an explicit criterion for equivalence.[13]

Lemma 6.50 ([32]). *Two conjunctive queries* (H, X) *and* $(\hat{\mathrm{H}}, \hat{X})$ *are equivalent if and only if there exist surjective functions* $s : X \to \hat{X}$ *and* $\hat{s} : \hat{X} \to X$ *that can be extended to homomorphisms* $h \in \mathsf{Hom}(\mathrm{H} \to \hat{\mathrm{H}})$ *and* $\hat{h} \in \mathsf{Hom}(\hat{\mathrm{H}} \to \mathrm{H})$*, respectively.*

We do not want to distinguish between two conjunctive queries that are equal up to consistently renaming both, the quantified as well as the free variables. Hence we say that two conjunctive queries (H, X) and $(\hat{\mathrm{H}}, \hat{X})$ are *isomorphic* if and only if there is an isomorphism from H to $\hat{\mathrm{H}}$ that bijectively maps X to $\hat{X}$. We write $(\mathrm{H}, X) \cong (\hat{\mathrm{H}}, \hat{X})$.

We now prove Lemma 6.9 in the more general context of structures. We start by introducing the necessary preliminaries with respect to cores of structures and homomorphic equivalence: Two structures H and $\hat{\mathrm{H}}$ are *homomorphically equivalent* if there exist homomorphisms from H to $\hat{\mathrm{H}}$ and from $\hat{\mathrm{H}}$ to H. H is called a *core* if it is not homomorphically equivalent to a proper substructure of H. As for every proper substructure of H, the identity function is a homomorphism, we obtain

Observation 6.51. *A structure* H *is a core if and only if there exists no homomorphism from* H *to a proper substructure. Hence, every endomorphism of a core is an automorphism.*

We say that a substructure $\hat{\mathrm{H}}$ of H is a core of H if $\hat{\mathrm{H}}$ and H are homomorphically equivalent and $\hat{\mathrm{H}}$ is a core.

Lemma 6.52 (cf. Lemma 13.9 in [65]). *Let* H *and* G *be homomorphically equivalent and let* $\hat{\mathrm{H}}$ *and* $\hat{\mathrm{G}}$ *be cores of* H *and* G*, respectively. Then* $\hat{\mathrm{H}}$ *and* $\hat{\mathrm{G}}$ *are isomorphic. In particular, all cores of a structure are isomorphic.*

Corollary 6.53. *All minimal elements in a single equivalence class with respect to homomorphic equivalence are isomorphic.*

[13]Note that in case of graphs, Lemma 6.50 is identical with Corollary 6.37. While Chen and Mengel gave a more involved proof of the lemma, we point out that the generalization to structures can be proved just as easy as we demonstrated it for graphs in Chapter 6.4.

Now Lemma 6.52 and Corollary 6.53 allow us to speak of *the* core of a structure — we write $\mathsf{core}(\mathrm{H})$ — and *the* minimal representative of a homomorphic equivalence class. Our goal is to achieve a similar result for equivalence of conjunctive queries. To this end, we use the *augmented core* of a graphical conjunctive query which refines the notion of a core.

Definition 6.54. Given a conjunctive query (H, X), we obtain an augmented structure $\mathsf{aug}(\mathrm{H}, X)$ from H by adding a new relation

$$E_{\mathsf{aug}}(\mathrm{H}) = \{(x, x') \in X^2 \mid x \neq x'\}\,.$$

Note that this also adds a new relation symbol E_{aug} to the signature. We let H_c be the structure obtained from $\mathsf{core}(\mathsf{aug}(\mathrm{A}, X))$ by removing $E_{\mathsf{aug}}(\mathrm{H})$ and we define the *augmented core* of (H, X) to be $\mathsf{core}_{\mathsf{aug}}(\mathrm{H}, X) := (\mathrm{H}_c, X)$.

We note that $\mathsf{core}_{\mathsf{aug}}$ is well-defined, i.e., that every element $x \in X$ is contained in the core of $\mathsf{aug}(\mathrm{H}, X)$. To see this, we observe that $E_{\mathsf{aug}}(\mathrm{H})$ induces a clique without loops on X and hence, any substructure $\hat{\mathrm{H}}$ of H such that there is a homomorphism from H to $\hat{\mathrm{H}}$ must contain every element $x \in X$.

Lemma 6.55. *Let* (H, X) *and* $(\hat{\mathrm{H}}, \hat{H})$ *be two graphical conjunctive queries. Then it holds that* (H, X) *and* $(\hat{\mathrm{H}}, \hat{H})$ *are equivalent if and only if* $\mathsf{aug}(\mathrm{H}, X)$ *and* $\mathsf{aug}(\hat{\mathrm{H}}, \hat{H})$ *are homomorphically equivalent. Moreover,* $\mathsf{aug}(\mathrm{H}, X)$ *is minimal with respect to homomorphic equivalence if and only if* (H, X) *is minimal.*

Proof. Every homomorphism from $\mathsf{aug}(\mathrm{H}, X)$ to $\mathsf{aug}(\hat{\mathrm{H}}, \hat{H})$ must surjectively map X to $\hat{X}$ as E_{aug} induces a clique (without self-loops) on X and $\hat{X}$. On the other hand, every homomorphism from H to $\hat{\mathrm{H}}$ that surjectively maps X to $\hat{X}$ is a homomorphism from $\mathsf{aug}(\mathrm{H}, X)$ to $\mathsf{aug}(\hat{\mathrm{H}}, \hat{H})$.

If $\mathsf{aug}(\mathrm{H}, X)$ is not minimal with respect to homomorphic equivalence then it is not a core. Hence $\mathsf{aug}(\mathrm{H}, X)$ is homomorphically equivalent to a proper substructure F of $\mathsf{aug}(\mathrm{H}, X)$. Hence F must contain all vertices in X and $E_{\mathsf{aug}}(\mathrm{F}) = E_{\mathsf{aug}}(\mathrm{H})$, because otherwise there would be no homomorphism from $\mathsf{aug}(\mathrm{H}, X)$ to F. It follows that $\mathrm{F} = \mathsf{aug}(\hat{\mathrm{H}}, X)$ for some structure $\hat{\mathrm{H}}$. By the first part, it follows that (H, X) and $(\hat{\mathrm{H}}, X)$ are equivalent. As $\hat{\mathrm{H}}$ is a proper substructure of H is follows that (H, X) is not minimal.

On the other hand, if (H, X) is not minimal, then there exists a conjunctive query $(\hat{\mathrm{H}}, \hat{H})$ such that $\hat{\mathrm{H}}$ is a proper substructure of H and there are homomorphisms h from H to $\hat{\mathrm{H}}$ and $\hat{h}$ from $\hat{\mathrm{H}}$ to H that surjectively map X to $\hat{X}$ and $\hat{X}$ to X respectively. Hence h and $\hat{h}$ are also homomorphisms from $\mathsf{aug}(\mathrm{H}, X)$ to $\mathsf{aug}(\hat{\mathrm{H}}, \hat{H})$ and from $\mathsf{aug}(\hat{\mathrm{H}}, \hat{H})$ to $\mathsf{aug}(\mathrm{H}, X)$, respectively. Therefore $\mathsf{aug}(\mathrm{H}, X)$ is not minimal with respect to homomorphic equivalence. ∎

The previous lemma, together with Lemma 6.52 and Corollary 6.53, immediately implies the following:

Corollary 6.56. *For all graphical conjunctive queries* (H, X) *and* $(\hat{\mathrm{H}}, \hat{H})$ *it holds that*

(1) $(\mathrm{H}, X) \sim \mathsf{core}_{\mathsf{aug}}(\mathrm{H}, X)$.

(2) If (H, X) *is minimal then* $(\mathrm{H}, X) \cong \mathsf{core}_{\mathsf{aug}}(\mathrm{H}, X)$.

(3) If (H, X) *and* $(\hat{\mathrm{H}}, \hat{H})$ *are minimal and* $(\mathrm{H}, X) \sim (\hat{\mathrm{H}}, \hat{H})$ *then we have that* $(\mathrm{H}, X) \cong (\hat{\mathrm{H}}, \hat{H})$.

(4) If (H, X) *is minimal and* h *is an endomorphism of* H *that surjectively maps* X *to* X*, then* h *is an automorphism.*

Note that Lemma 6.9 follows from (4), restricted to the signature of graphs.

6.5.3 The Generalized Classification Theorem

Before generalizing our main theorem, we need to point out that all structural parameters we have seen for conjunctive queries over the signature of graphs are generalized to structures via the Gaifman graph. We furthermore observe that the reduction from color-prescribed to uncolored homomorphisms in Chapter 6.2.1 as well as the transformation of existential and universal positive queries, possibly including inequalities and non-monotone constraints over the free variables, can be done completely analogously as in the case of graphs. However, we did not give a formal proof of #W[1]-easiness in case of bounded dominating star size yet, which is hence provided first.

Theorem 6.57 (Theorem 6.19 for structures). *Let* Δ *be a class of conjunctive queries with bounded dominating star size. Then* $\#\textsc{Hom}(\Delta)$ *is* #W[1]*-easy.*

Proof. We construct a parameterized Turing reduction to $\#\textsc{p-MC}(\Pi_0)$ which is #W[1]-equivalent by Theorem 2.59.

Given an instance of $\#\textsc{Hom}(\Delta)$, i.e., a conjunctive query $(\mathrm{H}, X) \in \Delta$ and a structure G, we will construct in FPT time a query $(\hat{\mathrm{H}}, X) \in \Pi_0$ and a structure $\hat{\mathrm{G}}$ such that

$$\#\mathsf{Hom}(\mathrm{H}, X \to \mathrm{G}) = \#\mathsf{Hom}(\hat{\mathrm{H}}, X \to \hat{\mathrm{G}})\,,$$

using an "oracle for #W[1]" with the additional restriction that the parameter of every query to the oracle only depends on (H, X). The latter oracle will be implemented by using the provided oracle access to $\#\textsc{p-MC}(\Pi_0)$.

