Counting and Finding Homomorphisms is Universal for Parameterized Complexity Theory

Marc Roth
Cluster of Excellence (MMCI), Saarland Informatics Campus (SIC), Saarbrücken, Germany
mroth@mmci.uni-saarland.de

Philip Wellnitz
Max Planck Institute for Informatics, Saarland Informatics Campus (SIC), Saarbrücken, Germany
wellnitz@mpi-inf.mpg.de

Abstract

Counting homomorphisms from a graph H into another graph G is a fundamental problem of (parameterized) counting complexity theory. In this work, we study the case where *both* graphs H and G stem from given classes of graphs: $H \in \mathcal{H}$ and $G \in \mathcal{G}$. By this, we combine the *structurally restricted* version of this problem (where the class $\mathcal{G} = \top$ is the set of all graphs), with the *language-restricted* version (where the class $\mathcal{H} = \top$ is the set of all graphs). The structurally restricted version allows an exhaustive complexity classification for classes $\mathcal{H}$: Either we can count all homomorphisms in polynomial time (if the treewidth of $\mathcal{H}$ is bounded), or the problem becomes #W[1]-hard [Dalmau, Jonsson, Th.Comp.Sci'04]. In contrast, in this work, we show that the combined view most likely does not admit such a complexity dichotomy.

Our main result is a construction based on Kneser graphs that associates every problem P in #W[1] with two classes of graphs $\mathcal{H}$ and $\mathcal{G}$ such that the problem P is *equivalent* to the problem $\#\textsc{Hom}(\mathcal{H} \to \mathcal{G})$ of counting homomorphisms from a graph in $\mathcal{H}$ to a graph in $\mathcal{G}$. In view of Ladner's seminal work on the existence of NP-intermediate problems [J.ACM'75] and its adaptations to the parameterized setting, a classification of the class #W[1] in fixed-parameter tractable and #W[1]-complete cases is unlikely. Hence, obtaining a complete classification for the problem $\#\textsc{Hom}(\mathcal{H} \to \mathcal{G})$ seems unlikely. Further, our proofs easily adapt to W[1] and the problem of deciding whether a homomorphism between graphs exists.

In search of complexity dichotomies, we hence turn to special graph classes. Those classes include line graphs, claw-free graphs, perfect graphs, and combinations thereof, and F-colorable graphs for fixed graphs F: If the class $\mathcal{G}$ is one of those classes and the class $\mathcal{H}$ is closed under taking minors, then we establish explicit criteria for the class $\mathcal{H}$ that partition the family of problems $\#\textsc{Hom}(\mathcal{H} \to \mathcal{G})$ into polynomial-time solvable and #W[1]-hard cases. In particular, we can drop the condition of $\mathcal{H}$ being minor-closed for F-colorable graphs. As a consequence, we are able to lift the framework of graph motif parameters due to Curticapean, Dell and Marx [STOC'17] to F-colorable graphs and provide an exhaustive classification for the parameterized subgraph counting problem on F-colorable graphs. As a special case, we obtain an easy proof of the parameterized intractability result of the problem of counting k-matchings in bipartite graphs.

2012 ACM Subject Classification Theory of computation → Parameterized complexity and exact algorithms; Theory of computation → Problems, reductions and completeness; Mathematics of computing → Combinatorics; Mathematics of computing → Graph theory

Keywords and phrases Parameterized complexity theory, counting problems, graph homomorphisms, Kneser graphs

Acknowledgements We thank Karl Bringmann and Holger Dell for fruitful discussions and valuable feedback on early drafts of this work.

1 Introduction

Homomorphisms between Graphs

Given graphs H and G, a fundamental question is whether (or how often) we can "find" the graph H in the graph G. Depending on the application, different notions of "finding" the graph H are studied: In the classical *subgraph isomorphism* problem [14, 55] (also consult e.g. [2, 48]), the goal is to search for subgraphs of G that are *isomorphic* to the graph H. In contrast, allowing multiple vertices of the graph H to be mapped to the same vertex of G, only requiring edge relations to be preserved, we obtain a relaxation of the subgraph isomorphism problem called the *(graph) homomorphism* problem. The problem of finding (and by extension counting) homomorphisms in graphs has a tight connection to problems related to conjunctive queries in data bases [11, 20], as well as applications in e.g. artificial intelligence [27]. As it turns out, once we want to count all "occurrences" of the graph H in G, understanding the graph homomorphism problem is already enough to understand various other notions of "finding" graphs in other graphs: As Curticapean, Dell, and Marx [16] proved, counting e.g. isomorphic subgraphs is the same as counting *linear combinations* of graph homomorphisms. Hence, in this work, we focus on counting graph homomorphisms.

Formally, given two classes of graphs $\mathcal{H}$ and $\mathcal{G}$, the (decision) problem $\textsc{Hom}(\mathcal{H} \to \mathcal{G})$ is defined as follows: Given graphs $H \in \mathcal{H}$ and $G \in \mathcal{G}$, decide whether there is a mapping $h : V(H) \to V(G)$ such that for any edge uv in $V(H)$, the edge $h(u)h(v)$ exists in $V(G)$. The problem of finding graph homomorphisms, also called H-colorings, has been studied since the late 1970s and 1980s [32, 45, 1]. In its most general form, where both classes $\mathcal{H}$ and $\mathcal{G}$ contain all graphs (denoted by $\mathcal{H} = \mathcal{G} = \top$) the problem $\textsc{Hom}(\mathcal{H} \to \mathcal{G})$ is NP-complete: Checking whether a graph H admits a homomorphism to the complete graph on 3 vertices is equivalent to checking whether H is 3-colorable, a classical NP-hard problem (see e.g. [31]).

Motivated by the hardness in the general case, special cases of the problem $\textsc{Hom}(\mathcal{H} \to \mathcal{G})$ have been studied: The *language-restricted* version of the problem $\textsc{Hom}(\mathcal{H} \to \mathcal{G})$ only assumes the class $\mathcal{H} = \top$ to be the set of all graphs and restricts the class $\mathcal{G}$. Note that the above example of checking whether a graph is 3-colorable falls into this framework (by setting $\mathcal{G} = \{K_3\}$), so the problem $\textsc{Hom}(\top \to \mathcal{G})$ is NP-hard in general as well. However, if the class $\mathcal{G}$ contains only *bipartite* graphs, the problem $\textsc{Hom}(\top \to \mathcal{G})$ is solvable in polynomial time [38]. In fact, Hell and Nešetřil [38] also prove the following hardness result: If the class $\mathcal{G}$ contains a non-bipartite graph, the problem $\textsc{Hom}(\top \to \mathcal{G})$ is NP-hard. Together, this yields a *complexity dichotomy*: for any problem $\textsc{Hom}(\top \to \mathcal{G})$, we obtain its complexity just by looking at the class $\mathcal{G}$. Based on this dichotomy, Feder and Vardi conjectured [27] that a similar dichotomy is possible for constraint satisfaction problems; this Feder-Vardi-Conjecture was proved recently by Bulatov [9] and, independently, by Zhuk [59].

Having understood the problem $\textsc{Hom}(\top \to \mathcal{G})$, the focus shifted to understanding "the other side", that is the case where the class $\mathcal{G}$ contains all graphs instead. For this *structurally restricted* version of the problem $\textsc{Hom}(\mathcal{H} \to \mathcal{G})$, the success story continues: As Grohe [34] proved, if every graph in the graph class $\mathcal{H}$ contains only graphs for which the so-called *treewidth* is small,[1] then the problem $\textsc{Hom}(\mathcal{H} \to \top)$ is solvable in polynomial-time again. Otherwise, using the now classical "Excluded-Grid-Theorem" [50], the graph class $\mathcal{H}$ (very roughly speaking) contains something *looking like* a grid, and finding homomorphisms

[1] Strictly speaking, the graphs in the class $\mathcal{H}$ only need to be *homomorphically equivalent* to graphs with small treewidth.

from a grid in another graph is NP-hard, as it can be used to solve the classical clique detection problem (and finding a clique in turn is a classical NP-hard problem). In fact, Grohe's dichotomy is even stronger: It shows that from a *parametrized complexity* view, the problem $\textsc{Hom}(\mathcal{H} \to \top)$ is either what is called "fixed-parameter tractable" (solvable in time $f(|V(H)|) \cdot |V(G)|^{O(1)}$ for graphs $H \in \mathcal{H}$, $G \in \top$) or "W[1]-hard" (essentially a parametrized equivalent of NP-hardness). We formalize these notions later; also consult [49, 29, 18] for an in-depth introduction to parametrized complexity theory.

The Doubly Restricted Version of Counting Homomorphisms

A natural generalization of finding a solution to a problem is to *count* all solutions. From an algorithmic point of view, counting all solutions may be way harder than finding a solution: While finding a perfect matching in a graph has a classical polynomial-time algorithm, *counting* all perfect matchings is known to be #P-complete [57].

Formally, for two classes of graphs $\mathcal{H}$ and $\mathcal{G}$, the counting version of the homomorphism problem (denoted by $\#\textsc{Hom}(\mathcal{H} \to \mathcal{G})$) is defined as follows: Given graphs $H \in \mathcal{H}$ and $G \in \mathcal{G}$, compute the number of (graph) homomorphism from the graph H to the graph G. Similarly to the decision realm, the language-restricted version $\#\textsc{Hom}(\top \to \mathcal{G})$ has been studied in the context of the counting constraint satisfaction problem: The dichotomy theorem of Dyer and Greenhill implies that the problem $\#\textsc{Hom}(\top \to \mathcal{G})$ is #P-complete if the class $\mathcal{G}$ contains a graph with a connected component that is neither an isolated vertex with or without self-loop, nor a complete graph with all self-loops, nor a complete bipartite graph without self-loops [23]. Otherwise, the problem $\#\textsc{Hom}(\top \to \mathcal{G})$ is solvable in polynomial time (cf. [23, Lemma 4.1]). In a subsequent line of research, this classification was lifted to general counting constraint satisfaction problems [8, 24, 10].

The structurally restricted version of the graph homomorphism problem has been studied in the counting regime as well: A counting analogue of Grohe's dichotomy was established by Dalmau and Jonsson [19]. In [19] they prove that the counting problem $\#\textsc{Hom}(\mathcal{H} \to \top)$ is solvable in polynomial time if and only if there is a constant bound on the treewidth of the graphs in the class $\mathcal{H}$; otherwise the problem $\#\textsc{Hom}(\mathcal{H} \to \top)$ is complete for the class #W[1] (where the class #W[1] is the counting equivalent of W[1]).

Initiated by the breakthrough result by Curticapean, Dell, and Marx [16], a line of research [52, 53, 20] lifted the dichotomy of Dalmau and Jonnson [19] to all parameterized counting problems that can be expressed as linear combinations of homomorphisms, subsuming counting of subgraphs, counting of induced subgraphs and even counting of answers to existential first-order queries. This lifting technique is sometimes also called *complexity monotonicity*.

Counting Homomorphisms is Universal for W[1] and #W[1]

The previous results provide a surprisingly clean picture of the complexity landscape of the problems of finding and counting graph homomorphisms for both, the language-restricted and the structurally restricted version. However, none of the previous results are applicable for the *doubly restricted* version: Instead of restricting only $\mathcal{H}$ or $\mathcal{G}$, we consider the problem $\textsc{Hom}(\mathcal{H} \to \mathcal{G})$ where *both* classes are fixed. This can be seen as a special case of both the structurally restricted version and the language-restricted version. In particular, the known dichotomies only translate for certain pairs of classes $\mathcal{H}, \mathcal{G}$, leaving a wide gap in the complexity landscape to be explored. In particular, it is imaginable that for real-world instances, *both* graphs H and G have a certain structure that can be exploited. In fact,

we show that the doubly restricted version can express *any* problem in W[1] and #W[1], respectively. Intuitively, this means that if we want to understand any problem P in #W[1], we may instead consider an equivalent problem $\#\textsc{Hom}(\mathcal{H}_P \to \mathcal{G}_P)$. In particular, any algorithm or hardness obtained for $\#\textsc{Hom}(\mathcal{H}_P \to \mathcal{G}_P)$ directly translates to the original problem P.

▶ **Theorem 1.1** (Universality for W[1] and #W[1]). *For any problem P in* W[1]*, there are classes $\mathcal{H} = \mathcal{H}_P$ and $\mathcal{G} = \mathcal{G}_P$ such that*

$$P \equiv_{\mathrm{T}}^{\mathrm{fpt}} \textsc{Hom}(\mathcal{H} \to \mathcal{G}),$$

and for any problem P' in #W[1]*, there are classes $\mathcal{H}' = \mathcal{H}'_P$ and $\mathcal{G}' = \mathcal{G}'_P$ such that*

$$P' \equiv_{\mathrm{T}}^{\mathrm{fpt}} \#\textsc{Hom}(\mathcal{H}' \to \mathcal{G}'),$$

where $\equiv_{\mathrm{T}}^{\mathrm{fpt}}$ denotes interreducibility (sometimes also called equivalence) with respect to parameterized Turing-reductions.
The classes $\mathcal{H}$ and $\mathcal{H}'$ are recursively enumerable and the classes $\mathcal{G}$ and $\mathcal{G}'$ are recursive.

Theorem 1.1 in turn also makes a clear categorization of the problems $\textsc{Hom}(\mathcal{H} \to \mathcal{G})$ into "easy" (that is fixed-parameter tractable) and "hard" (that is W[1]-hard or #W[1]-hard) cases unlikely: A general partition of the class W[1] in fixed-parameter tractable and W[1]-complete problems is very unlikely as indicated by Ladner's seminal result [40] and its adaptation to the parameterized setting by Downey and Fellows [22]. A similar reasoning applies to #W[1].

Note that Theorem 1.1, in particular its consequences for the absence of parameterized dichotomies, are independent from the "non-dichotomy" results of [6] and [12], which rule out a P vs. NP/#P dichotomy for the structurally restricted versions: In [6], Bodirsky and Grohe prove a P vs. NP non-dichotomy by a modification of Ladners Theorem [40]; however, this has no direct implications from neither a parametrized complexity nor a counting complexity point of view. Independently, in [12], Chen, Thurley, and Weyer proved a similar result also for the counting version and hence obtained a P vs. #P non-dichotomy result; again, this has no direct implications for our setting.

Dichotomies for F-Colorable Graphs and König Graphs

Having established the doubly restricted version of the problem $\#\textsc{Hom}(\mathcal{H} \to \mathcal{G})$ as interesting in general, we proceed to demonstrate examples of both, (1) how existing complexity dichotomies translate to the doubly restricted setting, as well as (2) how we can exploit the existence of structure in both classes to obtain new complexity dichotomies for certain pairs of graph classes.

Note that if we fix a graph class $\mathcal{G}$ for which the corresponding language restricted problem $\#\textsc{Hom}(\top \to \mathcal{G})$ is already "easy", then the same is true for any graph class $\mathcal{H}$ and the problem $\#\textsc{Hom}(\mathcal{H} \to \mathcal{G})$.[2] While it may be possible to improve the running time of known algorithms for special classes $\mathcal{H}$ in such a case, in this work we focus on investigating classes $\mathcal{G}$ where the problem $\#\textsc{Hom}(\mathcal{H} \to \top)$ is hard. (Note further that a similar statement is true for classes $\mathcal{H}$ where the structurally-restricted problem $\#\textsc{Hom}(\mathcal{H} \to \top)$ is "easy".)

As a first example how known dichotomies can be adapted to yield new results for the doubly-restricted setting, we consider the case where the class $\mathcal{G} = \mathcal{G}_F$ is the set of all

[2] For instance if $\mathcal{G}$ is the class of all planar graphs, Eppstein [26] gave an fixed-parameter tractable algorithm for $\#\textsc{Hom}(\top \to \mathcal{G})$; a similar result is known even for classes of bounded local treewidth [30]. Hence, there are also fixed-parameter tractable algorithms solving $\#\textsc{Hom}(\mathcal{H} \to \mathcal{G})$ for any graph class $\mathcal{H}$.

F-colorable graphs for some fixed graph F.[3] For example, if F is chosen to be the graph consisting of a single edge, then the problem $\#\text{Hom}(\mathcal{H} \to \mathcal{G}_F)$ is the problem of counting homomorphisms from a graph $H \in \mathcal{H}$ to a bipartite graph G. As it turns out, it is possible to refine the dichotomy by Dalmau and Jonsson [19] for the case $\mathcal{G} = \top$ to work for F-colorable graphs as well:

▶ **Theorem 1.2.** *Let F be a graph, and let $\mathcal{H}$ be a recursively enumerable class of graphs.*
(1) *If the treewidth of the class $\mathcal{H} \cap \mathcal{G}_F$ is bounded, then the problem $\#\text{Hom}(\mathcal{H} \to \mathcal{G}_F)$ is polynomial-time solvable.*
(2) *Otherwise, the problem $\#\text{Hom}(\mathcal{H} \to \mathcal{G}_F)$ is $\#\mathsf{W}[1]$-complete.*

While the proof Theorem 1.2 is conceptually close to the proof by Dalmau and Jonsson [19], we can combine Theorem 1.2 with the aforementioned complexity monotonicty [16] to lift the result to the realm of counting subgraphs:

▶ **Theorem 1.3** (Intuitive version)**.** *Let F be a fixed graph and let $\mathcal{H}$ be a recursively enumerable class of graphs. Given a graph $H \in \mathcal{H}$ and a graph $G \in \mathcal{G}_F$, we wish to compute the number of subgraphs $\#\mathsf{Sub}(H \to G)$ of G that are isomorphic to H.*
(1) *If the matching number of $\mathcal{H} \cap \mathcal{G}_F$ is bounded then the problem $\#\text{Sub}(\mathcal{H} \to \mathcal{G})$ is polynomial-time solvable.*
(2) *Otherwise, the problem $\#\text{Sub}(\mathcal{H} \to \mathcal{G})$ is $\#\mathsf{W}[1]$-complete.*

Note that Theorem 1.3 subsumes a dichotomy for counting subgraphs in bipartite graphs and, in particular, Theorem 1.3 yields an alternative and *easy* proof of $\#\mathsf{W}[1]$-hardness of counting k-matchings in bipartite graphs [17]. Further, as an example of a new result which follows from Theorem 1.3, we obtain $\#\mathsf{W}[1]$-hardness for the problem of counting triangle packings in 3-colorable graphs: This problem asks, given parameter k and a 3-colorable graph G, to compute the number of possibilities to embed k vertex-disjoint triangles into G.