Let $\mathbb{G} = \mathbb{G}(\mathrm{H})$ be the Gaifman graph of H, let Y be the set of quantified variables in (H, X) and let $Y_1, \dots, Y_\ell$ be the connected components of $\mathbb{G}[Y]$. Furthermore, for each $i \in [\ell]$ we set c_i to be the number of vertices in X that are adjacent to a vertex in Y_i in $\mathbb{G}$. Note that every c_i is bounded by some overall constant as Δ has bounded dominating star size. Next we add new relation symbols $\hat{E}_1, \dots, \hat{E}_\ell$ with arities $c_1, \dots, c_\ell$ to the signature. We proceed for each $i \in [\ell]$ as follows: Let $x^i_1, \dots, x^i_{c_i} \in X$ be the elements in H that are adjacent to an element in Y_i in $\mathbb{G}$. Now, for every tuple $\vec{v} = (v_1, \dots, v_{c_i}) \in V(\mathrm{G})^{c_i}$, we check whether there is a homomorphism h from $\mathrm{H}[Y_i \cup \{x^i_1, \dots, x^i_{c_i}\}]$ to G such that $h(x^i_j) = v_j$ for all $j \in [c_i]$. Intuitively, this check is positive if $\vec{v}$ is a possible candidate for the image of the free variables $x^i_1, \dots, x^i_{c_i} \in X$ with respect to "neighborhood" Y_i. Next we add $\vec{v}$ to $\hat{E}_i(\mathrm{G})$ if and only if the check was positive and remark that we only need to perform $\ell \cdot |\mathrm{G}|^{O(1)}$ checks as all c_i are upper bounded by a constant. The resulting structure is $\hat{\mathrm{G}}$. We observe that every check can be done by querying an oracle to $\mathsf{W}[1]$ as it can be formulated as an instance of the problem of deciding the existence of a solution to a conjunctive query, which is known to be $\mathsf{W}[1]$-easy (cf. Theorem 7.22 in [65]). As we have access to an oracle for $\#\mathsf{W}[1]$, we can certainly simulate an oracle for $\mathsf{W}[1]$; if we know the number of solutions we also know whether one exists.

It remains to show how to construct $(\hat{\mathrm{H}}, X)$: We fix an ordering of the free variables X of φ. Next, starting from H, for every $i \in [\ell]$, let $x^i_1, \dots x^i_{c_i}$ be the ordered neighbors of vertices in Y_i in $\mathbb{G}$. We then add an atom $(x^i_1, \dots x^i_{c_i}) \in \hat{E}_i$ to H. In the end, we delete all quantified variables from H along with all tuples that contain a quantified variable and denote the resulting structure as $\hat{\mathrm{H}}$. Now it can be easily verified that

$$\#\mathsf{Hom}(\mathrm{H}, X \to \mathrm{G}) = \#\mathsf{Hom}(\hat{\mathrm{H}}, X \to \hat{\mathrm{G}})\,.$$

We summarize the reduction:

(1) Given (H, X) and G, construct $\hat{\mathrm{G}}$ by performing at most $\ell \cdot |\mathrm{G}|^{\mathsf{dss}(\mathrm{H},X)}$ checks, using the oracle.

(2) Construct H.

(3) Query the oracle to obtain $\#\mathsf{Hom}(\hat{\mathrm{H}}, X \to \hat{\mathrm{G}})$ and output the result.

This can be done in FPT time as the dominating star size of Δ is bounded. For the same reason, the arity of H is bounded by a constant. Furthermore, there is a computable function t such that the size of every oracle query is bounded by $t(|(\mathrm{H}, X)|) \cdot \mathsf{poly}(|\mathrm{G}|)$ and the parameter of every oracle query is bounded by $t(|(\mathrm{H}, X)|)$. ■

The following classification subsumes Theorem 6.6 in the general case of logical signatures with bounded arity.

Theorem 6.58 (Counting answers to conjunctive queries). *Let Δ be a recursively enumerable class of minimal conjunctive queries over arbitrary signatures with bounded arity.*

(1) If the treewidth of Δ and $\mathsf{contract}(\Delta)$ is bounded, then $\#\textsc{Hom}(\Delta)$ is polynomial-time computable.

(2) If the treewidth of Δ is unbounded and the treewidth of $\mathsf{contract}(\Delta)$ is bounded, then $\#\textsc{Hom}(\Delta)$ is $\mathsf{W}[1]$-equivalent.

(3) If the treewidth of $\mathsf{contract}(\Delta)$ is unbounded and the dominating star size of Δ is bounded, then $\#\textsc{Hom}(\Delta)$ is $\#\mathsf{W}[1]$-equivalent.

(4) If the dominating star size of Δ is unbounded, then $\#\textsc{Hom}(\Delta)$ is $\#\mathsf{W}[2]$-hard. In particular, given a formula δ with

$$\mathsf{dss}(\delta) \geq \max\{3, \mathsf{arity}(\delta)\}\,,$$

then computing $\#\mathsf{Hom}(\delta \to \star)$ cannot be done in time $O(n^{\mathsf{dss}(\delta)-\varepsilon})$ for any $\varepsilon > 0$ unless SETH fails.

(5) If additionally the linked matching number of Δ is unbounded, then $\#\textsc{Hom}(\Delta)$ is $\#\mathsf{A}[2]$-equivalent.

Furthermore, the classification remains true if Δ is a family of quantum queries, with the exception that the queries in the first case might only be fixed parameter tractable.

Proof. We have that (1), (2), and $\#\mathsf{W}[1]$-hardness in (3) follow from [31]. $\#\mathsf{W}[1]$-easiness in (3) follows from Theorem 6.57. (4) and hardness in (5) hold as we can apply the corresponding results for the Gaifman graphs (Theorem 6.22 and Theorem 6.30) and then reduce to the primal structures via Lemma 6.49. In particular, the condition

$$\mathsf{dss}(\delta) \geq \max\{3, \mathsf{arity}(\delta)\}$$

in (4) is necessary as the reduction in Lemma 6.49 takes time $n^{\mathsf{arity}(\delta)}$. After that, we can reduce color-prescribed homomorphisms to uncolored homomorphisms completely analogously as we did for graphs (Chapter 6.2.1). $\#\mathsf{A}[2]$-easiness follows from the fact that the general problem of counting answers to conjunctive queries is easy for $\#\mathsf{A}[2]$ by definition. Finally, the extension to quantum graphs holds by the complexity monotonicity property, which follows by a straight-forward adaption of the proof of Lemma 6.38 for structures. Alternatively, the original but more involved version of Chen and Mengel [32] can be applied. ∎

Now transforming universal and existential positive formulas, possibly with inequalities and non-monotone constraints over the free variables, to a quantum query can also be done completely analogously as for graphs, proving Theorem 6.5 and Theorem 6.7 in the general case. We point out that, in case of conjunctive queries with inequalities over the free variables, the criterion for a query to be contained in the support of the linear combination can be stated in terms of vertex contractions along matroid flats similar to the case of graphs (see Theorem 6.41).

Corollary 6.59. *The classification of Theorem 6.58 applies to all recursively enumerable families Δ of universal and existential positive formulas of bounded arity that might contain inequalities and non-monotone constraints over the free variables. In particular, if Δ is a set of conjunctive queries with inequalities over the free variables, the criteria for the classification can be stated explicitly as in Theorem 6.41.*

Chapter 7

Conclusions and Future Research

Quantum graphs and the complexity monotonicity property of computing linear combinations of homomorphism numbers have been shown to be a powerful framework for the parameterized and exact complexity analysis of counting problems. As we have seen, they do not only yield significantly simpler proofs of existing dichotomy results, but also new exhaustive and mostly explicit classifications for parameterized counting problems whose complexity has not or only very partially been resolved with prior methods.

Now let us go back to one of the original motivations for the study of the *parameterized* complexity of counting problems. The idea was to use parameterization as a possible relaxation of problems that are intractable in the classical sense. From this algorithmic point of view, the results of this work are quite negative: By the complexity monotonicity property, a parameterized counting problem is hard whenever an equivalent expression as homomorphism counting problem from quantum graphs with constituents of unbounded treewidth can be found. For many problems in this thesis, this induces only very few non-trivial tractability results.

Consider for example the problem #IndSub(Φ) for bipartite monotone graph properties Φ. The main result of Chapter 5.2 states that there are *no* non-trivial fixed-parameter tractable cases under ETH and thus completely rules out parameterization as a relaxation that yields efficient algorithms. Furthermore, any significant improvement over brute-force is impossible, unless ETH fails.

What can be done about problems like #IndSub(Φ)? We conjecture that approximate counting might be a very fruitful approach. Let us give some evidence for this claim: Jerrum and Meeks proved the existence of what is called a fixed-parameter tractable randomized approximation scheme (FPTRAS) for #IndSub(Φ) if Φ is the property of being connected [79].

This is a particular interesting result as, by Chapter 5.1, the quantum graph expressing of the number of connected induced subgraphs of size k contains the clique of size k as a coefficient. In contrast, the number of cliques of size k can most likely not be approximated up to a constant factor by an FPT algorithm as this would solve the decision version CLIQUE. However, the latter is known to not admit an FPT algorithm unless ETH fails [33, 34]. Consequently, complexity monotonicity *fails* for approximate counting and hence none of the reductions from #CLIQUE in this thesis rules out approximation algorithms.

For this reason, we suggest the combination of parameterization and approximation for intractable counting problems that avoid improvements if only one of the former is applied. Indeed, there have been some results in this field, such as the approximation algorithm for counting subgraphs with bounded treewidth due to Arvind and Raman [6], the approximation algorithm for counting even and odd subgraphs due to Jerrum and Meeks[81] and the method of "Extensor-coding" due to Brand, Dell and Husfeldt [15]. A more systematic study of parameterized randomization, including some implications for the structural complexity of approximate counting is due to Montoya and Müller [103] (see also [57, Chapter 32.7]). However, beyond these results, not much is known.

Appendix A

Proof of Lemma 4.11: Counting Subtrees of a Tree

The aim of this chapter is to prove Lemma 4.11, i.e., that counting locally injective homomorphisms is #P-hard and thus not solvable in polynomial time under standard assumptions. We start by giving an introduction to classical counting complexity which was established by Valiant in his seminal work about the complexity of computing the permanent [128]. A (non-parameterized) *counting problem* is a function $F : \{0,1\}^* \to \mathbb{N}$. The class of all counting problems solvable in polynomial time is called FP. On the other hand, the notion of intractability is #P-hardness. #P is the class of all counting problems reducible[1] to #SAT, the problem of computing the number of satisfying assignments of a given CNF formula. A counting problem F is #P-hard if there exists a polynomial-time Turing reduction from #SAT to F, that is, an algorithm with oracle access to F that solves #SAT in polynomial time. Toda [126] (see also [71, Chapter F.1]) proved that

$$\mathsf{PH} \subseteq \mathsf{P}^{\#\mathsf{P}},$$

where PH is the *Polynomial-Time Hierarchy* (cf. [71, Chapter 3.2]) and $\mathsf{P}^{\#\mathsf{P}}$ is the class of all languages that can be decided in polynomial time if provided oracle access to #SAT. As $\mathsf{NP} \subseteq \mathsf{PH}$ and the inclusion is assumed to be strict, this indicates that #P-hard problems are even harder than NP-complete problems.

For the proof of Lemma 4.11, we will first show #P-hardness of the following intermediate problem: Given two trees T_1, T_2, compute the number $\#\mathsf{Sub}(T_1 \to T_2)$ of subtrees of T_2 that are isomorphic to T_1. We call this problem #SUB$(\mathcal{T}, \mathcal{T})$.

[1](Many-one) reductions in counting complexity differ slightly from many-one reductions in the decision world. However, for the purpose of this chapter we only need Turing reductions. We recommend Chapter 6.2 in [71] to the interested reader.

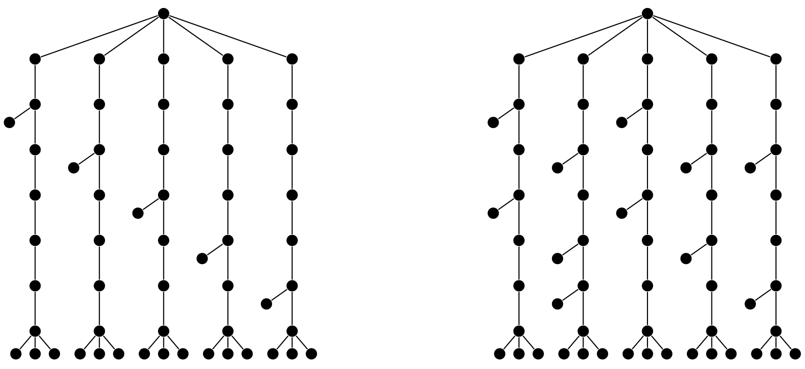


Figure A.1: Trees T_{id_5} (*left*) and T_B (*right*).

Lemma A.1. $\#\text{Sub}(\mathcal{T},\mathcal{T})$ *is* $\#\mathsf{P}$*-hard.*

Related results are #P-hardness for counting *all* subtrees of a given graph due to Jerrum [78] or even counting *all* subtrees of a given tree due to Goldberg and Jerrum [70]. As the number of non-isomorphic trees with n vertices is not bounded by a polynomial in n, we do not know how to reduce directly from those problems. Instead we use a construction quite similar to the "skeleton" graph in the construction of Goldberg and Jerrum [70] to reduce from the problem of computing the permanent.

Given a quadratic matrix A with elements $(a_{i,j})_{i,j\in[n]}$ the *permanent* of A is defined by

$$\mathsf{perm}(A) := \sum_{\pi\in\mathsf{Sym}_n}\prod_{i=1}^{n} a_{i,\pi(i)}\,,$$

where Sym_n is the symmetric group with n elements.

Theorem A.2 ([128]). *Computing the permanent is* $\#\mathsf{P}$*-hard even when restricted to matrices with entries from* $\{0,1\}$.

Proof (Proof of Lemma A.1). We reduce from computing the permanent of matrices with entries from $\{0,1\}$. Given a quadratic matrix A of size n, we construct a tree T_A as follows:

1. For every entry $a_{i,j}$ we create a vertex $v_{i,j}$ and add edges $\{v_{i,j},v_{i+1,j}\}$ for every $i\in[n-1]$ and $j\in[n]$.
2. Whenever $a_{i,j}=1$ we create a vertex $b_{i,j}$ and add edges $\{b_{i,j},v_{i,j}\}$.
3. For every column c_j we create a vertices u_j,w_j,x_j,y_j,z_j and add edges $\{u_j,v_{1,j}\}$, $\{v_{n,j},w_j\}$,$\{w_j,x_j\}$,$\{w_j,y_j\}$ and $\{w_j,z_j\}$.
4. Finally, we create a vertex r and add edges $\{r,u_j\}$ for all $j\in[n]$. In what follows, we call r the root.

We give an example in Figure A.1 for the matrix

$$B := \begin{pmatrix} 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \end{pmatrix} .$$

We claim that for all quadratic matrices A of size $n \geq 5$ with entries from the set $\{0, 1\}$ it holds that

$$\mathsf{perm}(A) = \#\mathsf{Sub}(T_{\mathsf{id}_n} \to T_A) ,$$

where id_n is the quadratic matrix of size n with 1s on the diagonal and 0s everywhere else. In the following we write v for a vertex in T_A and v' for a vertex in T_{id_n}. To prove the claim we first observe that whenever a subtree of T_A is isomorphic to T_{id_n}, the root r' of T_{id_n} has to be mapped to the root r of T_A by the isomorphism as the roots are the only vertices with degree n (which is why we needed $n \geq 5$ as every other vertex has degree ≤ 4). It follows that the vertices $u'_1, \dots, u'_n$ of T_{id_n} are mapped to the vertices $u_1, \dots, u_n$ of T_A which induces a permutation on n elements, that is, an element $\pi \in \mathsf{Sym}_n$.

We will now partition the subtrees of T_A isomorphic to T_{id_n} by those permutations and write $\#\mathsf{Sub}(T_{\mathsf{id}_n} \to T_A)[\pi]$ for the number of subtrees that induce π. Now fix π and consider a subtree that induces π. It holds that for all $j \in [n]$ the vertex w'_j has to be mapped to $w_{\pi(j)}$ as those are the only vertices with degree exactly 4 and furthermore, the vertices x'_j, y'_j, z'_j have to be mapped to $x_{\pi(j)}, y_{\pi(j)}, z_{\pi(j)}$ (possibly permuted but the subtree of T_A is the same). Now $v'_{i,i}$ is adjacent to $b'_{i,i}$ for each $i \in [n]$ and thus $v_{i,\pi(i)}$ has to be adjacent to $b_{i,\pi(i)}$, that is $a_{i,\pi(i)} = 1$. If this is not the case then there is no subtree that induces partition π. Furthermore there is at most one subtree isomorphic to T_{id_n} inducing π because the image is enforced by the vertices r', w'_j and $v'_{i,i}$ for all $i, j \in [n]$. Consequently

$$\#\mathsf{Sub}(T_{\mathsf{id}_n} \to T_A)[\pi] = 1 ,$$

if for all $i \in [n]$ it holds that $a_{i,\pi(i)} = 1$, and otherwise

$$\#\mathsf{Sub}(T_{\mathsf{id}_n} \to T_A)[\pi] = 0 .$$

Hence $\#\mathsf{Sub}(T_{\mathsf{id}_n} \to T_A)[\pi] = \prod_{i=1}^{n} a_{i,\pi(i)}$ and therefore

$$\mathsf{perm}(A) = \sum_{\pi \in \mathsf{Sym}_n} \prod_{i=1}^{n} a_{i,\pi(i)} = \sum_{\pi \in S_n} \#\mathsf{Sub}(T_{\mathsf{id}_n} \to T_A)[\pi] = \#\mathsf{Sub}(T_{\mathsf{id}_n} \to T_A) .$$

Now the reduction works as follows: If the input matrix A has size ≤ 4 we brute-force the output and otherwise we compute $\#\mathsf{Sub}(T_{\mathsf{id}_n} \to T_A)$ with the oracle for $\#\textsc{Sub}(\mathcal{T}, \mathcal{T})$. ■

The proof of Lemma 4.11 relies on the fact that locally injective homomorphisms from a tree to a tree are embeddings.

Proof (of Lemma 4.11). It is a well-known fact that a locally injective homomorphism h from a tree T_1 to a tree T_2 is (fully) injective. To see this, assume that there are vertices v and u in T_1 that are mapped to the same vertex in T_2. As T_1 is a tree there exists exactly one path

$$v = w_0, w_1, \ldots, w_\ell, w_{\ell+1} = u$$

between v and u in T_1. It holds that $\ell \geq 1$ as otherwise v and u would be adjacent and hence $h(u) = h(v)$ would have a self-loop in T_2 which is impossible. As h is locally injective we have that $h(v) \neq h(w_2)$, hence $u \neq w_2$, and as h is edge-preserving there are edges $\{h(v), h(w_1)\}$ and $\{h(w_1), h(w_2)\}$ and a path from $h(w_2)$ to $h(w_{\ell+1}) = h(u) = h(v)$ in T_2. This induces a cycle and contradicts the fact that T_2 is a tree.