As an example for completely new insights gained in the doubly restricted setting, we consider the cases where the class $\mathcal{G} = \mathcal{L}$ is the set of *line graphs* and where the set $\mathcal{G} = ♔$ is the set of *König graphs*, respectively; where a König graph is a line graph of a bipartite graph.[4] König graphs are of particular interest, as they are a subset of the well-studied classes of perfect graphs [13], line graphs (of arbitrary graphs) and thus of the claw-free graphs [3]. Consequently, the hardness results we obtain for König graphs hold for the three previous classes of graphs as well.

Being a well-studied object for almost a whole century [58], line graphs have applications in both graph theory (see e.g. [13]), but also in algorithm design (see e.g. [44]). The first thorough study of *homomorphisms between line graphs* is due to Nešetřil [47]; in particular, Nešetřil gave criteria when a homomorphism from $L(H)$ to $L(G)$ corresponds to a homomorphism from H to G. We further motivate the study of line graphs (and by extension König graphs) by demonstrating that the problem of *finding* a homomorphism to a line graph is always fixed-parameter tractable:

[3] Observe that containment in the class $\mathcal{G}_F$ is in general not solvable in polynomial time: If F is the triangle then $\mathcal{G}_F$, if considered as language, is the 3-coloring problem. For this reason, we model the problem $\#\text{Hom}(\mathcal{H} \to \mathcal{G})$ as a *parameterized promise* problem; the formal definition is given in Section 2.
[4] We chose this terminology due to the fact that König's theorem states that line graphs of bipartite graphs are perfect (see e.g. [13]). The symbol ♔ is used since "König" is the German word for "King".

▶ **Theorem 1.4.** *The decision problems* $\textsc{Hom}(\top \to \mathcal{L})$ *and thus* $\textsc{Hom}(\top \to \text{♔})$ *are fixed-parameter tractable. In particular, given a graph* H *and a line graph* L*, it is possible to decide the existence of a homomorphism from* H *to* L *in time*

$$f(|V(H)|) \cdot O(|V(L)|^2),$$

for some computable function f *independent of* H *and* L*.*

As it turns out, in contrast, *counting* all homomorphisms to König graph is in general #W[1]-hard; specifically, we prove the following:

▶ **Theorem 1.5.** *Let* $\mathcal{H}$ *be a recursively enumerable class of graphs. If* $\mathcal{H}$ *has unbounded treewidth and is closed under taking minors, then the problem* $\#\textsc{Hom}(\mathcal{H} \to \text{♔})$ *is* #W[1]*-complete.*

As argued before, the choice of König graphs induces the following consequences for perfect and claw-free graphs.

▶ **Corollary 1.6.** *Let* $\mathcal{C}$ *be one of the classes of line-graphs, claw-free graphs or perfect graphs, or a non-empty union thereof. Further, let* $\mathcal{H}$ *be a recursively enumerable class of graphs.*

(1) *If the treewidth of the class* $\mathcal{H}$ *is bounded, then the problem* $\#\textsc{Hom}(\mathcal{H} \to \mathcal{C})$ *is solvable in polynomial time.*

(2) *Otherwise, if the class* $\mathcal{H}$ *is additionally minor-closed, the problem* $\#\textsc{Hom}(\mathcal{H} \to \mathcal{C})$ *is* #W[1]*-complete.*

Note that by restricting the graph class $\mathcal{H}$ in Theorem 1.5, we are able to give a more detailed view of what makes counting homomorphisms to König graphs *hard*. In particular, the explicit criterion established by Theorem 1.5 suggests that we may only hope for fast algorithms for classes $\mathcal{H}$ that are not minor-closed.

However, in general, Theorem 1.5 does not answer the question whether we can classify the problem $\#\textsc{Hom}(\mathcal{H} \to \text{♔})$ into easy and hard problems, hence we answer that question with the following *implicit* dichotomy:

▶ **Theorem 1.7.** *Let* $\mathcal{H}$ *be a recursively enumerable class of graphs. Then the problem* $\#\textsc{Hom}(\mathcal{H} \to \text{♔})$ *is either fixed-parameter tractable or* #W[1]*-complete under parameterized Turing-reductions.*

Technical Overview

For our universality result (Theorem 1.1), we rely on known results regarding homomorphisms between *Kneser graphs*. More precisely, we use a computable function that associates each integer $n \geq 3$ with a Kneser graph $\mathrm{K}(n)$ such that there are no homomorphisms between $\mathrm{K}(n)$ and $\mathrm{K}(m)$ whenever $n \neq m$. Now, given some problem $P \in \#\mathsf{W}[1]$, we use the existence of a certain parameterized weakly parsimonious reduction $\mathbb{A}$ from P to the problem of counting homomorphisms from Kneser graphs $\mathrm{K}(n)$. In particular, for any instance x of the problem P, we have an efficiently computable mapping to a pair of graphs $x \mapsto (H_x, G_x)$ such that $P(x)$ is equal (up to a normalizing factor) to the number of homomorphisms from the Kneser graph H_x to the graph G_x, the latter of which can be assumed to allow a homomorphism to H_x. The main idea is then to choose $\mathcal{H}$ as the set of all graphs H_x and $\mathcal{G}$ as the set of all graphs G_x. Then we prove that P and $\#\textsc{Hom}(\mathcal{K} \to \mathcal{G})$ are interreducible. While the reduction $P \leq_{\mathrm{T}}^{\mathrm{fpt}} \#\textsc{Hom}(\mathcal{K} \to \mathcal{G})$ is immediate, we consider the construction of the backward reduction as our main technical contribution.

In particular, consider a pair of graphs $(H_x, G_y) \in \mathcal{H} \times \mathcal{G}$. In order to obtain a reduction $\#\textsc{Hom}(\mathcal{K} \to \mathcal{G}) \leq_{\mathrm{T}}^{\mathrm{fpt}} P$, that is to compute the number $\#\mathsf{Hom}(H_x \to G_y)$, we need to construct an instance to the problem P. This is easy if both H_x and G_y indeed correspond to the same instance $z = x = y$. If H_x and G_y do not correspond to a common instance, however, we need information about $\#\mathsf{Hom}(H_x \to G_y)$ from *somewhere else*, as any oracle to the problem P is useless in this situation. In our case, the construction ensures that $\#\mathsf{Hom}(H_x \to G_y) = 0$ in this situation; but in order to obtain this equality (while also maintaining decodability of the original instance x), an involved construction using Kneser graphs seems to be required.

An even more fundamental (but easier to solve) challenge is to *reversibly* encode any string x into a part of the graph G_x in such a way, that the number of homomorphisms to the graph G_x changes in a controlled way (in our case the number of homomorphisms stays in fact the same). As our constructed Kneser graphs have a chromatic number of at least 3, encoding a string x is possible using a comparably simple construction using paths. Implicitly, this step as well relies on deep theory about Kneser graphs, in particular we rely on Lovász' seminal result [42] which asserts that H_x cannot be mapped homomorphically into a graph with low chromatic number.

For our dichotomy results, as advertised, from a technical point of view, obtaining Theorem 1.2 is a rather simple lifting exercise from the result in [19]; we obtain Theorem 1.3 by a rather straightforward application of complexity monotonicity [16].

In contrast, the analysis of the complexity of counting homomorphisms to König graphs is technically more involved. The proof of the explicit classification for minor-closed classes $\mathcal{H}$ (Theorem 1.5) uses a gadget construction that, intuitively, associates each graph with a König graph, while keeping the number of grid-like substructures stable. In view of the diverse applications of the Grid-Tiling Problem (see e.g. [18, Chapter 14.4.1]), the construction might yield further intractability results for counting problems on König graphs (and thus on claw-free and perfect graphs) and might hence be of independent interest.

Finally, the implicit and exhaustive classification for counting homomorphisms to König graphs (Theorem 1.7) relies on Whitney's Isomorphism Theorem for line graphs [58] which allows to express the number of homomorphisms from a graph H to a König graph G as a finite linear combination of homomorphisms of the form

$$\sum_F \lambda_F \cdot \#\mathsf{Hom}(F \to L^{-1}(G)),$$

where the graphs F only depend on H and $\#\mathsf{Hom}(F \to L^{-1}(G))$ is the number of homomorphisms from F to the primal graph of G. Theorem 1.7 then follows by complexity monotonicity [16] and the classification of counting homomorphisms to bipartite graph as given by Theorem 1.2.

Organization of the Paper

We start with an introduction to the concepts and notation used in this work (including formal definitions of (parametrized) promise problems and reductions between them) in Section 2.

The proof of Theorem 1.1 is presented in Section 3. For completeness, we provide a sketch of the hardness proof of $\#\textsc{Hom}(\mathcal{H} \to \top)$ in Appendix A. Continuing, in Section 4 we prove the dichotomy for counting homomorphisms and subgraphs in F-colorable graphs, some proofs are deferred to the appendix Appendix B. Finally, in Section 5, we present the new dichotomy for König graphs.

2 Preliminaries

We write $[n]$ to denote the set $\{1, \ldots, n\}$. Further, we assume the binary alphabet $\{0, 1\}$. In particular, we assume that numbers are encoded binary as well, which allows us to abuse notation and write $\mathbb{N} \subseteq \{0,1\}^*$. Given a function $g : A \times B \to C$ and an element $a \in A$, we write $g(a, \star)$ for the function which maps $b \in B$ to $g(a, b)$. Given a finite set A we write $|A|$ and $\#A$ for the cardinality of A. Given two functions $f : A \to B$ and $g : B \to C$, we write $f \circ g$ for their *composition* that maps $x \in A$ to $g(f(x)) \in C$. A *partition* of a set A is a set of non-empty and pairwise disjoint subsets of A, called *blocks*, whose union is A.

2.1 Graphs and Homomorphisms

We consider undirected simple graphs without self-loops (unless stated otherwise) and we assume that graphs are encoded by their adjacency matrices. Given a graph G, we write $V(G)$ and $E(G)$ for the vertices and edges of G. A graph is called *complete* or a *clique* if all vertices are pairwise adjacent. A *subgraph* of G is a graph obtained from G by deleting vertices (including adjacent edges) and edges; more precisely, the graph F is a subgraph of G if $F = (V, E)$ such that $V \subseteq V(G)$ and $E \subseteq E(G) \cap V^2$. The graph F is called a *proper subgraph* if $F \neq G$. A graph M is a *minor* of a graph G if M can be obtained by a sequence of edge-contraction from a subgraph of G. Here the contraction of an edge $e = \{u, v\}$ is the operation of adding a new vertex uv which is made adjacent to all vertices that have been adjacent to u or v. After that, the vertices u, v and possible self-loops and multi-edges are deleted. Given a subset S of vertices of G, the *induced subgraph* $G[S]$ has vertices S and edges $E(G) \cap S^2$. Given a partition ρ of $V(G)$, the *quotient graph* G/ρ of G is obtained from G by identifying every pair of vertices that are contained in the same block of ρ. After that, multiple edges are deleted. Note that this construction induces self-loops if there is an edge between two vertices in the same block. Adopting the notation of [16], we denote quotient graphs without self-loops as *spasms*.

Given graphs H and G, a *homomorphism* from H to G is a mapping $h : V(H) \to V(G)$ that preserves the adjacency of vertices, that is, for every edge $\{u, v\}$ in $E(H)$, the graph G has the edge $\{h(u), h(v)\} \in E(G)$. If a homomorphism h is injective, then it is called an *embedding*. If an embedding h additionally satisfies that for every edge $\{h(u), h(v)\}$ in $E(G)$ there is an edge $\{u, v\}$ in $E(H)$, then h is called a *strong embedding* and a strong embedding is called an *isomorphism* if it is bijective. Two graphs H and G are called *isomorphic* if there exists an isomorphism from H to G. In this paper, we (implicitly) only work on isomorphism classes of graphs; we abuse notation and write $H = G$ if H and G are isomorphic. In particular, we denote $\top$ for the set of all (isomorphism types of) graphs. A homomorphism from H to itself is called an *endomorphism*. Further, a bijective endomorphism is called an *automorphism*. We write $\mathsf{Hom}(H \to G)$, $\mathsf{Emb}(H \to G)$ and $\mathsf{StrEmb}(H \to G)$ for the set of all homomorphisms, embeddings and strong embeddings from H to G, respectively. Furthermore, we write $\mathsf{Aut}(H)$ for the set of automorphisms of H, $\mathsf{Sub}(H \to G)$ for the set of subgraphs of G that are isomorphic to H and $\mathsf{IndSub}(H \to G)$ for the set of induced subgraphs of G that are isomorphic to H.

Homomorphic Equivalence and Cores

Two graphs H and G are called *homomorphically equivalent* if there is both a homomorphism from H to G and a homomorphism from G to H. Clearly, homomorphic equivalence is an equivalence relation; further the following is known.

▶ **Lemma 2.1** (See e.g. Chapter 1.6 in [39]). *The minimal representative of an equivalence class (with respect to homomorphic equivalence) is unique up to isomorphisms.*

It is thus well-defined to speak of "the" minimal representative of an equivalence class; we say that such a graph is a *core*. An equivalent definition of a core is given by the following known result.

▶ **Lemma 2.2** (See e.g. Chapter 1.6 in [39]). *A graph H is a core if and only if it there is no homomorphism from H to a proper subgraph of H. In particular, every endomorphism of a core is an automorphism.*

Colorings and Graph Parameters

An *H-coloring* of a graph G is a homomorphism $c \in \mathsf{Hom}(G \to H)$. We say that a graph G is *H-colorable* or allows a coloring into H if the graph G has an H-coloring. In particular, given a positive integer $k \in \mathbb{N}$, we say that G is k-colorable if it allows a coloring into the complete graph with k vertices. Given a graph G together with an H-coloring c, we say that a homomorphism $h \in \mathsf{Hom}(H \to G)$ is *color-prescribed* if $c(h(v)) = v$ for all $v \in V(H)$. We write $\#\mathsf{cp\text{-}Hom}(H \to G)$ for the set of all color-prescribed homomorphisms from H to G.

The following three graph parameters are of particular importance in this paper. First, the *chromatic number* of a graph G is defined to be the smallest k such that G is k-colorable. Second, the *odd girth* is defined to be the length of the smallest odd cycle in a graph, and undefined if no odd cycle exists. Third, we rely on the graph parameter of *treewidth*. Intuitively, a graph G has small treewidth if it has a "tree-like structure". In particular, if a graph G has a small treewidth, then the graph G also has small "separators". These separators allow for efficient dynamic programming algorithms for a wide range of problems that are known to be hard in the unrestricted setting. However, as we need the treewidth of a graph only in a black-box manner, we defer the reader to the literature (e.g. Chapter 7 in [18]) for the formal definition and a detailed exposition.

Tensor Products of Graphs

Given two graphs G and A, the *tensor product* $G \times A$ is the graph with vertices $V(G) \times V(A)$, where $\times$ is the Cartesian product of sets. Two vertices (g, a) and (g', a') are adjacent in $G \times A$ if the edge $\{g, g'\}$ is in $E(G)$ and the edge $\{a, a'\}$ is in $E(A)$. Now let H be a fixed graph. It is well-known that the function $\#\mathsf{Hom}(H \to \star)$ is linear with respect to $\times$ and multiplication [43, Equation 5.30], that is,

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

A further well-known fact about the tensor product reads as follows.

▶ **Fact 2.3** (Folklore). Let G, A and F be graphs. If either one of G or A is F-colorable, then so is their tensor product $G \times A$.

2.2 Parameterized Counting Problems and Promises

Recall that the problem $\#\textsc{Hom}(\mathcal{H} \to \mathcal{G})$ asks, given graphs $H \in \mathcal{H}$ and $G \in \mathcal{G}$, to compute the number of homomorphisms from H to G. However, this definition is informal in the sense that it leaves out the specification of the output if the input is *invalid*, that is, if the graph H does not belong to the class $\mathcal{H}$ or the graph G does not belong to the class $\mathcal{G}$.

A naive option to solve this issue, is to require the output to be 0 if the input is invalid. However, in this case we exclude a plethora of interesting cases from our studies, as even seemingly trivial instances might encode NP-hard problems. Consider for example the class $\mathcal{G}$ of 3-colorable graphs. Following the naive option, the problem $\#\text{Hom}(\mathcal{H} \to \mathcal{G})$ becomes NP-hard even if the class $\mathcal{H}$ only containts the graph K_1 consisting of a single vertex, as it encodes the 3-colorability problem: An instance (K_1, G) of the problem $\#\text{Hom}(\mathcal{H} \to \mathcal{G})$ is mapped to zero if and only if the graph G is 3-colorable. In particular, fixed-parameter tractability of this problem would yield an algorithm running in time $f(|K_1|) \cdot |V(G)|^{O(1)}$ which is a polynomial in $|V(G)|$, and thus imply $\mathsf{P} = \mathsf{NP}$. In sharp contrast, the number of homomorphisms from the graph K_1 to any graph G is just the number of vertices $|V(G)|$ and thus the hardness of the problem $\#\text{Hom}(\mathcal{H} \to \mathcal{G})$ stems only from enforcing invalid inputs to be mapped to zero.

Another option of solving the issue of invalid instances of the problem $\#\text{Hom}(\mathcal{H} \to \mathcal{G})$ is as follows: If a given instance (H, G) consists of graphs $H \in \mathcal{H}$ and $G \in \mathcal{G}$, then we are supposed to compute the number of homomorphisms $\#\mathsf{Hom}(H \to G)$ correctly; otherwise, we may output *any* number. Formally, this requires us to model the problem $\#\text{Hom}(\mathcal{H} \to \mathcal{G})$ as a *promise problem*.[5] In what follows, we thus present a concise but self-contained introduction to parameterized promise (counting) problems.

A *parameterization* κ is a polynomial-time computable function from the set $\{0,1\}^*$ to the natural numbers $\mathbb{N}$. Note that the assumption of polynomial-time computable parameterizations for both, decision and counting problems, is common (see e.g. [29, Definitions 1.1 and 14.1]), but not standard. We refer the reader to the discussion of this issue in Chapter 1.2 in the textbook of Flum and Grohe [29].