Therefore $\#\mathsf{Emb}(T_1 \to T_2) = \#\mathsf{Li\text{-}Hom}(T_1 \to T_2)$. By Fact 2.9, we have that for all H and G the following is true

$$\#\mathsf{Sub}(H \to G) = \frac{\#\mathsf{Emb}(H \to G)}{\#\mathsf{Aut}(H)}\,.$$

If H is a tree then H is planar and thus $\#\mathsf{Aut}(H)$ can be computed in polynomial time [98, 74]. Therefore #P-hardness of $\#\textsc{Li-Hom}(\mathcal{T})$ follows by reducing from $\#\textsc{Sub}(\mathcal{T}, \mathcal{T})$, which is hard by Lemma A.1: Given trees T_1 and T_2 we obtain $\#\mathsf{Li\text{-}Hom}(T_1 \to T_2)$ by querying the oracle and compute $\#\mathsf{Aut}(T_1)$ in polynomial time. Finally, we output

$$\frac{\#\mathsf{Li\text{-}Hom}(T_1 \to T_2)}{\#\mathsf{Aut}(T_1)} = \frac{\#\mathsf{Emb}(T_1 \to T_2)}{\#\mathsf{Aut}(T_1)} = \#\mathsf{Sub}(T_1 \to T_2)\,.$$

■

Appendix B

Modular Counting of Induced Subgraphs

In this part of the appendix, we show that our main result of Chapter 5.2 (Theorem 5.48) can easily be extended to counting modulo a fixed prime:

Theorem B.1. *Let p be a prime number, Φ be a computable graph property and let $\mathcal{K}$ be the set of all prime powers $t = p^k$ such that $\Phi(\mathsf{IS}_{2t}) \neq \Phi(K_{t,t})$. If $\mathcal{K}$ is infinite then $\mathsf{Mod}_p\textsc{IndSub}(\Phi)$ is $\mathsf{Mod}_p\mathsf{W}[1]$ hard. If additionally $\mathcal{K}$ is dense then it cannot be solved in time $f(k) \cdot n^{o(k)}$ for any computable function f unless ETH fails. This holds true even if the input graphs to $\mathsf{Mod}_p\textsc{IndSub}(\Phi)$ are restricted to be bipartite.*

Here $\mathsf{Mod}_p\textsc{IndSub}(\Phi)$ asks, given G and k, to compute the number of induced subgraphs with k vertices in G that satisfy Φ *modulo p*. The parameterized complexity class $\mathsf{Mod}_p\mathsf{W}[1]$ is defined by the problem of, given G and k, deciding whether the number of k-cliques in G is 0 modulo p, which is complete for the class (see [13] for $p = 2$ and Chapter 1.2.2 in [39] for the general case).

First of all, we point out that the modular counting version of Theorem 5.33 follows as corollary from the above theorem in the same way Theorem 5.33 follows from Theorem 5.48. For the proof of Theorem B.1 we will first establish the following matrix to be invertible; this fact is crucial for the color-prescribed variant of complexity monotonicity.

Lemma B.2. *Let H be a graph and let M be a quadratic matrix of size $2^{|E(H)|}$ such that the rows and columns are identified by the subsets of edges of H. Furthermore assume that the entries of M are given by*

$$M(S,T) := \#\mathsf{cp\text{-}Hom}(H[S] \to H[T])\,.$$

Then M is non-singular. This holds true even if M is considered as a matrix over $\mathbb{Z}_p$, that is, the field with p elements. In the latter case, the entries are taken modulo p.

Proof. We fix any linear extension $\lesssim$ of the subset inclusion relation on $E(H)$ and order the columns and rows of M accordingly. We claim that M is triangular. To see this we first observe that $M(S,S) = 1$ for every S, given by the identity homomorphism from $H[S]$ to $H[S]$ which is, of course, color-prescribed. Now consider $M(S,T)$ for some $T \neq S$ with $T \lesssim S$. It follows that there exists an edge $\{u,v\}$ in $S \backslash T$ since $\lesssim$ linearly extends subset inclusion. Now assume that there exists a color-prescribed homomorphism h from $H[S]$ to $H[T]$. By color-prescribedness we have that $h(u) = u$ and $h(v) = v$, contradicting the fact that h is a homomorphism and $\{u,v\} \notin T$. Hence $M(S,T) = 0$ and, consequently, M is triangular. ∎

Furthermore, we rely on the following fact stating that all required reductions in Chapter 5.2 work as well in the case of counting modular a prime number.

Fact B.3. *Let p be a fixed prime number. Then Lemma 5.45 and 5.47 remain true when counting is done modulo p if the graph H is restricted to be $K_{t,t}$ for some prime power $t = p^k$.*

The only two non-trivial observations required to verify Fact B.3 are, first, that $\hat{\chi}(\Phi, K_{t,t}) \neq 0 \mod p$ whenever $\Phi(K_{t,t}[\emptyset]) \neq \Phi(K_{t,t})$ (Lemma 5.36) and, second, that complexity monotonicity (Lemma 5.42) holds for computation modulo p as well, since non-singularity of the matrix M in the proof is given by Lemma B.2 even in case the entries of M are considered to be elements of $\mathbb{Z}_p$. The last ingredient for the proof of Theorem B.1, in particular for hardness under ETH, requires a method of isolating cliques that works in the parameterized setting. This is given by the following result of Williams et al.

Lemma B.4 (Lemma 2.1 in [137]). *Let $p \geq 2$ be an integer, G, H be undirected graphs. Let G' be a random induced subgraph of G such that each vertex is taken with probability $1/2$, independently. If there is at least one induced H in G, the number of induced H in G' is not a multiple of p with probability at least $2^{-|H|}$.*

Proof (of Theorem B.1). The proof is most similar to the proof of the tight lower bound under ETH in Theorem 5.48. We start our reduction from the problem of *finding* a clique of size k. In case $\mathcal{K}$ is dense and we aim to establish the ETH hardness result, we perform the following two initial steps before the main reduction:

1. Given G and k, we construct a graph $\hat{G}$ such that G contains a clique of size k if and only if $\hat{G}$ contains a clique of size ℓ where $k \leq \ell \leq ck$ for some overall constant c. The details of the construction are given in the proof of Theorem 5.48.

2. We use Lemma B.4 to isolate an ℓ-clique in $\hat{G}$, assuming there is any, with high probability.

For the main part of the reduction we then first apply the reduction from counting cliques to counting color-prescribed homomorphisms from the biclique as given by Lemma 5.49. In particular, this reduction is parsimonious. Finally, we proceed from this point on precisely as in the proof of Theorem 5.48, the correctness of which follows by Fact B.3.

We conclude by pointing out that, in case the randomized construction of Lemma B.4 was used, we can perform probability amplification by repeating the final algorithm 2^k times to end up in a constant success probability. ■

Appendix C

Normalization for #A[2]

In this chapter, we prove that the problem of counting answers to graphical conjunctive queries is #A[2]-equivalent. This result can be seen as a counting analogue of the Normalization Theorem of the A-Hierarchy due to Flum and Grohe [65, Chapter 8]. While most of their reductions directly translate to the counting version, one major step can be drastically simplified using the framework of quantum queries, namely the reduction from existential positive formulas to conjunctive queries.

After establishing the normalization result, we are going to show #A[2]-hardness of #CP-HOM(Grates) for some specific class Grates of all *grates*. This latter result is the starting point for our reduction in Chapter 6.3.5.

C.1 Hardness for Graphs without Self-Loops

We consider the problem of counting answers to arbitrary conjunctive queries over the signature of graphs. The main difficulty when proving #A[2]-hardness of this problem is that the defining problem #P-MC(Π_1) of #A[2] talks about first-order formulas in Π_1 which may contain relations of unbounded arity, and which may contain equalities and negations (cf. [65, Chapter 14]). Recall that until now, every set Φ of first-order formulas was assumed to have a constant bound on the arity of every signature of every formula in Φ and furthermore, that no formula in Φ contains equalities. In this subsection (and *only* in this subsection), we need to omit those restrictions. Given a class Φ of first-order formulas, we write $\Phi[r]$ for the subset of Φ that contains only formulas over signatures of arity at most r. We write Φ[GRAPHS] for the set of formulas in Φ that are over the signature of graphs and that do not "include" self-loops, that is, Ezz is not allowed as an atom for any variable z. In particular, we abuse notation and assume that the input graphs to #P-MC(CQ[GRAPHS]) do not contain self-loops as well. Furthermore, we let $\mathsf{CQ}_=$ be the set of all conjunctive queries that may contain equalities and CQ be the subset without equalities.

Remark C.1. The problem $\#\text{P-MC}(\mathsf{CQ}[\mathsf{GRAPHS}])$ is precisely the problem $\#\text{HOM}(\mathsf{G})$ where G is the set of all conjunctive queries over the signature of graphs without self-loops.

We are ready to prove the following normalization lemma for $\#\mathsf{A}[2]$.

Lemma C.2 (Normalization for #A[2]). *We have that*

$$\#\text{P-MC}(\Pi_1) \leq^{\mathrm{T}}_{\mathrm{fpt}} \#\text{P-MC}(\mathsf{CQ}[\mathsf{GRAPHS}])\,.$$

Proof. We construct a sequence of reductions:

Claim C.3. $\#\text{P-MC}(\Pi_1) \leq^{\mathrm{T}}_{\mathrm{fpt}} \#\text{P-MC}(\Sigma_1)$.

Proof. Let

$$\varphi = x_1 \dots x_k \forall y_1 \dots \forall y_\ell : \psi$$

be a formula in Π_1 such that ψ is quantifier free — note that $x_1, \dots, x_k$ denote the free variables of ψ — and let G be a structure over the same signature as φ. Furthermore, let $n = \#V(\mathrm{G})$.

We define $\varphi' \in \Sigma_1$ as follows:

$$\varphi' = x_1 \dots x_k \exists y_1 \dots \exists y_\ell : \neg\psi\,.$$

Now it can easily be verified that $\#\varphi(\mathrm{G}) = n^k - \#\varphi'(\mathrm{G})$. This induces the reduction. $\square$

Claim C.4. $\#\text{P-MC}(\Sigma_1) \leq^{\mathrm{T}}_{\mathrm{fpt}} \#\text{P-MC}(\Sigma_1^+[\mathsf{GRAPHS}])$, *where* Σ_1^+ *is the set of all existential* positive *formulas.*

Proof. Flum and Grohe proved the following sequence of reductions for every odd $t \geq 1$ in the decision realm (see [65, Chapter 8] and [63]):

$$\text{P-MC}(\Sigma_t) \leq^{\mathrm{T}}_{\mathrm{fpt}} \text{P-MC}(\Sigma_t^+) \leq^{\mathrm{T}}_{\mathrm{fpt}} \text{P-MC}(\Sigma_t^+[2]) \leq^{\mathrm{T}}_{\mathrm{fpt}} \text{P-MC}(\Sigma_t^+[\mathsf{GRAPHS}])\,.$$

A close look reveals that all of the above constructions work as well in the counting world, in particular for $t = 1$. $\square$

Claim C.5. $\#\text{P-MC}(\Sigma_1^+[\mathsf{GRAPHS}]) \leq^{\mathrm{T}}_{\mathrm{fpt}} \#\text{P-MC}(\mathsf{CQ}_=[\mathsf{GRAPHS}])$.

Proof. Every formula $\varphi \in \Sigma_1^+[\mathsf{GRAPHS}]$ is an existential-positive formula. Hence we can express φ as a quantum query over the same signature as in Theorem 6.42. We then use the oracle for $\#\text{P-MC}(\mathsf{CQ}_=[\mathsf{GRAPHS}])$ to compute every term in the linear combination. $\square$

Claim C.6. $\#\text{P-MC}(\mathsf{CQ}_=[\mathsf{GRAPHS}]) \leq^{\mathrm{T}}_{\mathrm{fpt}} \#\text{P-MC}(\mathsf{CQ}[\mathsf{GRAPHS}])$.

Proof. An equality $z_1 = z_2$ in a conjunctive query can easily be simulated by substituting every occurrence of z_2 by z_1 and removing the equality afterwards. If the substitution leads to self-loops we can just output zero, as the input graphs do not contain such. □

This concludes the proof of the normalization lemma. ■

The proof of Claim C.5 is considerably easier than its analogue in the decision world as we were able to make use of the framework of quantum graphs. The proof of Claim C.3 actually shows that #p-MC(Π_1) and #p-MC(Σ_1) are interreducible from which we conclude the following.

Corollary C.7. *All problems considered in Chapter 6 and the current chapter are* $\#\mathsf{A}[2]$*-easy.*

C.2 Counting Vertex Sets matching to a Clique

Let Γ be the class of the following conjunctive queries

$$\gamma_k := x_1 \dots x_k \exists y_1 \dots \exists y_k : \bigwedge_{i=1}^{k} E x_i y_i \wedge \bigwedge_{1 \le i < j \le k} E y_i y_j$$

Recall that G is the set of all conjunctive queries over the signature of graphs without self-loops. Using Remark C.1 and Corollary C.7, we can state Lemma C.2 in terms of #Hom as follows.

Corollary C.8. *The problem* #Hom(G) *is* $\#\mathsf{A}[2]$*-equivalent.*

In Chapter 6.2.1 we proved that #cp-Hom(Δ) reduces to #Hom(Δ) whenever Δ contains only minimal queries. The next lemma states that the reverse direction holds unconditionally.

Lemma C.9. *Let (H, X) be a graphical conjunctive query. Then there is an algorithm $\mathbb{A}$ with oracle access to $\#\mathsf{cp\text{-}Hom}(H, X \to \star)$ that computes $\#\mathsf{Hom}(H, X \to \star)$. Furthermore $\mathbb{A}$ runs in time $O(f(|H, X|) \cdot n^{O(1)})$ for some computable function f.*

Proof. Given a graph G for which we want to compute $\#\mathsf{Hom}(H, X \to G)$, we construct an H-colored graph G' as follows:

We first copy the vertex set $V(G)$ exactly $\#V(H)$ times and color the copies according to the vertices of H. If we write $v(G')$ for the set of vertices in G' whose color is $v \in V(G)$, Hence the vertices of G' are partitioned by

$$\mathcal{P} = \{v(G') \mid v \in V(H)\}.$$

Now let $\{u, v\}$ be an edge of H. Then, for every $a \in u(H')$ and $b \in v(H')$ we add the edge $\{a, b\}$ to G' if and only if (the initial vertices) a and b have been adjacent in G.

Note that this reduction yields indeed a H-colored graph, except for the case that G contains no edge. In this case, however, we can compute the number $\#\mathsf{Hom}(H, X \to G)$ by brute-force in linear time in $|V(G)|$. Otherwise, it can easily be verified that

$$\#\mathsf{Hom}(H, X \to G) = \#\mathsf{cp\text{-}Hom}(H, X \to G')\,. \qquad \blacksquare$$

The next lemma is required as a starting point for the reduction in Chapter 6.3.5. The proof is a straight-forward application of the minor reduction of Chapter 6.2.1.

Lemma C.10. *$\#\textsc{cp-Hom}(\Gamma)$ is $\#\mathsf{A}[2]$-hard.*

Proof. Given γ_{2k}, we can contract y_i to x_i for $i = 1, \ldots, k$ and then delete $x_{k+1}, \ldots, x_{2k}$. The resulting minor is the conjunctive query with k free variables and k quantified variables containing every edge between two vertices. It can easily be seen that every query with $\leq k$ free and $\leq k$ quantified variables is a minor of this query. Hence, by Lemma 6.10 and Corollary C.8, we have that $\#\textsc{cp-Hom}(\Gamma)$ is $\#\mathsf{A}[2]$-hard. $\blacksquare$

C.3 Proof of Lemma 6.26: Hardness for Grates

In what follows we formally prove the reduction required for Lemma 6.26.

Lemma C.11. *Let ω_k be the k-grate. Then there exists an algorithm $\mathbb{A}$ with oracle access to $\#\mathsf{cp\text{-}Hom}(\omega_k \to \star)$ that computes $\#\mathsf{cp\text{-}Hom}(\gamma_k \to \star)$. Furthermore $\mathbb{A}$ runs in time $O(k^2 \cdot n^2)$, where n is the number of vertices of the input graph.*

Proof. We modify the construction in the proof of Lemma 2.45 which was based on existing hardness proofs for variants of the grid tiling problem (cf. [45, Chapt. 14.4.1] and [39, Problem 1.12]).

Let (H_γ, X) be the query graph of γ_k. Without loss of generality we assume that the k free variables of γ_k are labeled $x^0_{k-1}, x^1_{k-2}, \ldots, x^{k-1}_0$, that the k quantified variables of γ_k are labeled $y^0_{k-1}, y^1_{k-2}, \ldots, y^{k-1}_0$ and that the atoms of γ_k are of the form $Ey^i_j x^i_j$. Then it is well-defined to write (H_ω, X) for the query graph of ω_k, i.e., the free variables coincide. Furthermore we have that the quantified variables of γ_k are a subset of the quantified variables of ω_k. In particular it holds that $V(H_\gamma) \subseteq V(H_\omega)$. While we give a formal construction of the reduction in the remainder of the proof, we encourage the reader to first consider the example in Figure C.1 to get an intuition.

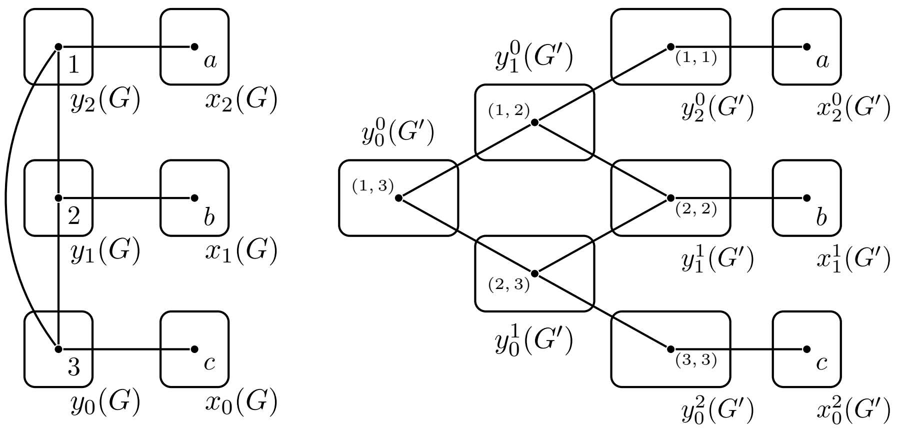


Figure C.1: Illustration of the construction of G' for $k = 3$. The graph G (*left*) is γ_3-colored and the mapping $a = \{x_2^0 \mapsto \alpha, x_1^1 \mapsto \beta, x_0^2 \mapsto \gamma\}$ is contained in $\mathsf{cp\text{-}Hom}(H_\gamma, X \to G)$. The graph G' (*right*) is ω_3-colored and a is contained in $\mathsf{cp\text{-}Hom}(H_\omega, X \to G)$ as well.

Now let G be a γ_k-colored graph for which we want to compute the number $\#\mathsf{cp\text{-}Hom}(H_\gamma, X \to G)$. Moreover we let $E^Y(G)$ denote the subset of edges of G that connect two vertices colored with quantified variables of γ_k. We construct a ω_k-colored graph G' as follows:

1. For every pair i, j with $i+j = k-1$ we add the vertices in $x_j^i(G)$ to G' and preserve the color.

2. For every pair i, j with $i + j = k - 1$ and for every vertex $u \in y_j^i(G)$ we add the vertex (u, u) to G' and preserve its color.

3. For every edge $\{u, v\} \in E(G)$ such that u is colored with y_j^i and v is colored with $x_{j'}^{i'}$ for some i, j, i', j', we add the edge $\{(u, u), v\}$ to G'.

4. For every pair i, j with $i + j < k - 1$ we add vertices
$$\{(u, u') \mid \{u, u'\} \in E^Y(G)\},$$
and color them with y_j^i. Note that this yields two vertices (u, u') and (u', u) for every edge $\{u, u'\}$ between vertices that are colored with quantified variables in G.

5. For every pair i, j with $i+j < k$ and $j < k-1$, we add an edge between $(u, u') \in y_j^i(G')$ and $(v, v') \in y_{j+1}^i(G')$ if and only if $u = v$.