▶ **Definition 2.4.** *A* parameterized promise counting problem *(or short "PPC problem") is a triple* (P, κ, Π) *consisting of a function* $P : \{0,1\}^* \to \mathbb{N}$*, a parameterization* $\kappa : \{0,1\}^* \to \mathbb{N}$*, and a promise* $\Pi \subseteq \{0,1\}^*$*. A* parameterized counting problem *(without promises) is a PPC problem with* $\Pi = \{0,1\}^*$*.*

A PPC problem (P, κ, Π) *is computable in time* t *if there exists a deterministic algorithm* $\mathbb{A}$ *that fulfills the following.*

1. *On input* $x \in \{0,1\}^*$*, the algorithm* $\mathbb{A}$ *runs in time* $t(|x|)$*.*
2. *On input* $x \in \Pi$*, the algorithm* $\mathbb{A}$ *outputs* $P(x)$*.*

In particular, we call the problem (P, κ, Π) fixed-parameter tractable *if there exists a computable function* f *such that the triple* (P, κ, Π) *can be computed in time* $f(\kappa(x)) \cdot |x|^{O(1)}$*.*

Further, we call $x \in \{0,1\}^*$ *an* instance *of the problem* (P, κ, Π) *and say that an instance* x *is* valid *if it is contained in the promise, that is* $x \in \Pi$*.*

Note that we obtain the standard definition of fixed-parameter tractability of (non-promise) counting problems if we set the promise Π to be $\{0,1\}^*$. Note further that parameterized decision problems with promises are obtained from Definition 2.4 by restricting the image of the function P to be $\{0,1\}$. In this case, Definition 2.4 coincides with the standard definition of (parameterized) promise problems (see e.g. Definition 3.1 in the full version [4] of [5]).

We consider the following family PPC problems.

[5] Goldreich [33, Chapter 2.4.1] states that *promise problems offer the most direct way of formulating natural computational problems.* Indeed, some of the most striking results in complexity theory implicitly rely on promise problems. Examples are "gap problems" and "uniqueness promises"; we refer the reader to [33, Chapter 2.4.1.2] for a discussion.

▶ **Definition 2.5.** *Let $\mathcal{H}$ and $\mathcal{G}$ be classes of graphs. The PPC problem $\#\textsc{Hom}(\mathcal{H} \to \mathcal{G})$ asks, given $H \in \mathcal{H}$ and $G \in \mathcal{G}$, to compute the number of homomorphisms $\#\mathsf{Hom}(H \to G)$; the parameter is $|V(H)|$. Formally, the promise of $\#\textsc{Hom}(\mathcal{H} \to \mathcal{G})$ is the set of all (encodings of) pairs $(H, G) \in \mathcal{H} \times \mathcal{G}$.*
Further, we define $\#\textsc{cp-Hom}(\mathcal{H} \to \mathcal{G})$ as the PPC problem of, given $H \in \mathcal{H}$, $G \in \mathcal{G}$ and an H-coloring of G, computing the number of color-prescribed homomorphisms $\#\mathsf{cp\text{-}Hom}(\mathcal{H} \to \mathcal{G})$; the parameter is $|V(H)|$. Again, the formal promise is defined as the set of all (encodings of) pairs $(H, G) \in \mathcal{H} \times \mathcal{G}$.

The *decision problems* $\textsc{Hom}(\mathcal{H} \to \mathcal{G})$ and $\textsc{cp-Hom}(\mathcal{H} \to \mathcal{G})$ are defined similarly, with the exception that the output is required to be 1 if the number of homomorphisms $\#\mathsf{Hom}(H \to G)$ is positive or the number of color-prescribed homomorphisms $\#\mathsf{cp\text{-}Hom}(H \to G)$ is positive, respectively.

▶ Remark 2.6. If membership of a graph in the class $\mathcal{G}$ can be tested in polynomial time and the class $\mathcal{H}$ is recursive, then there is no need to define the problem $\#\textsc{Hom}(\mathcal{H} \to \mathcal{G})$ as promise problem. Instead, we can define the output to be zero if a given pair (H, G) is not contained in $\mathcal{H} \times \mathcal{G}$; note that $H \in \mathcal{H}$ can be verified in time $f(H)$ for some computable function f as, by assumption, $\mathcal{H}$ is recursive.

Reductions and Hardness

In this paper, we consider the following two notions of reducibility for PPC problems.

▶ **Definition 2.7** (Parameterized (Weakly) Parsimonious Reductions)**.** *Let PPC problems (P, κ, Π) and (P', κ', Π') be given. A* parameterized weakly parsimonious reduction *from (P, κ, Π) to (P', κ', Π') is a pair of a deterministic algorithm $\mathbb{A}$ and a triple of computable functions (f, g, s) such that:*

1. *For all valid instances $x \in \Pi$, the algorithm $\mathbb{A}$ outputs a valid instance of (P', κ', Π'), that is $\mathbb{A}(x) \in \Pi'$.*
2. *We can compute $P(x)$ by computing P' on the computed instance $\mathbb{A}(x)$ and the function $g(x)$; in particular, we have that $P(x) = g(x) \cdot P'(\mathbb{A}(x))$.*
3. *The PPC problem (g, κ, Π) is fixed-parameter tractable.*
4. *On input $x \in \{0,1\}^*$, the algorithm $\mathbb{A}$ runs in time $f(\kappa(x)) \cdot |x|^{O(1)}$.*
5. *For all $x \in \{0,1\}$, the parameter of the instance $\mathbb{A}(x)$ is bounded by $s(\kappa(x))$, that is $\kappa'(\mathbb{A}(x)) \leq s(\kappa(x))$.*

We write $(P, \kappa, \Pi) \leq^{\text{w-fpt}} (P', \kappa', \Pi')$ if such a reduction exists. If g is the identity function on Π, then the reduction is called parsimonious *and we write $(P, \kappa, \Pi) \leq^{\text{fpt}} (P', \kappa', \Pi')$.*

▶ **Definition 2.8** (Parameterized Turing-reductions)**.** *Let (P, κ, Π) and (P', κ', Π') be PPC problems. A* parameterized Turing-reduction *from (P, κ, Π) to (P', κ', Π') is a pair of an algorithm $\mathbb{A}$ equipped with oracle access to the function P' and a pair (f, s) of computable functions such that:*

1. *On input $x \in \{0,1\}^*$, the algorithm $\mathbb{A}$ runs in time $f(\kappa(x)) \cdot |x|^{O(1)}$.*
2. *On input $x \in \Pi$, the algorithm $\mathbb{A}$ computes the function $P(x)$.*
3. *On input $x \in \Pi$, the algorithm $\mathbb{A}$ only queries the oracle on strings y with $y \in \Pi'$ and $\kappa'(y) \leq s(\kappa(x))$.*

We write $(P, \kappa, \Pi) \leq_{\text{T}}^{\text{fpt}} (P', \kappa', \Pi')$ if such a reduction exists.

Unsurprisingly, the previous notions of reducibility coincide with the common notions for reducibility between parameterized counting problems if the promises Π and Π' are trivial, that is,

if $\Pi = \Pi' = \{0,1\}^*$ (see e.g. [15, Definition 1.8]). Further, the following facts are straightforward to verify.

▶ **Fact 2.9.** Let (P, κ, Π) and (P', κ', Π') be PPC problems. We have that

$$(P, \kappa, \Pi) \leq^{\text{fpt}} (P', \kappa', \Pi') \implies (P, \kappa, \Pi) \leq^{\text{w-fpt}} (P', \kappa', \Pi') \implies (P, \kappa, \Pi) \leq_{\text{T}}^{\text{fpt}} (P', \kappa', \Pi').$$

▶ **Fact 2.10.** All of the notions of reducibility $\leq^{\text{fpt}}$, $\leq^{\text{w-fpt}}$, and $\leq_{\text{T}}^{\text{fpt}}$ are transitive.

▶ **Fact 2.11.** Let (P, κ, Π) and (P', κ', Π') be PPC problems and assume that (P, κ, Π) reduces to (P', κ', Π') with respect to any of $\leq^{\text{fpt}}$, $\leq^{\text{w-fpt}}$, or $\leq_{\text{T}}^{\text{fpt}}$. If (P', κ', Π') is fixed-parameter tractable, then (P, κ, Π) is also fixed-parameter tractable.

Evidence of fixed-parameter *in*tractability of parameterized counting problems (with promises) is given by hardness for the complexity class #W[1]. It is common to define #W[1] via the complete problem #Clique. The problem #Clique asks, given $k \in \mathbb{N}$ and a graph G, to compute the number of cliques of size k in G. Note that by Remark 2.6 the problem #Clique can be assumed to have no promise.

▶ **Definition 2.12** ([28, 46])**.** *The class* #W[1] *contains all parameterized counting problems without promises that can be reduced to* #Clique *by parameterized parsimonious reductions.*

Next, we extend the notion of #W[1]-hardness to PPC problems.

▶ **Definition 2.13.** *A PPC problem* (P, κ, Π) *is* #W[1]-hard *under parameterized parsimonious reductions if*

$$\#\textsc{Clique} \leq^{\text{fpt}} (P, \kappa, \Pi).$$

Hardness under $\leq^{\text{w-fpt}}$ *and* $\leq_{\text{T}}^{\text{fpt}}$ *is defined likewise. A parameterized counting problem without promises is* #W[1]-complete *if it is contained in* #W[1] *and* #W[1]*-hard.*

As $\#\textsc{Hom}(\mathcal{H} \to \mathcal{G})$ is a promise problem, it is formally not contained in #W[1]. However, it can be shown that $\#\textsc{Hom}(\mathcal{H} \to \mathcal{G})$ cannot be harder than #W[1]-complete problems:

▶ **Lemma 2.14.** *Let* $\mathcal{H}$ *and* $\mathcal{G}$ *be computable graph classes. We have that*

$$\#\textsc{Hom}(\mathcal{H} \to \mathcal{G}) \leq^{\text{fpt}} \#\textsc{Clique}.$$

Proof. It is known that $\#\textsc{Hom}(\mathcal{H} \to \top) \leq^{\text{fpt}} \#\textsc{Clique}$ by the more general result that #A[1] = #W[1] [29, Theorem 14.17]. The reduction $\#\textsc{Hom}(\mathcal{H} \to \mathcal{G}) \leq^{\text{fpt}} \#\textsc{Hom}(\mathcal{H} \to \top)$ is given by the identity function. ◀

As a concluding remark for this subsection, note that parameterized reductions, #Clique, and (hardness and completeness for) #W[1] have corresponding notions in the decision realm. In particular, Clique, that is, the problem of *deciding* the existence of a clique of size k, constitutes the canonical complete problem for W[1]. We refer the reader to the textbook of Flum and Grohe [29] for further details of parameterized decision complexity, as this work mainly deals with counting problems.

2.3 Quantum Graphs and Complexity Monotonicity

The framework of *Complexity Monotonicity* was recently introduced by Curticapean, Dell, and Marx in their breakthrough result regarding the complexity of the (induced) subgraph counting problem [16]. Very roughly speaking, the principle of complexity monotonicity states that

Computing a linear combination of homomorphism numbers is precisely as hard as computing its hardest term.

While linear combinations of homomorphisms have been modeled by so-called *graph motif parameters* in [16], we instead rely on the notion of *quantum graphs* as introduced by Lovász [43, Chapter 6].

▶ **Definition 2.15** (Quantum graphs). *A* quantum graph Q *is a formal linear combination of graphs with finite support. We write*

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

where λ_H is non-zero only for finitely many graphs. We write $\mathsf{supp}(Q)$ *for the set of all graphs H for which λ_H is non-zero. The elements of the support* $\mathsf{supp}(Q)$ *are called* constituents *of Q.*

Graph parameters extend to quantum graphs linearly. In particular, we define

$$\#\mathsf{Hom}(Q \to G) := \sum_H \lambda_H \cdot \#\mathsf{Hom}(H \to G). \tag{1}$$

Now, given a set $\mathcal{Q}$ of quantum graphs, we write $\mathsf{supp}(\mathcal{Q})$ for the set of all constituents of all quantum graphs in $\mathcal{Q}$. Furthermore, given a class $\mathcal{G}$ of graphs, the PPC problem $\#\textsc{Hom}(\mathcal{Q} \to \mathcal{G})$ is defined similarly as in case of (non-quantum) graphs: Given $(Q, G) \in \mathcal{Q} \times \mathcal{G}$, the goal is to compute the number $\#\mathsf{Hom}(Q \to G)$; the parameter is given by the description length $|Q|$ of Q.

The main result of Curticapean, Dell and Marx can be stated as follows:

▶ **Theorem 2.16** (Complexity Monotonicity [16]). *Let $\mathcal{Q}$ be a recursively enumerable class of quantum graphs. Then we have that*

$$\#\textsc{Hom}(\mathcal{Q} \to \top) \equiv_{\mathrm{T}}^{\mathrm{fpt}} \#\textsc{Hom}(\mathsf{supp}(\mathcal{Q}) \to \top).$$

The reduction $\#\textsc{Hom}(\mathcal{Q} \to \top) \leq_{\mathrm{T}}^{\mathrm{fpt}} \#\textsc{Hom}(\mathsf{supp}(\mathcal{Q}) \to \top)$ is trivial: Given a quantum graph Q and a graph G, we can compute the number $\#\mathsf{Hom}(Q \to G)$ as given by Equation (1). However, the other direction relies on a deep theory of Lovász [43, Chapters 5 and 6] and is given by the following lemma.

▶ **Lemma 2.17** (Lemma 3.6 in [16]). *Let Q be a quantum graph. There exists a deterministic algorithm $\mathbb{A}$ that is given oracle access to $\#\mathsf{Hom}(Q \to \star)$ and that, on input a graph G, computes the number $\#\mathsf{Hom}(H \to G)$ for every constituent H of Q. Furthermore, there exist computable functions f and s such that the running time of $\mathbb{A}$ is bounded by $f(|Q|) \cdot |V(G)|^{O(1)}$ and the number of vertices of every graph G' for which the oracle is queried, is bounded by $s(|Q|) \cdot |V(G)|$.*

In Section 4 we show that the previous lemma readily extends to the problem $\#\textsc{Hom}(\mathcal{Q} \to \mathcal{G}_F)$, where $\mathcal{G}_F$ is the set of all F-colorable graphs for some fixed graph F.

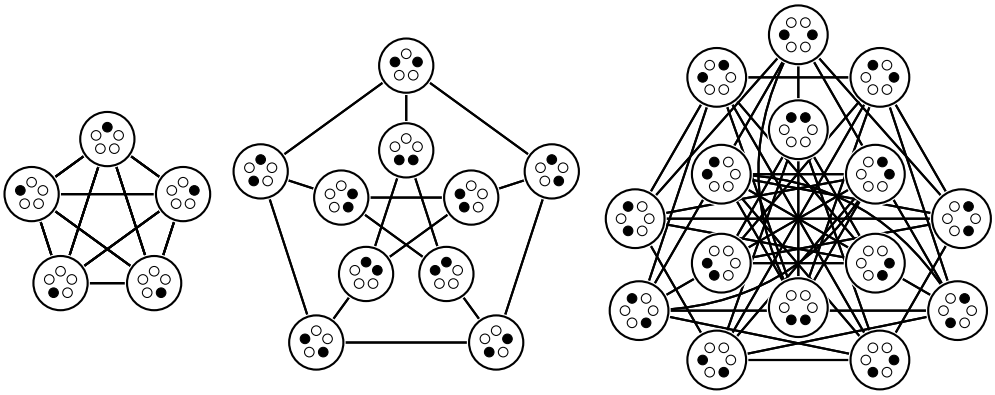

Figure 1 Examples for Kneser graphs: The Kneser graphs $K(5,1)$, $K(5,2)$, and $K(6,2)$; the graph $K(5,2)$ is also known as the *Petersen graph*

3 Counting and Finding Homomorphisms is Universal

In this part of the paper, we show that every parameterized counting problem in #W[1] is interreducible with a problem $\#\textsc{Hom}(\mathcal{H} \to \mathcal{G})$ with respect to parameterized Turing-reductions. Further, the proof shows that the analogous statement holds for (parameterized) decision problems in W[1] and a problem $\textsc{Hom}(\mathcal{H} \to \mathcal{G})$. The starting point is the following lemma; it follows from the standard hardness proof of $\#\textsc{Hom}(\mathcal{H} \to \top)$ for classes $\mathcal{H}$ of unbounded treewidth. We provide an exposition of the proof in Appendix A—see Lemma A.2.

▶ **Lemma 3.1.** *Let $\mathcal{H}$ be a computable class of connected cores of unbounded treewidth. Then the problem $\#\textsc{Hom}(\mathcal{H} \to \top)$ is #W[1]-hard under parameterized weakly parsimonious reductions. In particular, the images of the reductions can be assumed to contain only pairs (H, G) such that $H \in \mathcal{H}$ and G is connected and H-colorable.*

The remainder of this section is devoted to the proof of the following theorem.

▶ **Theorem 3.2.** *Let (F, κ) denote a problem in #W[1]. There are classes $\mathcal{H}$ and $\mathcal{G}$ such that*

$$(F, \kappa) \equiv_{\mathrm{T}}^{\mathrm{fpt}} \#\textsc{Hom}(\mathcal{H} \to \mathcal{G}).$$

Furthermore $\mathcal{H}$ is recursively enumerable and $\mathcal{G}$ is recursive.

In particular, we show that for any problem (F, κ) in #W[1], we can construct graph classes $\mathcal{H}$ and $\mathcal{G}$ such that we have for any graphs $H \in \mathcal{H}$ and $G \in \mathcal{G}$:

- If $\#\mathsf{Hom}(H \to G) \neq 0$, the pair (H, G) corresponds to exactly one instance x of the problem (F, κ), and we can obtain both x and $\kappa(x)$ from H and G.
- If $\#\mathsf{Hom}(H \to G) = 0$, the pair (H, G) does not correspond to an instance of (F, κ).

Counting Homomorphisms Between Kneser Graphs

By Lemma 3.1, the problem $\#\textsc{Hom}(\mathcal{H} \to \top)$ is #W[1]-hard for *any* (computable) class of connected cores. In particular, it is known that this is the case for (subclasses) of the class of *Kneser graphs*. Further, Kneser graphs have other nice properties which we exploit in the proof of Theorem 3.2. Formally, we start with the following definition; consider also Figure 1 for examples of Kneser graphs.

▶ **Definition 3.3** (see e.g. [36, Chapter 3]). *Given integers r and s, the* Kneser graph $\mathrm{K}(r,s)$ *is the graph that has as vertices the subsets of size s of $[r]$ and edges between two vertices if the corresponding sets are disjoint.*

Given a number $n \geq 3$, we set $\mathrm{K}(n) := \mathrm{K}((2n+1)(n-2), n(n-2))$. With this choice of the parameters r and s, we can use the following results; recall that the chromatic number of a graph G is the minimum k such that G allows a homomorphism to the complete graph of size k, and the odd girth of a graph is the length of the smallest cycle of odd length.

▶ **Fact 3.4** ([42] and Propositions 3.13, 3.14 in [36]). The graph $\mathrm{K}(n)$ has chromatic number n and odd girth $2n+1$. Furthermore, the graph $\mathrm{K}(n)$ is a core, that is, the graph $\mathrm{K}(n)$ is minimal with respect to homomorphic equivalence.