6. Similarly, for every pair i, j with $i + j < k$ and $i < k - 1$, we add an edge between $(u, u') \in y_j^i(G')$ and $(v, v') \in y_j^{i+1}(G')$ if and only if $u' = v'$.

Now G' is ω_k-colored and we claim that

Claim C.12. $\mathsf{cp\text{-}Hom}(H_\gamma, X \to G) = \mathsf{cp\text{-}Hom}(H_\omega, X \to G')$.

Proof. For the first direction let $a : X \to V(G)$ in $\mathsf{cp\text{-}Hom}(H_\gamma, X \to G)$. We write $v(G)$ for the set of vertices in G that have color v. As a can be extended to a homomorphism h that satisfies $h(v) \in v(G)$ for all $v \in V(H_M)$, it holds that $a(x^i_j) \in x^i_j(G)$ for every pair i, j with $i + j = k - 1$. Now let $u^i_j = h(y^i_j)$ for every pair i, j with $i + j = k - 1$. A homomorphism $h' : V(H_\omega) \to V(G')$ is constructed as follows.

(1) h' coincides with h (or a, respectively) on X,

(2) $h'(y^i_j) = (u^i_j, u^i_j)$ for every pair i, j with $i + j = k - 1$, and

(3) $h'(y^i_j) = (h(y^i_{j+1}), h(y^{i+1}_j))$ for every pair i, j with $i + j < k - 1$.

Note that Step (3) is well-defined as the image of vertices in H_γ corresponding to quantified variables is a clique (of size k) in G, because otherwise h would be no homomorphism. By construction of G' it holds that h' is a homomorphism satisfying $h'(v) \in v(G')$ for every $v \in V(H_\omega)$. Hence we have that $a \in \mathsf{cp\text{-}Hom}(H_\omega, X \to G')$.

For the other direction let $a : X \to V(G')$ in $\mathsf{cp\text{-}Hom}(H_\omega, X \to G')$ and let $h' : V(H_\omega) \to V(G')$ be the homomorphism extending a satisfying that $h'(v) \in v(G')$ for every $v \in V(H_\omega)$. Now let $(u^i_j, u^i_j) = h'(y^i_j)$ for every pair i, j with $i + j = k - 1$. A homomorphism $h : V(H_\gamma) \to V(G)$ is constructed as follows.

(1) h coincides with h' (or a, respectively) on X, and

(2) $h(y^i_j) = u^i_j$ for every pair i, j with $i + j = k - 1$.

Now it can easily be seen that h is indeed a homomorphism: There must be an edge between every pair of vertices $h(y^i_j)$ and $h(y^{i'}_{j'})$ for $i+j = i'+j' = k-1$ as otherwise there would be no path through the half-grid in G' connecting vertices (u^i_j, u^i_j) and $(u^{i'}_{j'}, u^{i'}_{j'})$. Hence $a \in \mathsf{cp\text{-}Hom}(H_\gamma, X \to G)$. □

We conclude that $\mathbb{A}$ just constructs G' from G, which takes time $O(k^2 \cdot n^2)$, and then queries the oracle for G'. ■

Proof (of Lemma 6.26). Follows by #A[2]-hardness of #CP-HOM(Γ) as given by Lemma C.10 and the reduction given by Lemma C.11. ■

Bibliography

[1] Amir Abboud, Karl Bringmann, Holger Dell, and Jesper Nederlof. More consequences of falsifying SETH and the orthogonal vectors conjecture. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, pages 253–266, 2018. doi:10.1145/3188745.3188938.

[2] Amir Abboud, Karl Bringmann, Danny Hermelin, and Dvir Shabtay. Seth-Based Lower Bounds for Subset Sum and Bicriteria Path. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 41–57, 2019. doi:10.1137/1.9781611975482.3.

[3] Shreeram S. Abhyankar. *Lectures on algebra. Vol. I.* World Scientific Publishing Co. Pte. Ltd., Hackensack, NJ, 2006. doi:10.1142/9789812773449.

[4] Serge Abiteboul, Richard Hull, and Victor Vianu. *Foundations of Databases.* Addison-Wesley, 1995. URL: http://webdam.inria.fr/Alice/.

[5] Stefan Arnborg, Derek G. Corneil, and Andrzej Proskurowski. Complexity of Finding Embeddings in a k-Tree. *SIAM J. on Algebraic Discrete Methods*, 8(2):277–284, 1987. doi:10.1137/0608024.

[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] Miriam Backens. A Complete Dichotomy for Complex-Valued Holant^c. In *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018, July 9-13, 2018, Prague, Czech Republic*, pages 12:1–12:14, 2018. doi:10.4230/LIPIcs.ICALP.2018.12.

[8] Nikhil Bansal and Ryan Williams. Regularity Lemmas and Combinatorial Algorithms. *Theory of Computing*, 8(1):69–94, 2012. doi:10.4086/toc.2012.v008a004.

[9] Alexander I. Barvinok. *Combinatorics and Complexity of Partition Functions*, volume 30 of *Algorithms and combinatorics.* Springer, 2016. doi:10.1007/978-3-319-51829-9.

[10] Umberto Bertelè and Francesco Brioschi. *Nonserial dynamic programming.* Academic Press, 1972.

[11] M.R. Best, Peter van Emde Boas, and Hendrik W. Lenstra Jr. A sharpened version of the Aanderaa-Rosenberg conjecture. *Stichting Mathematisch Centrum. Zuivere Wiskunde*, ZW 30/74:1–20, 1974.

[12] Norman Biggs. *Algebraic graph theory.* Cambridge Mathematical Library. Cambridge University Press, Cambridge, second edition, 1993. doi:10.1017/CBO9780511608704.

[13] Andreas Björklund, Holger Dell, and Thore Husfeldt. The Parity of Set Systems Under Random Restrictions with Applications to Exponential Time Problems. In *Automata, Languages, and Programming - 42nd International Colloquium, ICALP 2015, Kyoto, Japan, July 6-10, 2015, Proceedings, Part I*, pages 231–242, 2015. doi:10.1007/978-3-662-47672-7‘‘19.

[14] Hans L. Bodlaender. A Linear-Time Algorithm for Finding Tree-Decompositions of Small Treewidth. *SIAM J. Comput.*, 25(6):1305–1317, 1996. doi:10.1137/S0097539793251219.

[15] 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.

[16] Cornelius Brand, Holger Dell, and Marc Roth. Fine-Grained Dichotomies for the Tutte Plane and Boolean #CSP. *Algorithmica*, 81(2):541–556, 2019. doi:10.1007/s00453-018-0472-z.

[17] 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.

[18] Glen E. Bredon. *Introduction to compact transformation groups*, volume 46. Academic press, 1972.

[19] Graham R. Brightwell and Peter Winkler. Note on Counting Eulerian Circuits. *CoRR*, cs.CC/0405067, 2004. URL: https://arxiv.org/abs/cs/0405067v1.

[20] Jin-Yi Cai, Zhiguo Fu, Heng Guo, and Tyson Williams. A Holant Dichotomy: Is the FKT Algorithm Universal? In *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*, pages 1259–1276, 2015. doi:10.1109/FOCS.2015.81.

[21] Jin-Yi Cai, Sangxia Huang, and Pinyan Lu. From Holant to #CSP and Back: Dichotomy for Holant^c Problems. *Algorithmica*, 64(3):511–533, 2012. doi:10.1007/s00453-012-9626-6.

[22] 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.

[23] Jin-Yi Cai, Pinyan Lu, and Mingji Xia. Holographic Algorithms with Matchgates Capture Precisely Tractable Planar #CSP. *SIAM J. Comput.*, 46(3):853–889, 2017. doi:10.1137/16M1073984.

[24] Jin-Yi Cai, Pinyan Lu, and Mingji Xia. Dichotomy for Real Holant^c Problems. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018, New Orleans, LA, USA, January 7-10, 2018*, pages 1802–1821, 2018. doi:10.1137/1.9781611975031.118.

[25] Liming Cai and David W. Juedes. On the existence of subexponential parameterized algorithms. *J. Comput. Syst. Sci.*, 67(4):789–807, 2003. doi:10.1016/S0022-0000(03)00074-6.

[26] Chris Calabro, Russell Impagliazzo, and Ramamohan Paturi. The Complexity of Satisfiability of Small Depth Circuits. In *Parameterized and Exact Computation, 4th International Workshop, IWPEC 2009, Copenhagen, Denmark, September 10-11, 2009, Revised Selected Papers*, pages 75–85, 2009. doi:10.1007/978-3-642-11269-0‘‘6.

[27] Amit Chakrabarti, Subhash Khot, and Yaoyun Shi. Evasiveness of Subgraph Containment and Related Properties. *SIAM J. Comput.*, 31(3):866–875, 2001. doi:10.1137/S0097539700382005.

[28] Ashok K. Chandra and Philip M. Merlin. Optimal Implementation of Conjunctive Queries in Relational Data Bases. In *Proceedings of the 9th Annual ACM Symposium on Theory of Computing, May 4-6, 1977, Boulder, Colorado, USA*, pages 77–90, 1977. doi:10.1145/800105.803397.

[29] Chandra Chekuri and Julia Chuzhoy. Polynomial Bounds for the Grid-Minor Theorem. *J. ACM*, 63(5):40:1–40:65, 2016. doi:10.1145/2820609.

[30] Chandra Chekuri and Anand Rajaraman. Conjunctive query containment revisited. *Theoretical Computer Science*, 239(2):211–229, 2000. doi:10.1016/S0304-3975(99)00220-0.

[31] Hubie Chen and Stefan Mengel. A Trichotomy in the Complexity of Counting Answers to Conjunctive Queries. In *18th International Conference on Database Theory, ICDT 2015, March 23-27, 2015, Brussels, Belgium*, pages 110–126, 2015. doi:10.4230/LIPIcs.ICDT.2015.110.

[32] 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.

[33] 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. *Inf. Comput.*, 201(2):216–231, 2005. doi:10.1016/j.ic.2005.05.001.

[34] Jianer Chen, Xiuzhen Huang, Iyad A. Kanj, and Ge Xia. Strong computational lower bounds via parameterized complexity. *J. Comput. Syst. Sci.*, 72(8):1346–1367, 2006. doi:10.1016/j.jcss.2006.04.007.

[35] 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.

[36] Julia Chuzhoy. Excluded Grid Theorem: Improved and Simplified. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 645–654, 2015. doi:10.1145/2746539.2746551.

[37] Julia Chuzhoy and Zihan Tan. Towards Tight(er) Bounds for the Excluded Grid Theorem. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 1445–1464, 2019. doi:10.1137/1.9781611975482.88.

[38] Bruno Courcelle. Graph Rewriting: An Algebraic and Logic Approach. In *Handbook of Theoretical Computer Science, Volume B: Formal Models and Sematics (B)*, pages 193–242. Elsevier Science, 1990.

[39] 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/.

[40] 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.

[41] 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.

[42] 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, STACS2017, March 8-11, 2017, Hannover, Germany*, pages 25:1–25:15, 2017. doi:10.4230/LIPIcs.STACS.2017.25.

[43] 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.

[44] 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.

[45] 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.

[46] 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.

[47] Holger Dell, Thore Husfeldt, Dániel Marx, Nina Taslaman, and Martin Wahlen. Exponential Time Complexity of the Permanent and the Tutte Polynomial. *ACM Trans. Algorithms*, 10(4):21:1–21:32, 2014. doi:10.1145/2635812.

[48] Holger Dell, Marc Roth, and Philip Wellnitz. Counting Answers to Existential Questions. In *46th International Colloquium on Automata, Languages, and Programming, ICALP 2019, July 9-12, 2019, Patras, Greece.*, pages 113:1–113:15, 2019. doi:10.4230/LIPIcs.ICALP.2019.113.

[49] Holger Dell, Marc Roth, and Philip Wellnitz. Counting Answers to Existential Questions. *CoRR*, abs/1902.04960, 2019. URL: http://arxiv.org/abs/1902.04960, arXiv:1902.04960.

[50] Josep Díaz, Maria J. Serna, and Dimitrios M. Thilikos. Counting H-colorings of partial k-trees. *Theor. Comput. Sci.*, 281(1-2):291–309, 2002. doi:10.1016/S0304-3975(02)00017-8.

[51] Reinhard Diestel, Tommy R. Jensen, Konstantin Yu. Gorbunov, and Carsten Thomassen. Highly Connected Sets and the Excluded Grid Theorem. *J. Comb. Theory, Ser. B*, 75(1):61–73, 1999. doi:10.1006/jctb.1998.1862.

[52] Julian Dörfler, Marc Roth, Johannes Schmitt, and Philip Wellnitz. Counting Induced Subgraphs: An Algebraic Approach to #W[1]-hardness. In *44th International Symposium on Mathematical Foundations of Computer Science, MFCS 2019, August 26-30, 2019, Aachen, Germany, to appear*, 2019.

[53] Julian Dörfler, Marc Roth, Johannes Schmitt, and Philip Wellnitz. Counting Induced Subgraphs: An Algebraic Approach to #W[1]-hardness. *CoRR*, abs/1904.10479, 2019. URL: https://arxiv.org/abs/1904.10479, arXiv:1904.10479.

[54] Rodney G. Downey and Michael R. Fellows. Fixed parameter tractability and completeness. In *Complexity Theory: Current Research, Dagstuhl Workshop, February 2-8, 1992*, pages 191–225, 1992.

[55] Rodney G. Downey and Michael R. Fellows. Fixed-Parameter Tractability and Completeness I: Basic Results. *SIAM J. Comput.*, 24(4):873–921, 1995. doi:10.1137/S0097539792228228.

[56] Rodney G. Downey and Michael R. Fellows. Fixed-Parameter Tractability and Completeness II: On Completeness for W[1]. *Theor. Comput. Sci.*, 141(1&2):109–131, 1995. doi:10.1016/0304-3975(94)00097-3.

[57] Rodney G. Downey and Michael R. Fellows. *Fundamentals of Parameterized Complexity.* Texts in Computer Science. Springer, 2013. doi:10.1007/978-1-4471-5559-1.

[58] Arnaud Durand and Stefan Mengel. Structural Tractability of Counting of Solutions to Conjunctive Queries. *Theory Comput. Syst.*, 57(4):1202–1249, 2015. doi:10.1007/s00224-014-9543-y.

[59] Martin E. Dyer, Leslie Ann Goldberg, and Mark Jerrum. An approximation trichotomy for Boolean #CSP. *J. Comput. Syst. Sci.*, 76(3-4):267–277, 2010. doi:10.1016/j.jcss.2009.08.003.

[60] Allan L. Edmonds. Introduction to Transformation Groups. 2010.

[61] Jack Edmonds. Paths, Trees, and Flowers. *Canadian Journal of Mathematics*, 17:449–467, 1965. doi:10.4153/CJM-1965-045-4.

[62] Jirí Fiala and Jan Kratochvíl. Locally constrained graph homomorphisms - structure, complexity, and applications. *Computer Science Review*, 2(2):97–111, 2008. doi:10.1016/j.cosrev.2008.06.001.

[63] Jörg Flum and Martin Grohe. Fixed-Parameter Tractability, Definability, and Model-Checking. *SIAM J. Comput.*, 31(1):113–145, 2001. doi:10.1137/S0097539799360768.

[64] Jörg Flum and Martin Grohe. The Parameterized Complexity of Counting Problems. *SIAM J. Comput.*, 33(4):892–922, 2004. doi:10.1137/S0097539703427203.

[65] Jörg Flum and Martin Grohe. *Parameterized Complexity Theory.* Texts in Theoretical Computer Science. An EATCS Series. Springer, 2006. doi:10.1007/3-540-29953-X.

[66] Markus Frick and Martin Grohe. Deciding first-order properties of locally tree-decomposable structures. *J. ACM*, 48(6):1184–1206, 2001. doi:10.1145/504794.504798.

[67] François Le Gall. Powers of tensors and fast matrix multiplication. In *International Symposium on Symbolic and Algebraic Computation, ISSAC 2014, Kobe, Japan, July 23-25, 2014*, pages 296–303, 2014. doi:10.1145/2608628.2608664.

[68] Jiawei Gao, Russell Impagliazzo, Antonina Kolokolova, and Ryan Williams. Completeness for First-Order Properties on Sparse Structures with Algorithmic Applications. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 2162–2181, 2017. doi:10.1137/1.9781611974782.141.

[69] M. R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman, 1979.

[70] Leslie A. Goldberg and Mark Jerrum. Counting Unlabelled Subtrees of a Tree is #P-complete. *LMS Journal of Computation and Mathematics*, 3:117–124, 2000. doi:10.1112/S1461157000000243.

[71] Oded Goldreich. *Computational Complexity - A Conceptual Perspective.* Cambridge University Press, 2008.

[72] Heng Guo, Chao Liao, Pinyan Lu, and Chihao Zhang. Zeros of Holant problems: locations and algorithms. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 2262–2278, 2019. doi:10.1137/1.9781611975482.137.

[73] Rudolf Halin. S-functions for graphs. *J. Geom.*, 8(1-2):171–186, 1976.

[74] John E. Hopcroft and Robert E. Tarjan. A V^2 Algorithm for Determining Isomorphism of Planar Graphs. *Inf. Process. Lett.*, 1(1):32–34, 1971. doi:10.1016/0020-0190(71)90019-6.

[75] Sangxia Huang and Pinyan Lu. A Dichotomy for Real Weighted Holant Problems. *Computational Complexity*, 25(1):255–304, 2016. doi:10.1007/s00037-015-0118-3.

[76] Russell Impagliazzo and Ramamohan Paturi. On the Complexity of k-SAT. *J. Comput. Syst. Sci.*, 62(2):367–375, 2001. doi:10.1006/jcss.2000.1727.

[77] Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which Problems Have Strongly Exponential Complexity? *J. Comput. Syst. Sci.*, 63(4):512–530, 2001. doi:10.1006/jcss.2001.1774.

[78] Mark Jerrum. Counting Trees in a Graph is #P-Complete. *Inf. Process. Lett.*, 51(3):111–116, 1994. doi:10.1016/0020-0190(94)00085-9.

[79] 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.

[80] Mark Jerrum and Kitty Meeks. Some Hard Families of Parameterized Counting Problems. *TOCT*, 7(3):11:1–11:18, 2015. doi:10.1145/2786017.

[81] Mark Jerrum and Kitty Meeks. The parameterised complexity of counting even and odd induced subgraphs. *Combinatorica*, 37(5):965–990, 2017. doi:10.1007/s00493-016-3338-5.

[82] Mark Jerrum and Alistair Sinclair. Approximating the Permanent. *SIAM J. Comput.*, 18(6):1149–1178, 1989. doi:10.1137/0218077.

[83] Mark Jerrum and Alistair Sinclair. The Markov Chain Monte Carlo Method: An Approach to Approximate Counting and Integration. In Dorit S. Hochbaum, editor, *Approximation Algorithms for NP-hard Problems*, pages 482–520. PWS Publishing Co., Boston, MA, USA, 1997. URL: http://dl.acm.org/citation.cfm?id=241938.241950.

[84] Jakob Jonsson. *Simplicial Complexes of Graphs*, volume 1928 of *Lecture Notes in Mathematics*. Springer, Berlin, Heidelberg, 2008. doi:10.1007/978-3-540-75859-4.

[85] Jeff Kahn, Michael E. Saks, and Dean Sturtevant. A topological approach to evasiveness. *Combinatorica*, 4(4):297–306, 1984. doi:10.1007/BF02579140.

[86] Richard M. Karp. Reducibility Among Combinatorial Problems. In *Proceedings of a symposium on the Complexity of Computer Computations, held March 20-22, 1972, at the IBM Thomas J. Watson Research Center, Yorktown Heights, New York, USA*, pages 85–103, 1972. URL: http://www.cs.berkeley.edu/%7Eluca/cs172/karp.pdf.

[87] 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.