Note that by Lemma 2.2, the graph $\mathrm{K}(n)$ being a core implies that every endomorphism of $\mathrm{K}(n)$ is already an automorphism. Hence, the number of homomorphisms from the graph $\mathrm{K}(n)$ to itself is precisely the number of automorphisms $\#\mathsf{Aut}(\mathrm{K}(n))$.

▶ **Fact 3.5** (Folklore, see e.g. [56]). The graph $\mathrm{K}(r,s)$ is connected if $r > 2s$.
Hence, $\mathrm{K}(n)$ is connected.

An important property of Kneser graphs is the well-known fact that they constitute an antichain with respect to the homomorphism order. We provide a proof for convenience.

▶ **Lemma 3.6.** *Let n and m be distinct positive integers. Then we have*

$$\#\mathsf{Hom}(\mathrm{K}(m) \to \mathrm{K}(n)) = 0.$$

Proof. For every pair of graphs H and G with $\#\mathsf{Hom}(H \to G) \neq 0$, we have that the odd girth of H is bounded from below by the odd girth of G [39, Exercise 1.10.2]. Further, the chromatic number of H is bounded from above by the chromatic number of G [39, Proposition 1.8]. The lemma hence holds by Fact 3.4. ◀

Now, let $\mathcal{K}_{even}$ denote the set of all graphs $\mathrm{K}(n)$ with even n and let $\mathcal{K}_{odd}$ denote the set of all graphs $\mathrm{K}(n)$ with odd n.

Encoding Problems into Graphs Classes

A central tool for the proof of Theorem 3.2 is an encoding of arbitrary strings into graphs, which we discuss next. In particular, we use a disjoint union of paths for the encoding. To this end, let P_i be the path with i edges. Given a string $x = x[1]x[2]\cdots x[n] \in \{0,1\}^*$, we define $\mathsf{enc}(x)$ to be the graph that is the disjoint union of paths P_i for all $i \leq |x|$ with $x_i = 1$, as well as of $|x|$ isolated vertices. Consider Figure 2 for a visualization.

Next, we show how to use the encoding enc to (reversibly) encode an instance of an arbitrary problem in $\#\mathsf{W}[1]$ into a pair of graphs. To that end, let (F,κ) denote any problem in $\#\mathsf{W}[1]$. By Lemma 3.1, we have that $(F,\kappa) \leq^{\mathrm{fpt}} \#\textsc{Hom}(\mathcal{K}_{even} \to \top)$, that is, there is an algorithm $\mathbb{A} = \mathbb{A}_F$ and a triple (f,g,s) of computable functions such that for all strings $x \in \{0,1\}^*$ all of the following holds:

(a) The algorithm $\mathbb{A}$ computes a pair of graphs $\mathbb{A}(x) = (H_x, G_x)$, where $H_x \in \mathcal{K}_{even}$ and the graph G_x is connected and H_x-colorable.

(b) The answer to the instance x of the problem (F,κ) can be computed as

$$F(x) = g(x) \cdot \#\mathsf{Hom}(H_x \to G_x).$$

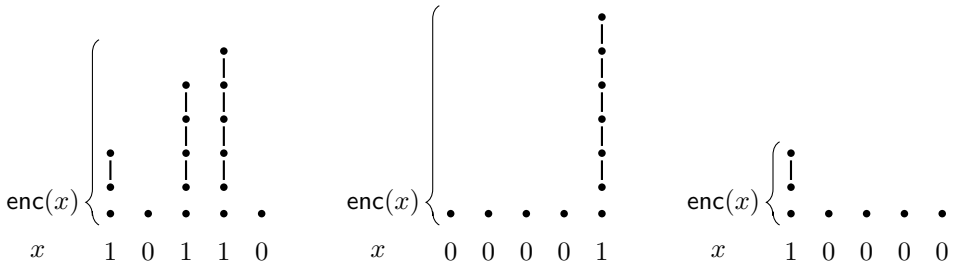

Figure 2 Examples of strings and their encoding into a graph.

(c) The problem (g, κ) is fixed-parameter tractable in time $f'(\kappa(x)) \cdot |x|^{O(1)}$.
(d) The algorithm $\mathbb{A}$ runs in time $f(\kappa(x)) \cdot |x|^{O(1)}$.
(e) The size of the computed graph $|V(H_x)|$ is at most $s(\kappa(x))$.

Now, let an instance x to (F, κ) be given. Using the graphs $\mathbb{A}(x) = (H_x, G_x)$ computed by the algorithm $\mathbb{A}$, we construct a pair of new graphs, which additionally encodes the original instance x as well as its parameter $\kappa(x)$ by setting

$$\hat{H}_x := H_x \cup \mathrm{K}(2\kappa(x) + 3), \text{ and} \tag{2}$$
$$\hat{G}_x := G_x \cup \mathsf{enc}(\langle x, H_x \rangle) \cup \mathrm{K}(2\kappa(x) + 3), \tag{3}$$

where $\langle x, H_x \rangle$ is any efficient encoding of the pair (x, H_x). We proceed to show that the constructed graphs $\hat{H}_x$ and $\hat{G}_x$ behave as intended; also consider Figure 3 for a visualization.

▶ **Lemma 3.7.** *Given a problem (F, κ) and an instance x to (F, κ), let the graphs $\hat{H}_x$ and $\hat{G}_x$ be defined as in Equations (2) and (3). Then we have that*
1. *The graph $\hat{G}_x$ is $\hat{H}_x$-colored and*
2. *The number of homomorphisms from $\hat{H}_x$ to $\hat{G}_x$ is given by*

$$\#\mathsf{Hom}(\hat{H}_x \to \hat{G}_x) = \#\mathsf{Hom}(H_x \to G_x) \cdot \#\mathsf{Aut}(\mathrm{K}(2\kappa(x) + 3)).$$

Proof. Recall that by definition, we have that $\hat{H}_x := H_x \cup \mathrm{K}(2\kappa(x) + 3)$ and $\hat{G}_x := G_x \cup \mathsf{enc}(\langle x, H_x \rangle) \cup \mathrm{K}(2\kappa(x) + 3)$, where $\mathsf{enc}(\star)$ encodes a string into a disjoint set of paths and isolated vertices.

Now, for the homomorphism from $\hat{G}_x$ to $\hat{H}_x$, note that the graph G_x is H_x-colored (by Lemma 3.1 and in particular Item a). Further, the graph $\mathrm{K}(2\kappa(x) + 3)$ has an automorphism and contains at least one edge, so there is a homomorphism from the graph $\mathsf{enc}(\langle x, H_x \rangle) \cup \mathrm{K}(2\kappa(x) + 3)$ into the graph $\mathrm{K}(2\kappa(x) + 3)$. In total, this completes the proof that $\hat{G}_x$ is $\hat{H}_x$-colored.

For the number of homomorphisms from $\hat{H}_x$ to $\hat{G}_x$, note that there are $\#\mathsf{Hom}(H_x \to G_x)$ many homomorphisms from H_x to G_x and $\#\mathsf{Aut}(\mathrm{K}(2\kappa(x) + 3))$ many homomorphisms from $\mathrm{K}(2\kappa(x) + 3)$ to itself. As the graphs $\hat{H}_x$ and $\hat{G}_x$ consist of the disjoint union of H_x and $\mathrm{K}(2\kappa(x) + 3)$, and G_x and $\mathrm{K}(2\kappa(x) + 3)$, respectively, we directly obtain a lower bound for the number of homomorphisms:

$$\#\mathsf{Hom}(\hat{H}_x \to \hat{G}_x) \geq \#\mathsf{Hom}(H_x \to G_x) \cdot \#\mathsf{Aut}(\mathrm{K}(2\kappa(x) + 3)). \tag{4}$$

To prove the upper bound, observe the following.

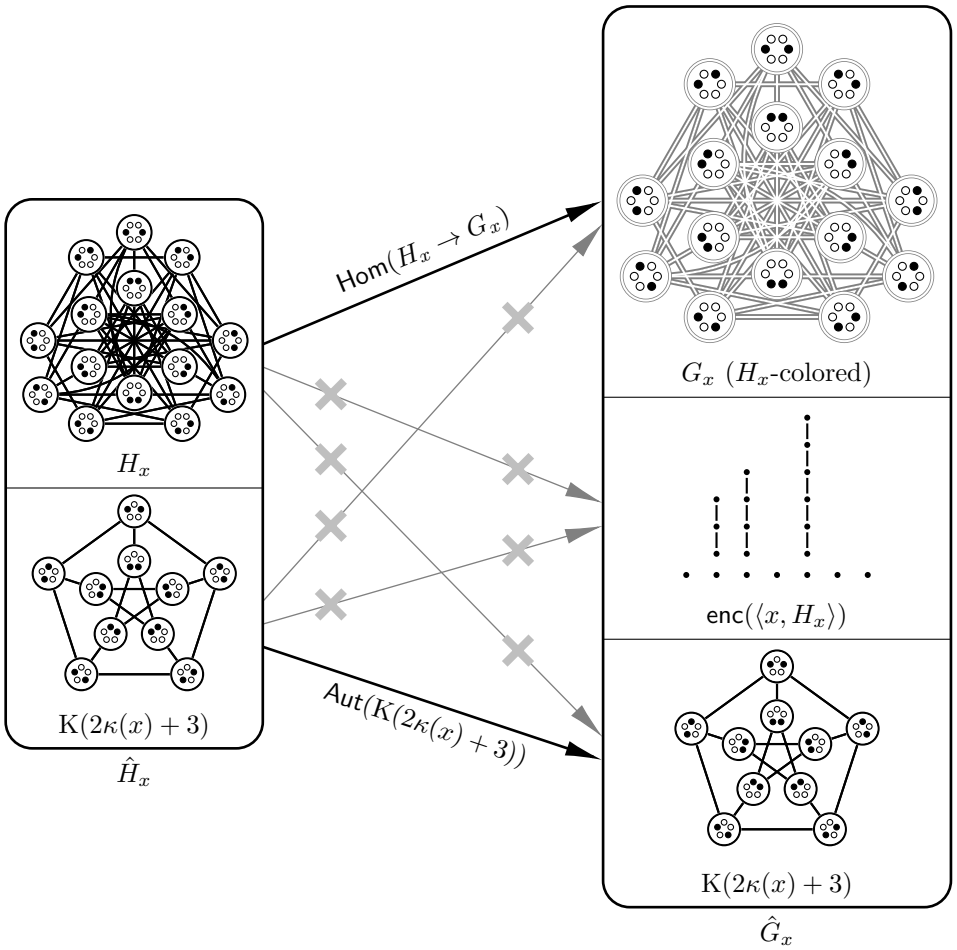

Figure 3 Lemma 3.7 illustrated. A cross denotes that no homomorphisms between the parts of the graphs exist. Note that the Kneser graphs used in the lemma differ from the ones depicted.

(i) The graph H_x cannot be mapped homomorphically to the graph $\mathrm{K}(2\kappa(x)+3)$, as H_x is contained in $\mathcal{K}_{even}$ and $\mathrm{K}(2\kappa(x)+3)$ is contained in $\mathcal{K}_{odd}$; hence both are distinct Kneser graphs and by Lemma 3.6 no homomorphisms between them are possible.

(ii) The graph $\mathrm{K}(2\kappa(x)+3)$ cannot be mapped homomorphically to the graph G_x. Suppose otherwise, that an homomorphism $h : \mathrm{K}(2\kappa(x)+3) \to G_x$ existed. As G_x is H_x-colorable, there is a homomorphism $c : G_x \to H_x$. Composing the homomorphisms h and c yields a homomorphism

$$h \circ c : \mathrm{K}(2\kappa(x)+3) \to H_x,$$

that is, a homomorphism from a graph in $\mathcal{K}_{odd}$ to a graph in $\mathcal{K}_{even}$, which, again, is not possible by Lemma 3.6.

(iii) None of the graphs H_x and $\mathrm{K}(2\kappa(x)+3)$ can be mapped homomorphically to the graph $\mathsf{enc}(\langle x, H_x \rangle)$, as paths have a chromatic number of at most 2, and both graphs H_x and $\mathrm{K}(2\kappa(x)+3)$ have a chromatic number of at least 3.

Hence, the homomorphisms counted in the lower bound (4) are already *all* homomorphisms from $\hat{H}_x$ to $\hat{G}_x$:

$$\#\mathsf{Hom}(\hat{H}_x \to \hat{G}_x) = \#\mathsf{Hom}(H_x \to G_x) \cdot \#\mathsf{Aut}(\mathrm{K}(2\kappa(x)+3)).$$

This completes the proof. ◀

Now let $\hat{\mathcal{H}}$ and $\hat{\mathcal{G}}$ be the sets of all graphs $\hat{H}_x$ and $\hat{G}_x$, respectively, corresponding to instances x to (F, κ) for which the function g is non-zero, that is $g(x) \neq 0$. For the classes $\hat{\mathcal{H}}$ and $\hat{\mathcal{G}}$ to be useful to us, we need to show that the only pairs of graphs $H \in \hat{\mathcal{H}}$ and $G \in \hat{\mathcal{G}}$ that admit a homomorphism from H to G are those, that correspond to the same pair $(x, \kappa(x))$. Formally, consider the following lemma.

▶ **Lemma 3.8.** *Let a problem $(F, \kappa) \in \#\mathsf{W}[1]$ and the corresponding graph classes $\hat{\mathcal{H}}$ and $\hat{\mathcal{G}}$ be given. For any graphs $K \in \hat{\mathcal{H}}$ and $G \in \hat{\mathcal{G}}$, if there is a homomorphism from H to G, then there is an instance x to (F, κ) that corresponds to both H and G, that is, $H = \hat{H}_x$ and $G = \hat{G}_x$.*

Proof. It suffices to show that there are no homomorphisms from H to G if the graphs H and G *do not* correspond to the same instance of (F, κ). Hence, assume that the graphs $K \in \hat{\mathcal{H}}$ and $G \in \hat{\mathcal{G}}$ correspond to distinct instances $x = x_K$ and $y = x_G$ of (F, κ). For the sake of contradiction, further assume that there is a homomorphism h from the graph H to the graph G. By (2) and (3), for some distinct integers a, b, an integer c, and a graph $H_x \neq K(a)$, we have that

$$H = \mathrm{K}(a) \cup \mathrm{K}(b) \text{ and}$$
$$G = G_1 \cup \mathsf{enc}(\langle y, H_y \rangle) \cup \mathrm{K}(c),$$

where G_1 is a connected graph that is H_y-colored.

Similar to (iii) from the proof of Lemma 3.7 we can show that, there are no homomorphisms from the graphs $\mathrm{K}(a)$ or $\mathrm{K}(b)$ to the graph $\mathsf{enc}(\langle x, H_x \rangle)$. Further, as the numbers a and b are distinct, only at most one of a and b may be equal to c. We distinguish two cases, depending on whether c is equal to either a or b, or not.

First, assume that the number b is the same as c. (The case $a = c$ is similar.) In this case, we have that $\mathrm{K}(b) = \mathrm{K}(c)$, and hence $\kappa(x) = \kappa(y)$. Further, by Lemma 3.6, the homomorphism h maps the graph $\mathrm{K}(a)$ to the graph G_1, as $\mathrm{K}(a)$ and $\mathrm{K}(c)$ are different Kneser graphs. Combining this homomorphism from $\mathrm{K}(a)$ to G_1 with the homomorphism from G_1 to H_y (which exists as G_1 is H_y-colorable) yields a homomorphism from $\mathrm{K}(a)$ to H_y. However, as $\mathrm{K}(a) = H_x$ and H_y are both Kneser graphs, a homomorphism between them is only possible if they are the same Kneser graph. This in turn, means that the instances x and y are the same, which is a contradiction.

Second, consider the case where the numbers a, b, and c are pairwise distinct. By Lemma 3.6, we obtain that there are no homomorphisms from the graph $\mathrm{K}(a)$ to the graph $\mathrm{K}(c)$, as well as that there are no homomorphisms from the graph $\mathrm{K}(b)$ to the graph $\mathrm{K}(c)$. Hence, the homomorphism h maps both graphs $\mathrm{K}(a)$ and $\mathrm{K}(b)$ to the graph G_1. Assume wlog. that $\mathrm{K}(a) = H_x$. Now, as before, we obtain a homomorphism from the graph H_x to the graph H_y and hence (by Lemma 3.6) $x = y$, which is a contradiction.

In total, if H and G do not correspond to the same instance x, there is no homomorphism from H to G. This concludes the proof. ◀

The Main Reductions

Using Lemmas 3.7 and 3.8, we proceed to show that the problems (F, κ) and $\#\textsc{Hom}(\hat{\mathcal{H}} \to \hat{\mathcal{G}})$ are interreducible with respect to parameterized Turing reductions.

Proof of Theorem 3.2. We start with the more involved direction.

▷ Claim 3.9. We have that $\#\textsc{Hom}(\hat{\mathcal{H}} \to \hat{\mathcal{G}}) \leq_{\mathrm{T}}^{\mathrm{fpt}} (F, \kappa)$.

Proof. Given graphs H and G and an oracle $\mathbb{O}$ for the problem (F, κ), we wish to compute the number of homomorphisms $\#\mathsf{Hom}(H \to G)$ if the promise $H \in \hat{\mathcal{H}}$ and $G \in \hat{\mathcal{G}}$ is fulfilled. Consider the following algorithm $\mathbb{B}$.

1. Verify that the graph H is the union of two Kneser graphs $\mathrm{K}(a) \in \mathcal{K}_{even}$ and $\mathrm{K}(b) \in \mathcal{K}_{odd}$. If this is not the case, output 0.
2. Verify that the graph G is the union of a set of paths $\mathcal{P}$ (and isolated vertices) and two connected components G_1 and G_2 that are not paths. If this is not the case, output 0.
3. Verify that either $G_1 = \mathrm{K}(b)$ or $G_2 = \mathrm{K}(b)$ holds. If this holds, assume w.l.o.g. that $G_2 = \mathrm{K}(b)$. Otherwise output 0.
4. Find a pair (c, H') such that $\mathsf{enc}(\langle c, H' \rangle) = \mathcal{P}$ or report that no decoding is possible (e.g. if the set of paths $\mathcal{P}$ is empty, contains the same path multiple times or the number of isolated vertices does not match). If the decoding failed, output 0.
5. Compute the parameter $\kappa(c)$ of the instance c. If we have that $2\kappa(c) + 3 \neq b$, output 0.
6. Verify that the graphs H' and $\mathrm{K}(a)$ are isomorphic. If they are not isomorphic, output 0.
7. Compute the value $g(c)$. If we have that $g(c) = 0$, output 0.
8. Query the oracle $\mathbb{O}$ on input c and obtain $\mathbb{O}(c)$. Output the number

$$\mathbb{O}(c) \cdot \#\mathsf{Aut}(\mathrm{K}(2\kappa(c) + 3)) \cdot g(c)^{-1}.$$

We first prove the required bound on the running time of the algorithm $\mathbb{B}$. On input H and G, Step 1 takes time only depending on $|V(H)|$; Step 2 can be done in time polynomial in $|V(G)|$. Step 3 takes again time only depending on $|V(H)|$. Considering Step 4, we observe that by the definition of the encoding enc and by the assumption that $\langle \star, \star \rangle$ is an efficient encoding of pairs, the decoding can be done in time polynomial in $|V(\mathcal{P})| \leq |V(G)|$. Step 5 can also be done in time polynomial in $|V(G)|$, as the function κ is computable in polynomial time in $|c|$. As the encoding $\mathsf{enc}(\langle c, H \rangle)$ contains an isolated vertex for every bit of the string c, we have that

$$|c| \leq |V(G)| \tag{5}$$

and hence the claimed running time for Step 5. Similarly to the Step 3, we can perform Step 6 in time only depending on $|V(H)|$. Now assume Step 7 is reached. In this case, we have that $2\kappa(c) + 3 = b$ and consequently

$$\kappa(c) \leq |V(H)|, \tag{6}$$

as the graph $\mathrm{K}(b)$ is a component of H and we have that $|V(\mathrm{K}(b))| \geq b$. Note that Steps 7 and 8 take time $f'(\kappa(c)) \cdot |c|^{O(1)}$, as the problem (g, κ) is fixed-parameter tractable (see (c)). W.l.o.g., we can assume that f' is monotonically increasing and thus (6) yields a running time bound of $f'(|V(H)|) \cdot |V(G)|^{O(1)}$; recall (5), that is $|c| \leq |V(G)|$.
Note that the last argument also shows that the parameter $\kappa(c)$ of the oracle query $\mathbb{O}(c)$ is bounded by $|V(H)|$.

It remains to prove the correctness of algorithm $\mathbb{B}$. To this end, assume that the promise is fulfilled, that is, $H \in \hat{\mathcal{H}}$ and $G \in \hat{\mathcal{G}}$. (If the promise is not fulfilled, we are not required to compute a correct output.)

Hence, for instances x and y, we have that

$$\begin{aligned} H = \hat{H}_y &= \mathrm{K}(a) \cup \mathrm{K}(b) \\ &= H_y \cup \mathrm{K}(2\kappa(y) + 3) \end{aligned}$$

and

$$\begin{aligned} G = \hat{G}_x &= G_1 \cup \mathcal{P} \cup G_2 \\ &= G_x \cup \mathsf{enc}(\langle x, H_x \rangle) \cup \mathrm{K}(2\kappa(x) + 3). \end{aligned}$$

Further, by construction we have that $g(x) \neq 0$. We consider three cases.

(i) $H_x \neq H_y$: The instances x and y are different. Hence by Lemma 3.8, there are no homomorphisms from H to G.
In the algorithm $\mathbb{B}$, in this case, the test in Step 6 fails, and $\mathbb{B}$ outputs 0, which is correct.

(ii) $H_x = H_y$ and $\kappa(x) \neq \kappa(y)$. Note that $H_x = H_y$ does not imply that the corresponding instances are the same; the algorithm $\mathbb{A}$ is not necessarily injective. Indeed, in this case the instances x and y differ and so, again by Lemma 3.8, there are no homomorphisms from the graph H to the graph G.
In the algorithm $\mathbb{B}$, in this case, the test in Step 3 fails, and $\mathbb{B}$ outputs 0, which is correct.

(iii) $H_x = K_y$ and $\kappa(x) = \kappa(y)$: In this case, we have that $\hat{H}_y = \hat{H}_x$. Hence, by Lemma 3.7, the number of homomorphisms from H to G is

$$\#\mathsf{Hom}(H \to G) = \#\mathsf{Hom}(H_x \to G_x) \cdot \#\mathsf{Aut}(\mathrm{K}(2\kappa(x) + 3)). \tag{7}$$

Note that the oracle $\mathbb{O}$ on input x computes the number

$$\mathbb{O}(x) = g(x) \cdot \#\mathsf{Hom}(H_x \to G_x), \tag{8}$$

and we may assume that $g(x) \neq 0$ by construction. Hence, combining (7) and (8) yields that we can compute the number of homomorphisms from the graph H to the graph G as follows:

$$\#\mathsf{Hom}(H \to G) = \mathbb{O}(x) \cdot g(x)^{-1} \cdot \#\mathsf{Aut}(\mathrm{K}(2\kappa(x) + 3)).$$

In the algorithm $\mathbb{B}$, it is easy to verify that the Steps 3 to 7 succeed and that in Step 8, we indeed return $\mathbb{O}(x) \cdot g(x)^{-1} \cdot \#\mathsf{Aut}(\mathrm{K}(2\kappa(x) + 3))$. Hence, the algorithm is correct in this case as well.

In total, the algorithm $\mathbb{B}$ correctly solves the problem $\#\textsc{Hom}(\hat{H} \to \hat{G})$. This finishes the proof of the reduction. ◁

Finally, we construct and verify the easy reduction.

▷ Claim 3.10. We have that $(F, \kappa) \leq_{\mathrm{T}}^{\mathrm{fpt}} \#\textsc{Hom}(\hat{\mathcal{H}} \to \hat{\mathcal{G}})$.

Proof. Given an instance x to (F, κ) and an oracle $\mathbb{O}$ solving the problem $\#\textsc{Hom}(\hat{\mathcal{H}} \to \hat{\mathcal{G}})$, we wish to compute the number $F(x)$. Recall that by Lemma 3.1, there is a reduction $(F, \kappa) \leq_{\mathrm{T}}^{\mathrm{fpt}} \#\textsc{Hom}(\mathcal{K}_{even} \to \top)$; let $\mathbb{A}$ again denote the corresponding algorithm. Recall further, that for the graphs $(H_x, G_x) = \mathbb{A}(x)$, we have that

$$F(x) = g(x) \cdot \#\mathsf{Hom}(H_x \to G_x).$$

Now, to compute the result $F(x)$, we first compute the value $g(x)$ in FPT time with respect to κ. If we observe $g(x) = 0$, we output 0. Otherwise, we simulate the algorithm $\mathbb{A}$ and obtain graphs H_x and G_x in time $f(\kappa(x)) \cdot |x|^{O(1)}$. After that, we can compute the graphs $\hat{H}_x$ and $\hat{G}_x$ in time $\hat{f}(\kappa(x)) \cdot |x|^{O(1)}$: The construction of the encoding $\mathsf{enc}(\langle x, H_x \rangle)$ can

be done in polynomial time in $|x|$ and $|V(H_x)|$. Note that $|V(H_x)|$ is bounded by $s(\kappa(x))$ and that the construction of $\mathrm{K}(2\kappa(x)+3)$ clearly takes time only depending on $\kappa(x)$. In particular, we have that the size of the graph $\hat{H}_x$ only depends on $\kappa(x)$. Hence, we can query the oracle $\mathbb{O}$ for the problem $\#\textsc{Hom}(\hat{\mathcal{H}} \to \hat{\mathcal{G}})$ on the graphs $\hat{H}_x$ and $\hat{G}_x$ and obtain the number of homomorphisms from $\hat{H}_x$ and $\hat{G}_x$.

Recall that by Lemma 3.7, we have that

$$\begin{aligned}\#\mathsf{Hom}(H_x \to G_x) &= \#\mathsf{Aut}(\mathrm{K}(2\kappa(x)+3))^{-1} \cdot \#\mathsf{Hom}(\hat{H}_x \to \hat{G}_x) \\ &= \#\mathsf{Aut}(\mathrm{K}(2\kappa(x)+3))^{-1} \cdot \mathbb{O}(\hat{H}_x, \hat{G}_x).\end{aligned}$$

Hence, to compute the result $F(x) = g(x) \cdot \#\mathsf{Hom}(H_x \to G_x)$, we can compute the number $\#\mathsf{Aut}(\mathrm{K}(2\kappa(x)+3))^{-1}$ in time only depending on $\kappa(x)$ and multiply with the result of the oracle query and the result previous computation of the value $g(x)$. This completes the proof, as we may assume that the algorithm $\mathbb{A}$ is correct. ◁

In total, by the reductions from Claims 3.9 and 3.10, we obtain

$$(F, \kappa) \equiv^{\mathrm{fpt}}_{\mathrm{T}} \#\textsc{Hom}(\mathcal{H} \to \mathcal{G}),$$

thus completing the proof. ◀

Note that the previous proof shows the corresponding theorem for the decision realm, if we choose the decision version of Lemma 3.1 as a starting point; the decision version of Lemma 3.1 can be found as Corollary A.4 in Appendix A.

▶ **Theorem 3.11.** *Let (F, κ) denote a problem in $\mathsf{W}[1]$. There are classes $\mathcal{H}$ and $\mathcal{G}$ such that*

$$(F, \kappa) \equiv^{\mathrm{fpt}}_{\mathrm{T}} \textsc{Hom}(\mathcal{H} \to \mathcal{G}).$$

Furthermore, $\mathcal{H}$ is recursively enumerable and $\mathcal{G}$ is recursive.

4 Counting Homomorphisms and Subgraphs in F-Colorable Graphs

Let $\mathcal{H}$ denote a recursively enumerable class of graphs. Further, given a fixed graph F, let $\mathcal{G}_F$ denote the class of all graphs G that admit a homomorphism to F, that is, the class of F-colorable graphs. In this section we establish that the existing dichotomy for counting homomorphisms due to Dalmau and Jonsson [19] extends to the PPC problem $\#\textsc{Hom}(\mathcal{H} \to \mathcal{G}_F)$; that is, counting the number of homomorphisms from a graph $H \in \mathcal{H}$ to a graph $G \in \mathcal{G}_F$. Note that the notion of $\mathcal{G}_F$ captures and generalizes the important special cases of the class of all bipartite graphs (when F is a single edge) or, more general, the class of all k-colorable graphs for any fixed number k (when F is the complete graph on k vertices).

▶ **Theorem 4.1.** *Let F be a graph, and let $\mathcal{H}$ be a recursively enumerable class of graphs.*

(1) *If the treewidth of $\mathcal{H} \cap \mathcal{G}_F$ is bounded then the PPC problem $\#\textsc{Hom}(\mathcal{H} \to \mathcal{G}_F)$ is polynomial-time solvable.*

(2) *Otherwise, the problem $\#\textsc{Hom}(\mathcal{H} \to \mathcal{G}_F)$ is $\#\mathsf{W}[1]$-hard.*

It turns out that the previous theorem can be proved by a refined analysis of the existing proof due to Dalmau and Jonsson [19]. For this reason, we defer the proof to Appendix B.1. In what follows, instead, we demonstrate that the previous classification for counting homomorphisms to F-colored graphs yields a complete classification for the associated subgraph counting

problem. More precisely, we define $\#\textsc{Sub}(\mathcal{H} \to \mathcal{G})$ as the PPC problem of, given graphs $H \in \mathcal{H}$ and $G \in \mathcal{G}$, computing the number $\#\mathsf{Sub}(H \to G)$, that is, the number of subgraphs in G that are isomorphic to H; the parameter is $|V(H)|$. Formally, the promise is the set $\mathcal{H} \times \mathcal{G}$.

An example of a problem $\#\textsc{Sub}(\mathcal{H} \to \mathcal{G})$ is the problem of computing the number of k-matchings in bipartite graphs; recall that a k-matching is a set of k edges that are pairwise disjoint. This problem was first shown to be $\#\mathsf{W}[1]$-hard by Curticapean and Marx [17] and constitutes the bottleneck for the intractable cases of the subgraph counting problem:

▶ **Theorem 4.2** ([17]). *Let $\mathcal{H}$ be a recursively enumerable class of graphs.*

(1) *If the matching number of the class $\mathcal{H}$ is bounded then the problem $\#\textsc{Sub}(\mathcal{H} \to \top)$ is polynomial-time solvable.*

(2) *Otherwise, the problem $\#\textsc{Sub}(\mathcal{H} \to \top)$ is $\#\mathsf{W}[1]$-hard.*

Here, the *matching number* of a graph is the size of its largest matching and a class of graphs $\mathcal{H}$ has bounded matching number if there exists an overall constant c such that the matching number of each graph $H \in \mathcal{H}$ is bounded by c.

Recently, Curticapean, Dell, and Marx [16] strongly generalized Theorem 4.2 with a much simpler proof. They key ingredient of their work is the algorithm given by Lemma 2.17. We further generalize their proof to F-colorable graphs and obtain the following strengthening of the classification for counting subgraphs.

▶ **Theorem 4.3.** *Let F be a fixed graph and let $\mathcal{H}$ be a recursively enumerable class of graphs.*

(1) *If the matching number of $\mathcal{H} \cap \mathcal{G}_F$ is bounded then the problem $\#\textsc{Sub}(\mathcal{H} \to \mathcal{G}_F)$ is polynomial-time solvable.*

(2) *Otherwise, the problem $\#\textsc{Sub}(\mathcal{H} \to \mathcal{G}_F)$ is $\#\mathsf{W}[1]$-hard.*

Due to space constraints and the fact that we only need to perform minor modifications of the arguments of Curticapean, Dell and Marx [16], we defer the proof to Appendix B.2.

5 Counting Homomorphisms in König Graphs

Given a graph G, its associated *line graph* $L(G)$ is the following graph: As vertices $L(G)$ has the edges of G and two vertices e and $\hat{e}$ of $L(G)$ are adjacent if the corresponding edges are neither equal nor disjoint, that is, $|e \cap \hat{e}| = 1$. We write $\mathcal{L}$ for the set of all line graphs. By König's Theorem, a line graph of a bipartite graph is also a perfect graph (see e.g. [13]). To simplify notation, we hence call a line graph of a bipartite graph a *König graph*. We write ♚ to denote the class of all König graphs[6]. This section is devoted to the complexity analysis of the problem $\#\textsc{Hom}(\mathcal{H} \to ♚)$ of counting homomorphisms from a graph from some arbitrary graph class $\mathcal{H}$ to a König graph.

5.1 Tractability of Deciding Homomorphisms in König Graphs

We start by investigating the decision version, that is, the problem $\textsc{Hom}(\top \to ♚)$. It turns out that if we are only interested in the existence, and not the number, of homomorphisms, then the problem becomes fixed-parameter tractable.

[6] The symbol ♚ is used since "König" is the German word for "King".

▶ **Theorem 5.1.** *The decision problems* $\textsc{Hom}(\top \to \mathcal{L})$ *and thus* $\textsc{Hom}(\top \to \text{♚})$ *are fixed-parameter tractable. In particular, given a graph* H *and a line graph* L*, it is possible to decide the existence of a homomorphism from* H *to* L *in time*

$$f(|V(H)|) \cdot O(|V(L)|^2),$$

for some computable function f *independent of* H *and* L*.*

Proof. We construct an algorithm $\mathbb{A}$ for the problem $\mathsf{Hom}(\top \to \mathcal{L})$ that, given a graphs $H \in \top$ and $L \in \mathcal{L}$, correctly decides whether there exists a homomorphism from H to L. Further, the algorithm $\mathbb{A}$ runs in time $f(|V(H)|) \cdot O(|V(L)|^2)$ for some computable function f independent of H and L. The algorithm $\mathbb{A}$ relies on the clique partition of line graphs [37, Chapter 8], stating that $E(L)$ can be partitioned into cliques such that every vertex of L is contained in at most 2 cliques. Here, every clique corresponds to a vertex of a primal graph of G such that $L(G) = L$. In particular, it is easy to see that the size of the largest clique in the partition is precisely the maximum degree of G. Consequently, our algorithm first computes a primal graph G of L, which can be done in time $O(|V(L)|^2)$ [41]. Next, we compute the maximum degree d of G, which can be done in time $O(|V(G)|) = O(|V(L)|^2)$.

Now let $k = |V(H)|$ and let $H_1, \ldots, H_\ell$ be the connected components of H. For every connected component H_i, we proceed as follows. If $d \geq k$ then there exists a homomorphism from H_i to L, as we can embed H_i into a clique of size d. Otherwise, the properties of the clique partition yield that the degree of L is bounded by $2k$: Every vertex of L is contained in at most two cliques and every clique is of size at most $d < k$. Consequently, we can perform a standard bounded search-tree algorithm: We guess the image $v \in V(L)$ of a vertex $h \in V(H_i)$. As the graph H_i is connected and $|V(H_i)| \leq k$, every homomorphism from H_i to L that maps h to v must also map every further vertex of H_i to a vertex in the k-neighborhood of v. As the maximum degree of L is at most $2k$, the size of the graph induced by the k-neighborhood of v is bounded by $(2k)^k$. We can then search for a homomorphism by brute-force; this takes time only depending on k. The final output is 1 if a homomorphism is found from every connected component H_i and 0 otherwise.

The total running time is bounded by

$$O(|V(L)|^2) + f(|V(H)|) \cdot O(|V(L)|) \leq f(|V(H)|) \cdot O(|V(L)|^2);$$

this completes the proof. ◀

5.2 An Explicit Criterion for Hardness of Counting Homomorphisms in König Graphs

Theorem 5.1 in turn further motivates the study of the counting version: The most interesting hardness results in counting complexity theory are concerned with problems that admit a tractable decision version [57]. In particular, we construct an explicit reduction from #Clique to prove the following hardness result.

▶ **Lemma 5.2.** *Let* $\mathcal{H}$ *be a recursively enumerable class of graphs. If* $\mathcal{H}$ *has unbounded treewidth and is closed under taking minors, then the problem* $\#\textsc{Hom}(\mathcal{H} \to \text{♚})$ *is* $\#\mathsf{W}[1]$*-hard.*

Note that König graphs are a subset of the perfect graphs [13], as well as a subset of the line graphs (of arbitrary graphs). As line graphs are also claw-free graphs [3], König graphs are also a subset of the claw-free graphs. Hence, the hardness result for the problem $\#\textsc{Hom}(\mathcal{H} \to \text{♚})$ extends to perfect graphs, line graphs, and claw-free graphs as well.