[88] Pieter W. Kasteleyn. Dimer Statistics and Phase Transitions. *Journal of Mathematical Physics*, 4(2):287–293, 1963. doi:10.1063/1.1703953.

[89] Ken-ichi Kawarabayashi and Yusuke Kobayashi. Linear min-max relation between the treewidth of H-minor-free graphs and its largest grid. In *29th International Symposium on Theoretical Aspects of Computer Science, STACS 2012, February 29th - March 3rd, 2012, Paris, France*, pages 278–289, 2012. doi:10.4230/LIPIcs.STACS.2012.278.

[90] Tsampikos Kottos and Uzy Smilansky. Periodic Orbit Theory and Spectral Statistics for quantum graphs. *Annals of Physics*, 274(1):76 – 124, 1999. doi:https://doi.org/10.1006/aphy.1999.5904.

[91] 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.

[92] Richard E. Ladner. On the structure of polynomial time reducibility. *J. ACM*, 22(1):155–171, 1975. doi:10.1145/321864.321877.

[93] Serge Lang. *Algebra (3. ed.)*. Addison-Wesley, 1993.

[94] Alexander Leaf and Paul D. Seymour. Tree-width and planar minors. *J. Comb. Theory, Ser. B*, 111:38–53, 2015. doi:10.1016/j.jctb.2014.09.003.

[95] László Lovász. *Large Networks and Graph Limits*, volume 60 of *Colloquium Publications*. American Mathematical Society, 2012. URL: http://www.ams.org/bookstore-getitem/item=COLL-60.

[96] Frank H. Lutz. Some Results Related to the Evasiveness Conjecture. *J. Comb. Theory, Ser. B*, 81(1):110–124, 2001. doi:10.1006/jctb.2000.2000.

[97] Dániel Marx, Paul D. Seymour, and Paul Wollan. Rooted grid minors. *Journal of Combinatorial Theory, Series B*, 122:428–437, 2017. doi:10.1016/j.jctb.2016.07.003.

[98] Rudolf Mathon. A Note on the Graph Isomorphism Counting Problem. *Inf. Process. Lett.*, 8(3):131–132, 1979. doi:10.1016/0020-0190(79)90004-8.

[99] Stephen B. Maurer. Matrix Generalizations of Some Theorems on Trees, Cycles and Cocycles in Graphs. *SIAM Journal on Applied Mathematics*, 30(1):143–148, 1976. doi:10.1137/0130017.

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

[101] 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.

[102] Carl A. Miller. Evasiveness of graph properties and topological fixed-point theorems. *Foundations and Trends in Theoretical Computer Science*, 7(4):337–415, 2013. doi:10.1561/0400000055.

[103] Juan Andrés Montoya and Moritz Müller. Parameterized Random Complexity. *Theory Comput. Syst.*, 52(2):221–270, 2013. doi:10.1007/s00224-011-9381-0.

[104] Jaroslav Nešetřil. Homomorphisms of derivative graphs. *Discrete Mathematics*, 1(3):257–268, 1971. doi:10.1016/0012-365X(71)90014-8.

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

[106] Robert Oliver. Fixed-point sets of group actions on finite acyclic complexes. *Commentarii Mathematici Helvetici*, 50(1):155–177, 1975. doi:10.1007/BF02565743.

[107] James G. Oxley. *Matroid theory*. Oxford University Press, 1992.

[108] Viresh Patel and Guus Regts. Deterministic Polynomial-Time Approximation Algorithms for Partition Functions and Graph Polynomials. *SIAM J. Comput.*, 46(6):1893–1919, 2017. doi:10.1137/16M1101003.

[109] Mihai Patrascu and Ryan Williams. On the Possibility of Faster SAT Algorithms. In *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2010, Austin, Texas, USA, January 17-19, 2010*, pages 1065–1075, 2010. doi:10.1137/1.9781611973075.86.

[110] J. Scott Provan and Michael O. Ball. The Complexity of Counting Cuts and of Computing the Probability that a Graph is Connected. *SIAM J. Comput.*, 12(4):777–788, 1983. doi:10.1137/0212053.

[111] Bruce A. Reed. *Tree Width and Tangles: A New Connectivity Measure and Some Applications*, pages 87–162. London Mathematical Society Lecture Note Series. Cambridge University Press, 1997. doi:10.1017/CBO9780511662119.006.

[112] Ronald L. Rivest and Jean Vuillemin. On Recognizing Graph Properties from Adjacency Matrices. *Theor. Comput. Sci.*, 3(3):371–384, 1976. doi:10.1016/0304-3975(76)90053-0.

[113] Neil Robertson and Paul D. Seymour. Graph minors. III. Planar tree-width. *J. Comb. Theory, Ser. B*, 36(1):49–64, 1984. doi:10.1016/0095-8956(84)90013-3.

[114] Neil Robertson and Paul D. Seymour. Graph minors. II. Algorithmic Aspects of Tree-Width. *J. Algorithms*, 7(3):309–322, 1986. doi:10.1016/0196-6774(86)90023-4.

[115] Neil Robertson and Paul D. Seymour. Graph minors. V. Excluding a planar graph. *J. Comb. Theory, Ser. B*, 41(1):92–114, 1986. doi:10.1016/0095-8956(86)90030-4.

[116] Neil Robertson, Paul D. Seymour, and Robin Thomas. Quickly Excluding a Planar Graph. *J. Comb. Theory, Ser. B*, 62(2):323–348, 1994. doi:10.1006/jctb.1994.1073.

[117] Arnold L. Rosenberg. On the Time Required to Recognize Properties of Graphs: A Problem. *SIGACT News*, 5(4):15–16, October 1973. doi:10.1145/1008299.1008302.

[118] Gian-Carlo Rota. On the foundations of combinatorial theory I. Theory of Möbius functions. *Zeitschrift für Wahrscheinlichkeitstheorie und verwandte Gebiete*, 2(4):340–368, 1964.

[119] 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.

[120] Marc Roth and Johannes Schmitt. Counting induced subgraphs: A Topological Approach to #W[1]-hardness. In *13th International Symposium on Parameterized and Exact Computation, IPEC 2018, August 20-24, 2018, Helsinki, Finland*, pages 24:1–24:14, 2018. doi:10.4230/LIPIcs.IPEC.2018.24.

[121] 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.

[122] Neil Sloane. The On-Line Encyclopedia of Integer Sequences. URL: http://oeis.org. 2019.

[123] Paul A. Smith. Fixed-point theorems for periodic transformations. *American Journal of Mathematics*, 63(1):1–8, 1941. URL: https://www.jstor.org/stable/2371271.

[124] Richard P. Stanley. *Enumerative Combinatorics: Volume 1*. Cambridge University Press, 2011.

[125] Harold N. V. Temperley and Michael E. Fisher. Dimer problem in statistical mechanics-an exact result. *The Philosophical Magazine: A Journal of Theoretical Experimental and Applied Physics*, 6(68):1061–1063, 1961. doi:10.1080/14786436108243366.

[126] Seinosuke Toda. PP is as Hard as the Polynomial-Time Hierarchy. *SIAM J. Comput.*, 20(5):865–877, 1991. doi:10.1137/0220053.

[127] 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.

[128] Leslie G. Valiant. The Complexity of Computing the Permanent. *Theor. Comput. Sci.*, 8:189–201, 1979. doi:10.1016/0304-3975(79)90044-6.

[129] Leslie G. Valiant. The Complexity of Enumeration and Reliability Problems. *SIAM J. Comput.*, 8(3):410–421, 1979. doi:10.1137/0208032.

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

[131] Tatyana van Aardenne-Ehrenfest and Nicolaas G. de Bruijn. Circuits and trees in oriented linear graphs. *Simon Stevin : Wis- en Natuurkundig Tijdschrift*, 28:203–217, 1951.

[132] Alan J. Weir. The Sylow subgroups of the symmetric groups. *Proc. Amer. Math. Soc.*, 6:534–541, 1955. doi:10.2307/2033425.

[133] Dominic J.A. Welsh. *Matroid theory.* Courier Corporation, 2010.

[134] Ryan Williams. A new algorithm for optimal 2-constraint satisfaction and its implications. *Theor. Comput. Sci.*, 348(2-3):357–365, 2005. doi:10.1016/j.tcs.2005.09.023.

[135] Ryan Williams. Non-uniform ACC circuit lower bounds. In *Proceedings of the 26th Annual IEEE Conference on Computational Complexity, CCC 2011, San Jose, California, USA, June 8-10, 2011*, pages 115–125, 2011. doi:10.1109/CCC.2011.36.

[136] Ryan Williams. Faster decision of first-order graph properties. In *Joint Meeting of the Twenty-Third EACSL Annual Conference on Computer Science Logic (CSL) and the Twenty-Ninth Annual ACM/IEEE Symposium on Logic in Computer Science (LICS), CSL-LICS '14, Vienna, Austria, July 14 - 18, 2014*, pages 80:1–80:6, 2014. doi:10.1145/2603088.2603121.

[137] Virginia Vassilevska Williams, Joshua R. Wang, Richard Ryan Williams, and Huacheng Yu. Finding Four-Node Subgraphs in Triangle Time. In *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, 2015. doi:10.1137/1.9781611973730.111.

[138] Virginia Vassilevska Williams and Ryan Williams. Finding, Minimizing, and Counting Weighted Subgraphs. *SIAM J. Comput.*, 42(3):831–854, 2013. doi:10.1137/09076619X.

[139] Virginia Vassilevska Williams and Ryan Williams. Subcubic Equivalences Between Path, Matrix, and Triangle Problems. *J. ACM*, 65(5):27:1–27:38, 2018. doi:10.1145/3186893.

[140] Mingji Xia, Peng Zhang, and Wenbo Zhao. Computational complexity of counting problems on 3-regular planar graphs. *Theor. Comput. Sci.*, 384(1):111–125, 2007. doi:10.1016/j.tcs.2007.05.023.

[141] Thomas Zaslavsky. The Möbius function and the characteristic polynomial. *Combinatorial geometries*, 29:114–138, 1987.

Index