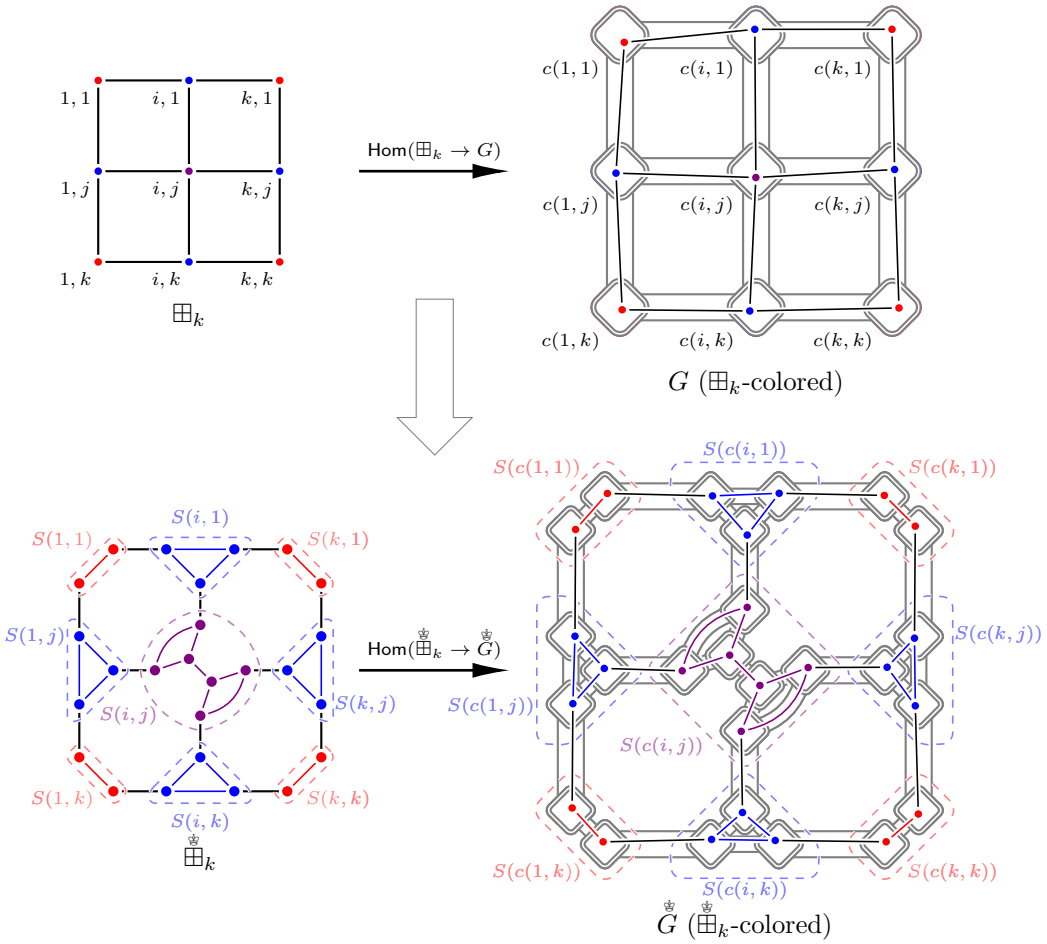

Figure 4 General overview of the reduction from Lemma 5.2: Corner vertices (red), border vertices (blue), and interior vertices (purple) get replaced by corresponding gadgets; the resulting graph $\overset{\scriptscriptstyle\bigstar}{G}$ is a König graph. We use $c(\star)$ to denote the set of all vertices colored with $\star$ and $S(\star)$ to denote the gadget corresponding to $\star$; the numbers i and j denote intermediate columns and rows.

To prove Lemma 5.2, we use a gadget construction that transforms an arbitrary graph G into a König graph such that the number of grid-like subgraphs remains stable. In view of the diverse applications of the Grid-Tiling Problem (see e.g. [18, Chapter 14.4.1]), the construction might yield further intractability results for counting problems on König graphs (and hence on claw-free and perfect graphs).

Proof. We write $\boxplus_k$ for the $k \times k$ square grid, that is, the graph with the vertices

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

and two vertices (i,j) and (i',j') are adjacent if $|i-i'|+|j-j'|=1$. Now let $\boxplus$ be the set of all square grids $\boxplus_k$ for $k \in \mathbb{N}$. We prove a reduction from the problem #CP-Hom($\boxplus \to \top$), which is known to be #W[1]-hard and constitutes an important intermediate step in the proof of the classification of the homomorphism counting problem due to Dalmau and Jonsson [19]. A sketch of the #W[1]-hardness proof can be found in Appendix A and the full proof can be found e.g. in [15, Lemma 5.7].

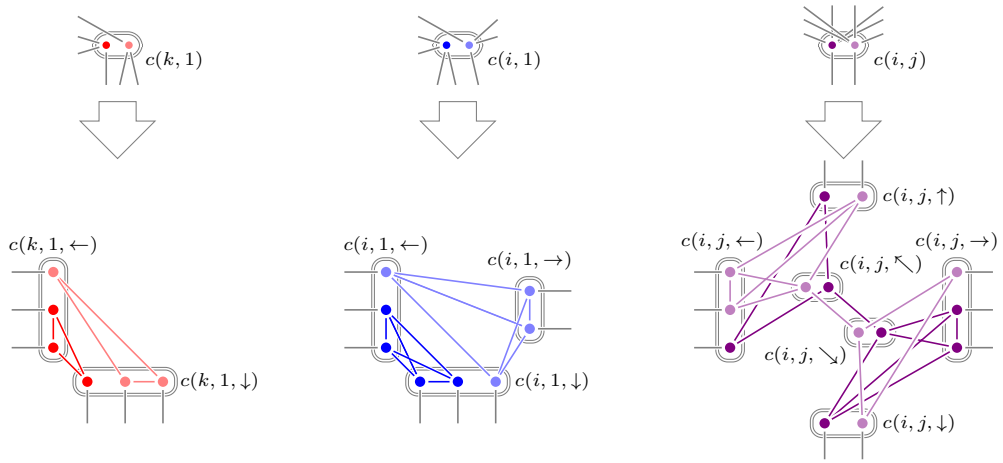

Figure 5 The gadgets in detail.

Let us recall the definition of the problem #CP-HOM($\boxplus \to \top$). This problem expects as input a pair of a square grid $\boxplus_k$ and a graph G that is $\boxplus_k$-colored by some given coloring c. The task is to compute the number of color-prescribed homomorphisms from $\boxplus_k$ to G, that is, homomorphisms $h \in \mathsf{Hom}(\boxplus_k \to G)$ that additionally satisfy $c(h(v)) = v$ for every vertex v of the grid.

For the first part of the reduction, we present a construction that maps a $\boxplus_k$-colored graph G to a vertex-colored König graph $\mathring{G}$; consider Figure 4 for an overview of the construction. Let c be the coloring of G. We partition the vertices of G into three disjoint sets (again, consider Figure 4):

(1) A vertex v is called a *corner vertex* if its coloring $c(v)$ is one of the values $(1,1)$, $(1,k)$, $(k,1)$, and (k,k).

(2) A vertex v is called a *border vertex* if its coloring $c(v)$ satisfies

$$c(v) \in \{(i,j) \in [k]^2 \mid i \in \{1,k\} \vee j \in \{1,k\}\} \setminus \{(1,1),(1,k),(k,1),(k,k)\}.$$

(3) All remaining vertices are called *interior vertices*.

We construct a gadget (graph) $S(v)$ for each vertex $v \in V(G)$. Here, the graph $S(v)$ depends on whether v is a corner, a border or an interior vertex.

(1) The vertex v is a corner vertex. Assume that $c(v) = (1,1)$; the other cases are symmetric. Now let $N_\rightarrow$ be the set of neighbors of v that are colored by c with $(1,2)$ and let $N_\downarrow$ be the set of all neighbors of v that are colored by c with $(2,1)$. Note that that those are all neighbors of v as G is $\boxplus_k$-colored. For each vertex $u \in N_\rightarrow$, we add a vertex $v^u_\rightarrow$ and color it with $(1,1,\rightarrow)$. For each vertex $u \in N_\downarrow$, we add a vertex $v^u_\downarrow$ and color it with $(1,1,\downarrow)$. The graph $S(v)$ is then obtained by making all of the previous vertices adjacent to each other.

(2) The vertex v is a border vertex. Assume that $c(v) = (1,j)$; the other cases are symmetric. Now let $N_\rightarrow$ be the set of neighbors of v that are colored by c with $(1,j+1)$, let $N_\leftarrow$ be the set of neighbors of v that are colored by c with $(1,j-1)$, and let $N_\downarrow$ be the set of all neighbors of v that are colored by c with $(2,j)$. For each vertex $u \in N_\rightarrow$, we add a vertex $v^u_\rightarrow$ and color it with $(1,i,\rightarrow)$. We proceed similarly with $N_\leftarrow$ and $N_\downarrow$. The graph $S(v)$ is then obtained by making all of the previous vertices adjacent.

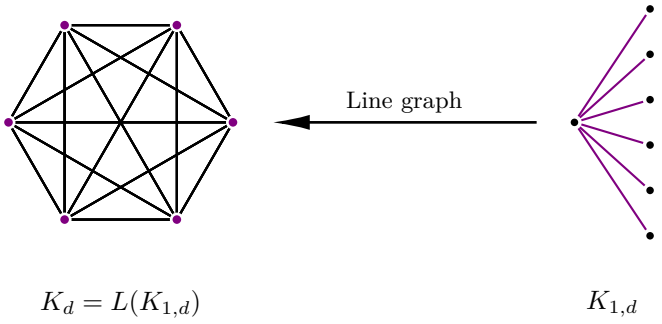

Figure 6 A clique K_d of size d is the line graph of a star $K_{1,d}$ with d rays.

(3) The vertex v is an interior vertex. Let v have color $c(v) = (i, j)$ and let $N_\rightarrow$, $N_\leftarrow$, $N_\uparrow$ and $N_\downarrow$ be the sets of neighbors of v that are colored by c with $(i, j+1)$, $(i, j-1)$, $(i-1, j)$ and $(i+1, j)$, respectively. We add a new vertex $v^u_\rightarrow$ for each vertex $u \in N_\rightarrow$ and color it with $(i, j, \rightarrow)$; we proceed similarly with the sets $N_\uparrow, N_\leftarrow$, and $N_\downarrow$. Next, we add two new vertices $v^{\nwarrow}$ and $v^{\searrow}$, color them $(i, j, \nwarrow)$ and $(i, j, \searrow)$, and connect them by an edge. Then we create two cliques: The first clique contains the vertex $v^\star$ and all vertices that we colored with $(i, j, \leftarrow)$ or with $(i, j, \uparrow)$. The second clique contains the vertex $v^{\searrow}$ and all vertices that we colored with $(i, j, \rightarrow)$ or with $(i, j, \downarrow)$. The resulting graph is $S(v)$.

The graph $\mathring{G}$ is obtained by connecting the gadgets as follows: Let $\{v, w\} \in E(G)$ denote an edge of G and assume that the vertex v has color $c(v) = (i, j)$ and the vertex w has color $c(w) = (i, j+1)$; the remaining cases are processed similarly. By construction, the graph $S(v)$ contains a vertex $v^w_\rightarrow$ and the graph $S(w)$ contains a vertex $w^v_\leftarrow$. We connect those two vertices with an edge.

We first observe that this construction yields a planar graph if it is applied to the grid itself (Again, consider Figure 4). Further, when applied to the graph G, we indeed obtain a König graph:

▷ Claim 5.3. If the graph G is $\boxplus_k$-colored, then the graph $\mathring{G}$ is a König graph.

Proof. We construct a bipartite graph B such that the line graph $L(B)$ of B is the graph $\mathring{G}$. To this end, we observe that the gadgets of corner and border vertices are cliques, and the gadgets of interior vertices are two cliques that are connected by a single edge. Hence, the entire graph $\mathring{G}$ is obtained by connecting vertex disjoint cliques with edges that are pairwise disjoint.

Now observe that cliques are the line graphs of stars. More precisely, let $K_{1,d}$ be the complete bipartite graph with 1 vertex on the left side and d vertices on the right side, then its line graph $L(K_{1,d})$ is the clique of size d; consider Figure 6 for a visualization. In what follows, we say that the single vertex on the left side of $K_{1,d}$ is the *center* and the d vertices on the right side are the *rays*.

Now, adding an edge between two vertex disjoint cliques corresponds to merging the right vertices of the corresponding rays of the primal graphs (Consider Figure 7 for a visualization). Consequently, we can construct a graph B whose line graph is $\mathring{G}$ by merging right vertices of the rays corresponding to the edges that connect the cliques of the gadgets.

Finally, it is easy to see that B is bipartite: A 2-coloring is given by the function that maps the centers to 1 and the (identifications of) rays to 2. ◁

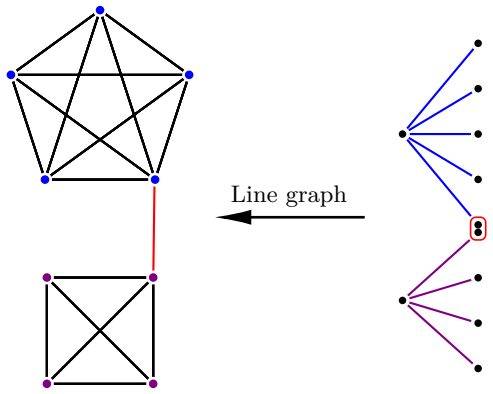

Figure 7 Connecting two vertex disjoint cliques corresponds to merging vertices in the corresponding primal graphs. Note that the resulting primal graph stays bipartite.

Now, recall that we colored the vertices of $\tilde{G}$ with triples $(i, j, \star)$, where $\star$ is one of the symbols $\rightarrow, \leftarrow, \uparrow, \downarrow, \nwarrow$, and $\searrow$. Let $\hat{c}$ be this coloring of the graph $\tilde{G}$. Observe that the coloring $\hat{c}$ is a bijection if $G = \boxplus_k$ as illustrated in Figure 4. Hence, we can identify the vertices of $\tilde{\boxplus}_k$ with their colors.

Note that the original coloring c is *not* a $\tilde{\boxplus}_k$-coloring of G, as two vertices of G of the same color are adjacent in the gadget construction. However, adding self-loops to $\tilde{\boxplus}_k$ induces a proper coloring:

▷ Claim 5.4. Let $\tilde{\boxplus}_k^\circ$ be the graph obtained from $\tilde{\boxplus}_k$ by adding a self-loop to every vertex. Then the mapping $\hat{c}$ is a $\tilde{\boxplus}_k^\circ$-coloring of $\tilde{G}$.

Proof. We have to show that $\hat{c}$ is a homomorphism. To this end, consider an edge $\{x, y\}$ of the graph $\tilde{G}$. By construction of $\tilde{G}$, there are four cases for x and y:

- The vertices x and y are contained in the same clique. By definition of the mapping $\hat{c}$, we have that

  $$\hat{c}(x) = (i, j, \star_1) \text{ and } \hat{c}(y) = (i, j, \star_2)$$

  for some numbers $i, j \in [k]$ and $\star_1, \star_2 \in \{\rightarrow, \leftarrow, \uparrow, \downarrow, \nwarrow, \searrow\}$. If we have that $\star_1 = \star_2$, then the colors of x and y are the same, that is $\hat{c}(x) = \hat{c}(y)$. Hence, we have that $\{\hat{c}(x), \hat{c}(y)\} \in E(\tilde{\boxplus}_k^\circ)$, as we added all self-loops. Otherwise, if $\star_1 \neq \star_2$, then the graph $\tilde{\boxplus}_k$ has the edge $\{(i, j, \star_1), (i, j, \star_2)\}$, and hence we have that the edge $\{\hat{c}(x), \hat{c}(y)\}$ is in $E(\tilde{\boxplus}_k^\circ)$ by construction.
- The vertices x and y satisfy $x = v^{\searrow}$ and $y = v^{\nwarrow}$ for some $v \in V(G)$. By definition of the mapping $\hat{c}$, we have that

  $$\hat{c}(x) = (i, j, \searrow) \text{ and } \hat{c}(y) = (i, j, \nwarrow).$$

  By construction, we immediately get that $\{\hat{c}(x), \hat{c}(y)\} \in E(\tilde{\boxplus}_k^\circ)$.
- The vertices x and y satisfy $x = v^w_{\rightarrow}$ and $y = w^v_{\leftarrow}$ for some vertices $v, w \in V(G)$. By definition of the mapping $\hat{c}$, we have that

  $$\hat{c}(x) = (i, j, \rightarrow) \text{ and } \hat{c}(y) = (i, j+1, \leftarrow),$$

  which are again adjacent in $\tilde{\boxplus}_k^\circ$ by construction.

- The vertices x and y satisfy $x = v^w_\downarrow$ and $y = w^v_\uparrow$ for some vertices $v, w \in V(G)$. This case is similar to the previous case.

In total, we obtain that $\hat{c}$ is indeed a $\overset{\Large\color{gold}\ddot\smile}{\boxplus}^\circ_k$-coloring of the graph $\overset{\Large\color{gold}\ddot\smile}{G}$. ◁

Now let h be a homomorphism from $\overset{\Large\color{gold}\ddot\smile}{\boxplus}_k$ to $\overset{\Large\color{gold}\ddot\smile}{G}$ for some $\boxplus_k$-colored graph G. We call h *colorful* if for every color (or vertex) $(i, j, \star)$ in $V(\overset{\Large\color{gold}\ddot\smile}{\boxplus}_k)$, there is a vertex in the image of h that is colored with $(i, j, \star)$. Denote the set of all colorful homomorphisms from $\overset{\Large\color{gold}\ddot\smile}{\boxplus}_k$ to $\overset{\Large\color{gold}\ddot\smile}{G}$ with $\mathsf{cf\text{-}Hom}(\overset{\Large\color{gold}\ddot\smile}{\boxplus}_k \to \overset{\Large\color{gold}\ddot\smile}{G})$.

▷ Claim 5.5. The number of colorful homomorphisms from $\overset{\Large\color{gold}\ddot\smile}{\boxplus}_k$ to $\overset{\Large\color{gold}\ddot\smile}{G}$ can be computed in time

$$2^{|V(\overset{\Large\color{gold}\ddot\smile}{\boxplus}_k)|} \cdot |V(\overset{\Large\color{gold}\ddot\smile}{G})|^{O(1)}$$

by querying the oracle for $\#\textsc{Hom}(\mathcal{H} \to \ddot\smile)$. Further, every oracle query $(\hat{H}, \hat{G})$ satisfies that the size $|V(\hat{H})|$ only depends on the size $|V(\overset{\Large\color{gold}\ddot\smile}{\boxplus}_k)|$.

Proof. By the principle of "inclusion and exclusion", we have that the number of colorful homomorphisms can be computed as

$$\#\mathsf{cf\text{-}Hom}(\overset{\Large\color{gold}\ddot\smile}{\boxplus}_k \to \overset{\Large\color{gold}\ddot\smile}{G}) = \sum_{J \subseteq V(\overset{\Large\color{gold}\ddot\smile}{\boxplus}_k)} (-1)^{|J|} \cdot \#\mathsf{Hom}(\overset{\Large\color{gold}\ddot\smile}{\boxplus}_k \to \overset{\Large\color{gold}\ddot\smile}{G} \setminus J), \tag{9}$$

where $\overset{\Large\color{gold}\ddot\smile}{G} \setminus J$ is the graph obtained from $\overset{\Large\color{gold}\ddot\smile}{G}$ by deleting all vertices that are colored by $\hat{c}$ with a color in J. Hence we can compute, using the previous equation, the number of colorful homomorphisms in time $2^{|V(\overset{\Large\color{gold}\ddot\smile}{\boxplus}_k)|} \cdot |V(\overset{\Large\color{gold}\ddot\smile}{G})|^{O(1)}$ if an oracle to the function $A \mapsto \#\mathsf{Hom}(\overset{\Large\color{gold}\ddot\smile}{\boxplus}_k \to A)$ is provided. In particular, König graphs are closed under the removal of vertices: Deleting a vertex in a König graph is equivalent to deleting an edge in primal bipartite graph and bipartiteness is closed under the removal of edges. It hence suffices to restrict the graphs A to the class $\ddot\smile$.

Now recall that the graph $\overset{\Large\color{gold}\ddot\smile}{\boxplus}_k$ is planar. By the Excluded Grid Theorem [50], every class $\mathcal{H}$ of unbounded treewidth contains arbitrary large grids as minors. Furthermore, every planar graph is the minor of some grid [51]. As the class $\mathcal{H}$ is minor-closed, we hence obtain that $\overset{\Large\color{gold}\ddot\smile}{\boxplus}_k$ is contained in $\mathcal{H}$ for every $k \in \mathbb{N}$. Consequently, we can compute (9) using the given oracle for $\#\textsc{Hom}(\mathcal{H} \to \ddot\smile)$. ◁

Note that the composition of a colorful homomorphism and the $\overset{\Large\color{gold}\ddot\smile}{\boxplus}^\circ_k$-coloring $\hat{c}$ of $\overset{\Large\color{gold}\ddot\smile}{G}$ is a bijective homomorphism from $\overset{\Large\color{gold}\ddot\smile}{\boxplus}_k$ to $\overset{\Large\color{gold}\ddot\smile}{\boxplus}^\circ_k$. In particular, bijectivity yields that we do not need the self-loops of the graph $\overset{\Large\color{gold}\ddot\smile}{\boxplus}^\circ_k$. Consequently, we obtain a bijective endomorphism, that is, an automorphism of the graph $\overset{\Large\color{gold}\ddot\smile}{\boxplus}_k$. Formally, we can show the following:

▷ Claim 5.6. The number of colorful homomorphisms from $\overset{\Large\color{gold}\ddot\smile}{\boxplus}_k$ to $\overset{\Large\color{gold}\ddot\smile}{G}$ is the same as $\#\mathsf{cf\text{-}Hom}(\overset{\Large\color{gold}\ddot\smile}{\boxplus}_k \to \overset{\Large\color{gold}\ddot\smile}{G}) = \#\mathsf{Aut}(\overset{\Large\color{gold}\ddot\smile}{\boxplus}_k) \cdot \#\mathsf{cp\text{-}Hom}(\overset{\Large\color{gold}\ddot\smile}{\boxplus}^\circ_k \to \overset{\Large\color{gold}\ddot\smile}{G})$.

Proof. Observe that for an automorphism, self-loops are irrelevant: $\#\mathsf{Aut}(\overset{\Large\color{gold}\ddot\smile}{\boxplus}_k) = \#\mathsf{Aut}(\overset{\Large\color{gold}\ddot\smile}{\boxplus}^\circ_k)$.

Now, define two colorful homomorphisms to be equivalent if their image is equal. Then, every equivalence class has size $\#\mathsf{Aut}(\overset{\Large\color{gold}\ddot\smile}{\boxplus}_k)$ and is represented by a homomorphism for which the induced automorphism is the identity. This homomorphism is then not only colorful but also color-prescribed. ◁

Finally, we show that the number of homomorphisms from $\overset{\Large\color{gold}\ddot\smile}{\boxplus}_k^\circ$ to $\overset{\Large\color{gold}\ddot\smile}{G}$ is the same as the number of homomorphisms from $\boxplus_k$ to G: $\#\mathsf{cp\text{-}Hom}(\overset{\Large\color{gold}\ddot\smile}{\boxplus}_k^\circ \to \overset{\Large\color{gold}\ddot\smile}{G}) = \#\mathsf{cp\text{-}Hom}(\boxplus_k \to G)$.

The argument is depicted in Figure 4: Let $h \in \mathsf{cp\text{-}Hom}(\boxplus_k \to G)$ denote a color-prescribed homomorphism. We define the homomorphism $\hat{h} \in \mathsf{cp\text{-}Hom}(\overset{\Large\color{gold}\ddot\smile}{\boxplus}_k^\circ \to \overset{\Large\color{gold}\ddot\smile}{G})$ as follows: For every $i, j \in [k]$ we set

$$\begin{aligned}
\hat{h}(i,j,\rightarrow) &:= h(i,j)^{h(i,j+1)}_{\rightarrow}\\
\hat{h}(i,j,\leftarrow) &:= h(i,j)^{h(i,j-1)}_{\leftarrow}\\
\hat{h}(i,j,\downarrow) &:= h(i,j)^{h(i+1,j)}_{\downarrow}\\
\hat{h}(i,j,\uparrow) &:= h(i,j)^{h(i-1,j)}_{\uparrow}\\
\hat{h}(i,j,\searrow) &:= h(i,j)^{\searrow}\\
\hat{h}(i,j,\nwarrow) &:= h(i,j)^{\nwarrow}
\end{aligned}$$

The construction of $\overset{\Large\color{gold}\ddot\smile}{G}$ immediately yields that $\hat{h}$ is a (color-prescribed) homomorphism if h is color-prescribed homomorphism. Furthermore, the mapping $h \mapsto \hat{h}$ is a bijection. This concludes the proof. ◀

Now, consider the following application of Lemma 5.2.

▶ **Theorem 5.7.** *Let $\mathcal{C}$ be one of the classes of line-graphs, claw-free graphs or perfect graphs, or a non-empty union thereof. Further, let $\mathcal{H}$ be a recursively enumerable class of graphs.*

(1) *If the treewidth of the class $\mathcal{H}$ is bounded, then the problem $\#\textsc{Hom}(\mathcal{H} \to \mathcal{C})$ is solvable in polynomial time.*

(2) *Otherwise, if the class $\mathcal{H}$ is additionally minor-closed, the problem $\#\textsc{Hom}(\mathcal{H} \to \mathcal{C})$ is $\#\mathsf{W}[1]$-hard.*

Proof. We immediately obtain the reduction $\#\textsc{Hom}(\mathcal{H} \to \mathcal{C}) \leq_{\mathrm{T}}^{\mathrm{fpt}} \#\textsc{Hom}(\mathcal{H} \to \top)$. In particular, the reduction is the identity and preserves not only fixed-parameter tractability, but also polynomial-time tractability.

By the classification of Dalmau and Jonsson [19], the problem $\#\textsc{Hom}(\mathcal{H} \to \top)$ is solvable in polynomial time if the class $\mathcal{H}$ has bounded treewidth. If the class $\mathcal{H}$ has unbounded treewidth, $\#\mathsf{W}[1]$-hardness follows from Lemma 5.2, as the set of König graphs is a subset of claw-free graphs [3], a subset of perfect graphs [13] and, of course, a subset of line graphs. ◀

5.3 An Implicit Criterion for Hardness of Counting Homomorphisms in König Graphs

We complement the explicit criterion for hardness for the problem $\#\textsc{Hom}(\mathcal{H} \to \overset{\Large\color{gold}\ddot\smile}{\boxplus})$ from Theorem 5.7 (which only works if the class $\mathcal{H}$ is closed under taking minors) with the following implicit exhaustive complexity classification.

▶ **Theorem 5.8.** *Let $\mathcal{H}$ be a recursively enumerable class of graphs. Then the problem $\#\textsc{Hom}(\mathcal{H} \to \overset{\Large\color{gold}\ddot\smile}{\boxplus})$ is either fixed-parameter tractable or $\#\mathsf{W}[1]$-hard under parameterized Turing-reductions.*

In particular, Theorem 5.8 shows that the negative result from Section 3 does not apply to König graphs.

A central ingredient for the proof of Theorem 5.8 is the following lemma.

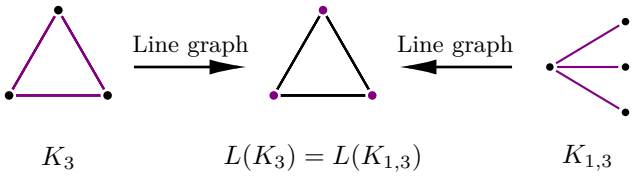

Figure 8 The claw $K_{1,3}$ and the triangle K_3 have the same line graph K_3.

▶ **Lemma 5.9.** *Let H be a graph. There exists a quantum graph $Q[H]$ such that we have for any bipartite graph G:*

$$\#\mathsf{Hom}(H \to L(G)) = \#\mathsf{Hom}(Q[H] \to G). \tag{10}$$

In particular, the mapping $H \mapsto Q[H]$ is computable.

The proof of Lemma 5.9 uses known transformations between linear combinations of homomorphisms, subgraphs and induced subgraphs (see e.g. Chapter 5.2.3 in [43] and [16, Section 3]), as well as Whitney's Isomorphism Theorem:

▶ **Theorem 5.10** ([58])**.** *Let H be a connected line graph that is not isomorphic to the triangle. Then the graph F such that $L(F) = H$ is uniquely defined up to isolated vertices. More precisely, every graph F' that satisfies $L(F') = H$ and that does not contain isolated vertices is isomorphic to F.*

Note that both the triangle and the claw have the triangle as line graph, consider Figure 8. The previous theorem states that the triangle is the only line graph whose primal graph is not uniquely defined. Furthermore, the Isomorphism Theorem (Theorem 5.10) allows us to define the following function; recall that $\mathcal{G}_P$ is the set of bipartite graphs.

$$L^{-1} : \♚ \to \mathcal{G}_P, \text{ such that } L^{-1}(L(G)) := G \text{ for every bipartite graph } G.$$

Note that the function L^{-1} is well-defined by Theorem 5.10 and the fact that the triangle is not bipartite. In particular, we have that $L^{-1}(K_3) = K_{1,3}$, where K_3 is the triangle and $K_{1,3}$ is the claw, that is, the complete bipartite graph with one vertex on the left side and three vertices on the right side (Again, consult Figure 8 for a visualization.) Similarly, it is well defined to write $L(F)$ for a König graph which is the line graph of the (uniquely determined) bipartite graph F without isolated vertices.

Further, again by Whitney's Isomorphism Theorem, we obtain the following lemma.

▶ **Lemma 5.11.** *Let H be a line graph and let $L(G)$ be a König graph. Then we have that*

$$\#\mathsf{IndSub}(H \to L(G)) = \begin{cases} \#\mathsf{Sub}(F \to G) & \text{if } H = L(F) \in \♚ \\ 0 & \text{otherwise} \end{cases}. \tag{11}$$

Proof. Assume first that the graph $H = L(F)$ is a König graph. By Theorem 5.10 and the fact that triangles are not bipartite, we have that G is the unique bipartite graph whose line graph is isomorphic to $L(G)$. Let $S \subseteq V(L(G)) = E(G)$ such that the induced subgraph $L(G)[S]$ is isomorphic to $L(F)$ and let G' be the subgraph of G with vertices

$$V(G') := \{v \in V(G) \mid \exists e \in S : v \in e\},$$

and edges $E(G') := S$.

By construction, the line graph of $L(G')$ is isomorphic to $L(F)$. Hence, we obtain that the graphs F and G' are isomorphic by Theorem 5.10 and the fact that none of the graphs F and G' is the triangle; note that a bipartite graph cannot contain a subgraph isomorphic to a triangle.

Now let G' be a subgraph of G that is isomorphic to F. As the graph F does not contain isolated vertices, we have that G' is determined by its set of edges. We set $S = E(G')$ and consider the induced subgraph $L(G)[S]$. By construction, we have that the graph $L(G)[S]$ is isomorphic to $L(G')$ which in turn is isomorphic to $L(F)$ (as the graphs G' and F are isomorphic). This shows correctness for $H = L(F) \in \♔$.

If the graph H is not a König graph, then H is not a triangle. Thus, we have that $H = L(F')$ for some non-bipartite graph F' which is uniquely determined (up to isolated vertices) by Theorem 5.10. In this case, the same argument as before shows that any induced subgraph of $L(G)$ that is isomorphic to $L(F')$ yields a subgraph of G that is isomorphic to F'. As the graph G is bipartite and the graph F' is not, such an induced subgraph cannot exist; note that bipartite graphs are closed under taking subgraphs. ◀

The last ingredient for the proof of Lemma 5.9 is the following well-known identity which relates (strong) embeddings and (induced) subgraphs.

▶ **Fact 5.12.** For all graphs H and G we have that

$$\#\mathsf{Emb}(H \to G) = \#\mathsf{Aut}(H) \cdot \#\mathsf{Sub}(H \to G) \text{ and} \tag{12}$$

$$\#\mathsf{StrEmb}(H \to G) = \#\mathsf{Aut}(H) \cdot \#\mathsf{IndSub}(H \to G). \tag{13}$$

Finally, we are ready to prove Lemma 5.9.

Proof of Lemma 5.9. We rely on a stepwise transformation of linear combinations of homomorphisms, embeddings, and strong embeddings as given by Lovász [43, Chapter 5.2.3] and as used by Curticapean, Dell and Marx [16, Section 3].

For the formal statement, we need to introduce some further notation. Given a graph H, we write $\mathsf{Part}(H)$ for the set of all partitions ρ of the vertex set $V(H)$ such that the quotient H/ρ is a spasm, that is, H/ρ does not contain self-loops. Furthermore, we write $H' \supseteq H$ if the graph H' can be obtained from H by adding edges. Now, let us state the transformations: For all graphs H and G, we have that

$$\#\mathsf{Hom}(H \to G) = \sum_{\rho \in \mathsf{Part}(H)} \#\mathsf{Emb}(H/\rho \to G), \text{ and} \tag{14}$$

$$\#\mathsf{Emb}(H \to G) = \sum_{\rho \in \mathsf{Part}(H)} \mu(\emptyset, \rho) \cdot \#\mathsf{Hom}(H/\rho \to G), \tag{15}$$

where μ is the so-called Möbius function over the partition lattice.[7] Furthermore, we have that

$$\#\mathsf{Emb}(H \to G) = \sum_{H' \supseteq H} \#\mathsf{StrEmb}(H' \to G), \text{ and} \tag{16}$$

$$\#\mathsf{StrEmb}(H \to G) = \sum_{H' \supseteq H} (-1)^{|E(H')| - |E(H)|} \cdot \#\mathsf{Emb}(H' \to G). \tag{17}$$

[7] As we do not need the formal definition of the Möbius function here, we refer the interested reader to e.g. [54] instead.

Now let us start with the construction of the quantum graph $Q[H]$ and the proof of Equation (10). We have that

$$\#\mathsf{Hom}(H \to L(G)) \overset{(14)}{=} \sum_{\rho \in \mathsf{Part}(H)} \#\mathsf{Emb}(H/\rho \to L(G))\,,$$
$$\#\mathsf{Emb}(H/\rho \to L(G)) \overset{(16)}{=} \sum_{H' \supseteq H/\rho} \#\mathsf{StrEmb}(H' \to L(G))\,, \text{ and}$$
$$\#\mathsf{StrEmb}(H' \to L(G)) \overset{(13)}{=} \#\mathsf{Aut}(H') \cdot \#\mathsf{IndSub}(H' \to L(G))\,.$$

Combining (11) with the fact that line graphs are closed under taking induced subgraphs [3], we obtain that

$$\#\mathsf{IndSub}(H' \to L(G)) = 0,$$

whenever H' is not a König graph. Consequently,

$$\#\mathsf{Emb}(H/\rho \to L(G)) = \sum_{\substack{H' \supseteq H/\rho \\ H' = L(F) \in \♔}} \#\mathsf{Aut}(H') \cdot \#\mathsf{IndSub}(H \to L(G)).$$

For a graph $H' = L(F) \in \♔$ we have that

$$\begin{aligned}\#\mathsf{IndSub}(H' \to L(G)) &\overset{(11)}{=} \#\mathsf{Sub}(F \to G) \\ &\overset{(12)}{=} \#\mathsf{Aut}(F)^{-1} \cdot \#\mathsf{Emb}(F \to G) \\ &\overset{(15)}{=} \#\mathsf{Aut}(F)^{-1} \cdot \sum_{\rho \in \mathsf{Part}(F)} \mu(\emptyset, \rho) \cdot \#\mathsf{Hom}(F/\rho \to G).\end{aligned}$$

We can thus successively apply the previous transformations and obtain:

$$\#\mathsf{Hom}(H \to L(G)) = \sum_{\rho \in \mathsf{Part}(H)} \sum_{\substack{H' \supseteq H/\rho \\ H' = L(F) \in \♔}} \sum_{\delta \in \mathsf{Part}(F)} \frac{\#\mathsf{Aut}(H') \cdot \mu(\emptyset, \delta)}{\#\mathsf{Aut}(F)} \cdot \#\mathsf{Hom}(F/\delta \to G)\,.$$

The desired quantum graph $Q[H]$ is obtained by collecting for isomorphic terms, that is,

$$Q[H] := \sum_{F'} \lambda_{F'} \cdot F'\,,$$

where

$$\lambda_{F'} := \sum_{\rho \in \mathsf{Part}(H)} \sum_{\substack{H' \supseteq H/\rho \\ H' = L(F) \in \♔}} \sum_{\substack{\delta \in \mathsf{Part}(F) \\ F/\delta = F'}} \frac{\#\mathsf{Aut}(H') \cdot \mu(\emptyset, \delta)}{\#\mathsf{Aut}(F)}\,.$$

◀

Using Lemma 5.9, we obtain a proof for Theorem 5.8 as follows.

▶ **Theorem 5.8** (repeated). *Let $\mathcal{H}$ be a recursively enumerable class of graphs. Then the problem $\#\textsc{Hom}(\mathcal{H} \to \♔)$ is either fixed-parameter tractable or $\#\mathsf{W}[1]$-hard under parameterized Turing-reductions.*

Proof. Let $P = P_1$ denote any path of length 1, and let $\mathcal{G}_P$ denote the class of all P-colorable graph; that is, $\mathcal{G}_P$ is the class of all bipartite graphs. Now, consider the following class of graphs

$$\hat{\mathcal{H}} := \bigcup_{H \in \mathcal{H}} \mathsf{supp}(Q[H]) \cap \mathcal{G}_P,$$

We show the following reductions

$$\#\textsc{Hom}(\mathcal{H} \to \hexagon) \equiv^{\text{fpt}}_{\text{T}} \#\textsc{Hom}(\hat{\mathcal{H}} \to \mathcal{G}_P).$$

Note that this imples Theorem 5.8 by the classification of F-colorable graphs (Theorem 4.1).

For the direction $\#\textsc{Hom}(\mathcal{H} \to \hexagon) \leq^{\text{fpt}}_{\text{T}} \#\textsc{Hom}(\hat{\mathcal{H}} \to \mathcal{G}_P)$, we assume that a graph $H \in \mathcal{H}$ and a König graph $L(G)$ are given. By Lemma 5.2, we can compute (in time only depending on H) the quantum graph $Q[H]$ that satisfies

$$\#\mathsf{Hom}(H \to L(G)) = \#\mathsf{Hom}(Q[H] \to G) = \sum_{F \in \mathsf{supp}(Q[H])} \lambda_F \cdot \#\mathsf{Hom}(F \to G), \tag{18}$$

where the λ_F are the coefficients of $Q[H]$. Now, as $L(G)$ is a König graph, we have that G is bipartite. Hence, by Observation B.1, there is no homomorphism from F to G whenever F is not bipartite, that is we have that $\#\mathsf{Hom}(F \to G) = 0$. Note that we can verify whether a graph $F \in \mathsf{supp}(Q[H])$ is bipartite in time only depending on $|V(H)|$. All further terms $\#\mathsf{Hom}(F \to G)$ with $F \in \mathcal{G}_P$ can be obtained by querying the oracle for the problem $\#\textsc{Hom}(\hat{\mathcal{H}} \to \mathcal{G}_P)$. Finally, we compute and the linear combination given by (18). This completes the first reduction.

For the other direction, $\#\textsc{Hom}(\hat{\mathcal{H}} \to \mathcal{G}_P) \leq^{\text{fpt}}_{\text{T}} \#\textsc{Hom}(\mathcal{H} \to \hexagon)$, we assume that graphs $F \in \hat{\mathcal{H}}$ and $G \in \mathcal{G}_{P_2}$ are given. By definition of the class $\hat{\mathcal{H}}$, we have that the graph F is a bipartite constituent of the quantum graph $Q[H]$ for some $H \in \mathcal{H}$. As $\mathcal{H}$ is recursively enumerable and the mapping $H \mapsto Q[H]$ is computable, we can compute in time only depending on $|V(F)|$ the quantum graph $Q[H]$.

By Fact 2.3, we have that the tensor product $G \times A$ is bipartite for every graph A as the graph G is bipartite. Therefore, the graph $L(G \times A)$ is a König graph for every (not necessarily bipartite) graph A. Hence, we can query the oracle for the problem $\#\textsc{Hom}(\mathcal{H} \to \hexagon)$ to compute for every graph A whose size only depends on $|V(F)|$ the following values:

$$\begin{aligned}
\#\mathsf{Hom}(H \to L(G \times A)) =& \#\mathsf{Hom}(Q[H] \to G \times A) \\
=& \sum_{F' \in \mathsf{supp}(Q[H])} \lambda_{F'} \cdot \#\mathsf{Hom}(F' \to G \times A) \\
=& \sum_{F' \in \mathsf{supp}(Q[H])} \lambda_{F'} \cdot \#\mathsf{Hom}(F' \to G) \cdot \#\mathsf{Hom}(F' \to A).
\end{aligned}$$

Now, (the proof of) Lemma B.4 shows that the induced system of linear equations is solvable for the proper choices of A. In particular, the size of those choices only depends on $|V(H)|$, which itself only depends on $|V(F)|$. As the graph F is a constituent of the quantum graph $Q[H]$, and thus the corresponding coefficient λ_F is non-zero, we can compute and output the number $\#\mathsf{Hom}(F \to G)$ in time only depending on $|V(H)|$, which itself only depends on $|V(F)|$. This completes the second reduction, and hence the proof. ◀

References

1 Michael O. Albertson and Karen L. Collins. Homomorphisms of 3-chromatic Graphs. *Discrete Mathematics*, 54(2):127–132, 1985. doi:10.1016/0012-365X(85)90073-1.

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

3 Lowell W. Beineke. Characterizations of Derived Graphs. *Journal of Combinatorial Theory*, 9(2):129 – 135, 1970. doi:https://doi.org/10.1016/S0021-9800(70)80019-9.

4 Arnab Bhattacharyya, Suprovat Ghoshal, Karthik C. S., and Pasin Manurangsi. Parameterized Intractability of Even Set and Shortest Vector Problem from Gap-ETH. *CoRR*, abs/1803.09717, 2018. URL: http://arxiv.org/abs/1803.09717, arXiv:1803.09717.

5 Arnab Bhattacharyya, Suprovat Ghoshal, Karthik C. S., and Pasin Manurangsi. Parameterized Intractability of Even Set and Shortest Vector Problem from Gap-ETH. In *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018, July 9-13, 2018, Prague, Czech Republic*, pages 17:1–17:15, 2018. doi:10.4230/LIPIcs.ICALP.2018.17.

6 Manuel Bodirsky and Martin Grohe. Non-dichotomies in constraint satisfaction complexity. In *Automata, Languages and Programming, 35th International Colloquium, ICALP 2008, Reykjavik, Iceland, July 7-11, 2008, Proceedings, Part II - Track B: Logic, Semantics, and Theory of Programming & Track C: Security and Cryptography Foundations*, pages 184–196, 2008. doi:10.1007/978-3-540-70583-3_16.

7 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.

8 Andrei A. Bulatov. The complexity of the counting constraint satisfaction problem. *J. ACM*, 60(5):34:1–34:41, 2013. doi:10.1145/2528400.

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

10 Jin-Yi Cai and Xi Chen. Complexity of counting CSP with complex weights. In *Proceedings of the 44th Symposium on Theory of Computing Conference, STOC 2012, New York, NY, USA, May 19 - 22, 2012*, pages 909–920, 2012. doi:10.1145/2213977.2214059.

11 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.

12 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.

13 Maria Chudnovsky, Neil Robertson, Paul Seymour, and Robin Thomas. The Strong Perfect Graph Theorem. *Annals of Mathematics*, 164(1):51–229, 2006. URL: http://www.jstor.org/stable/20159988.

14 Stephen A. Cook. The Complexity of Theorem-Proving Procedures. In *Proceedings of the 3rd Annual ACM Symposium on Theory of Computing, May 3-5, 1971, Shaker Heights, Ohio, USA*, pages 151–158, 1971. doi:10.1145/800157.805047.

15 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/.

16 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.

17 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.

18 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.

19 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.

20 Holger Dell, Marc Roth, and Philip Wellnitz. Counting Answers to Existential Questions. In *46th International Colloquium on Automata, Languages, and Programming, ICALP 2019, July 8-12, 2019, Patras, Greece, to appear*, 2019.

21 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.

22 Rod Downey and Michael Fellows. Fixed-parameter tractability and completeness III: Some structural aspects of the W hierarchy. In *Complexity theory*, pages 191–225. Cambridge University Press, 1993.

23 Martin E. Dyer and Catherine S. Greenhill. The complexity of counting graph homomorphisms. *Random Struct. Algorithms*, 17(3-4):260–289, 2000.

24 Martin E. Dyer and David Richerby. On the complexity of #CSP. In *Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC 2010, Cambridge, Massachusetts, USA, 5-8 June 2010*, pages 725–734, 2010. doi:10.1145/1806689.1806789.

25 Jack Edmonds. Paths, Trees, and Flowers. *Canadian Journal of Mathematics*, 17:449–467, 1965. doi:10.4153/CJM-1965-045-4.

26 David Eppstein. Subgraph Isomorphism in Planar Graphs and Related Problems. *J. Graph Algorithms Appl.*, 3(3):1–27, 1999. doi:10.7155/jgaa.00014.

27 Tomás Feder and Moshe Y. Vardi. The Computational Structure of Monotone Monadic SNP and Constraint Satisfaction: A Study through Datalog and Group Theory. *SIAM J. Comput.*, 28(1):57–104, 1998. doi:10.1137/S0097539794266766.

28 Jörg Flum and Martin Grohe. The Parameterized Complexity of Counting Problems. *SIAM J. Comput.*, 33(4):892–922, 2004. doi:10.1137/S0097539703427203.

29 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.

30 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.

31 M. R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.

32 M. R. Garey, David S. Johnson, and H. C. So. An Application of Graph Coloring to Printed Circuit Testing (Working Paper). In *16th Annual Symposium on Foundations of Computer Science, Berkeley, California, USA, October 13-15, 1975*, pages 178–183, 1975. doi:10.1109/SFCS.1975.3.

33 Oded Goldreich. *Computational Complexity - A Conceptual Perspective*. Cambridge University Press, 2008.

34 Martin Grohe. The complexity of homomorphism and constraint satisfaction problems seen from the other side. *J. ACM*, 54(1):1:1–1:24, 2007. doi:10.1145/1206035.1206036.

35 Martin Grohe, Thomas Schwentick, and Luc Segoufin. When is the evaluation of conjunctive queries tractable? In *Proceedings on 33rd Annual ACM Symposium on Theory of Computing, July 6-8, 2001, Heraklion, Crete, Greece*, pages 657–666, 2001. doi:10.1145/380752.380867.

36 Geňa Hahn and Claude Tardif. Graph homomorphisms: Structure and Symmetry. In *Graph symmetry*, pages 107–166. Springer, 1997.

37 Frank Harary. *Graph theory*. Addison-Wesley series in mathematics. Addison-Wesley Pub. Co., 1969.

38 Pavol Hell and Jaroslav Nesetril. On the Complexity of H-coloring. *J. Comb. Theory, Ser. B*, 48(1):92–110, 1990. doi:10.1016/0095-8956(90)90132-J.

39 Pavol Hell and Jaroslav Nešetřil. *Graphs and Homomorphisms*. Oxford Lecture Series in Mathematics and Its Applications. Oxford University Press, Oxford, 2004. URL: http://cds.cern.ch/record/1413573.

40 Richard E. Ladner. On the structure of polynomial time reducibility. *J. ACM*, 22(1):155–171, 1975. doi:10.1145/321864.321877.

41 Philippe G. H. Lehot. An Optimal Algorithm to Detect a Line Graph and Output Its Root Graph. *J. ACM*, 21(4):569–575, October 1974. doi:10.1145/321850.321853.

42 László Lovász. Kneser's Conjecture, Chromatic Number, and Homotopy. *J. Comb. Theory, Ser. A*, 25(3):319–324, 1978. doi:10.1016/0097-3165(78)90022-5.

43 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.

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

45 Hermann A. Maurer, Ivan Hal Sudborough, and Emo Welzl. On the Complexity of the General Coloring Problem. *Information and Control*, 51(2):128–145, 1981. doi:10.1016/S0019-9958(81)90226-6.

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

47 Jaroslav Nešetřil. Homomorphisms of Derivative Graphs. *Discrete Mathematics*, 1(3):257–268, 1971. doi:10.1016/0012-365X(71)90014-8.

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

49 Rolf Niedermeier. Invitation to fixed-parameter algorithms. *Oxford Lecture Series in Mathematics and its Applications*, 31, 2002.

50 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.

51 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.

52 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.

53 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.

54 Richard P. Stanley. *Enumerative Combinatorics: Volume 1*. Cambridge University Press, 2011.

55 Julian R. Ullmann. An algorithm for subgraph isomorphism. *J. ACM*, 23(1):31–42, 1976. URL: http://doi.acm.org/10.1145/321921.321925, doi:10.1145/321921.321925.

56 Mario Valencia-Pabon and Juan-Carlos Vera. On the diameter of kneser graphs. *Discrete Mathematics*, 305(1):383 – 385, 2005. doi:https://doi.org/10.1016/j.disc.2005.10.001.

57 Leslie G. Valiant. The Complexity of Computing the Permanent. *Theor. Comput. Sci.*, 8:189–201, 1979. doi:10.1016/0304-3975(79)90044-6.

58 Hassler Whitney. Congruent Graphs and the Connectivity of Graphs. In *Hassler Whitney Collected Papers*, pages 61–79. Birkhäuser Boston, 1992. doi:https://doi.org/10.1007/978-1-4612-2972-8_4.

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

A On Hardness of #Hom($\mathcal{H} \to \top$)

In this section, we take a closer look at the proof of the following complexity classification which is due to Dalmau and Jonsson.

▶ **Theorem A.1** ([19]). *Let $\mathcal{H}$ be a recursively enumerable class of graphs.*
(1) *If the treewidth of $\mathcal{H}$ is bounded then $\#\textsc{Hom}(\mathcal{H} \to \top)$ is solvable in polynomial time.*
(2) *Otherwise, $\#\textsc{Hom}(\mathcal{H} \to \top)$ is $\#\mathsf{W}[1]$-hard under parameterized Turing-reductions.*

In particular, we are interested in the proof of Statement (2) of Theorem A.1, that is $\#\mathsf{W}[1]$-hardness for the problem $\#\textsc{Hom}(\mathcal{H} \to \top)$. The strategy of the proof is a line of reasoning based on the Excluded Grid Theorem[8], which is, by now, well-established (see e.g. [19, 35, 15, 20]). Our goal in this section is to show the following consequences of the known proofs of Statement (2) of Theorem A.1; recall that a core is a graph without a homomorphism from itself to any of its proper subgraphs.

▶ **Lemma A.2.** *Let $\mathcal{H}$ be a recursively enumerable class of graphs of unbounded treewidth.*
(1) *There exists a parameterized Turing-reduction from the problem $\#\textsc{Clique}$ to the problem $\#\textsc{Hom}(\mathcal{H} \to \top)$ such that every oracle query (H, G) satisfies that the graph G is H-colorable.*
(2) *If, additionally, the class $\mathcal{H}$ only contains connected cores, then there exists a parameterized weakly parsimonious reduction from the problem $\#\textsc{Clique}$ to the problem $\#\textsc{Hom}(\mathcal{H} \to \top)$ such that every pair (H, G) in the image of the reduction satisfies that the graph G is connected and H-colorable.*

To accommodate readers unfamiliar with the proof of Theorem A.1 on the one hand, but to refrain from including the proof in full detail on the other hand, we provide an outline only. For technical reasons, we start by proving that the instances of the problem $\#\textsc{Clique}$ can be assumed to be connected graphs.

▶ **Lemma A.3.** *Let $\#\textsc{ConnClique}$ be the problem of, given a* connected *graph G and a positive integer k, computing the number of cliques of size k in G. Then we have that*

$$\#\textsc{Clique} \leq^{\text{fpt}} \#\textsc{ConnClique}.$$

Proof. Let (G, k) be an instance of $\#\textsc{Clique}$. If the number k is 1, then the number of k-cliques in G is just the number of vertices $|V(G)|$ of G. Hence, the reduction can output $(P_{|V(G)|-1}, 2)$, where P_i is the path with $i+1$ edges.

If the number k is 2, then the number of k-cliques in G is just the number of edges $|E(G)|$ of G. Hence, the reduction can output $(P_{|E(G)|-1}, 2)$.

Otherwise, let $C_1 \ldots, C_n$ be the connected components of G. For each $i \in 1, \ldots, n-1$, we add an edge between an arbitrary vertex in component C_i and an arbitrary vertex in component C_{i+1}. As the number k is at least 3, this operation does not change the number of k-cliques and thus the reduction can output the modified connected graph and k. ◀

Outline of the proofs of (2) of Theorem A.1 and Lemma A.2. Let $\mathcal{H}$ denote a recursively enumerable class of graphs of unbounded treewidth. The goal is to show that the problem $\#\textsc{Hom}(\mathcal{H} \to \top)$ is $\#\mathsf{W}[1]$-hard.

[8] Recall that the Excluded Grid Theorem states that every class $\mathcal{H}$ of unbounded treewidth contains arbitrarily large grid minors [50].

The first step is the reduction from #Clique to #cp-Hom($\boxplus \to \top$), where $\boxplus$ is the set of all $k \times k$ square grids $\boxplus_k$ for $k \in \mathbb{N}$. Intuitively, given an instance (G, k) of #Clique, we construct a $\boxplus_k$-colored graph G' as follows: For each $i \in \{1, \dots, k\}$, we add the set

$$V_{i,i} := \{(v, v) \mid v \in V(G)\}$$

to the vertices of G'. For every $i, j \in \{1, \dots, k\}$ with $i \neq j$, we add the set

$$V_{i,j} := \{(u, v) \mid \{u, v\} \in E(G)\}$$

to the vertices of G'. Finally, we add an edge between two vertices $(v, u) \in V_{i,j}$ and $(v', u') \in V_{i',j'}$ if $v = v'$ and $i = i'$, or if $u = u'$ and $j = j'$. It can the be shown that the number of k-cliques in G equals the number

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

where the factor $(k!)^{-1}$ stems from the fact that the vertices of a k-clique are not ordered. Furthermore, the resulting graph G' is connected if G is connected and G' is $\boxplus_k$-colorable given by the homomorphism h that maps every vertex in $V_{i,j}$ to the grid vertex (i, j).

The second step relies on the Excluded Grid Theorem [50]: If the class $\mathcal{H}$ has unbounded treewidth, then for any number k, there is a graph $H_k \in \mathcal{H}$ that has the grid $\boxplus_k$ as a minor. Using this property of the class $\mathcal{H}$, it can be shown that

$$\#\text{cp-Hom}(\boxplus \to \top) \leq^{\text{fpt}} \#\text{cp-Hom}(\mathcal{H} \to \top).$$

A very clear presentation of this reduction is given by Curticapean [15, Lemma 5.8].[9] In particular, given graphs $\boxplus_k$ and G, the reduction outputs a pair $(H_k, \hat{G})$ such that the graph $\hat{G}$ is H_k-colorable and $\hat{G}$ is connected if both G and H_k are connected. Furthermore, note that the graph H_k can be found as $\mathcal{H}$ is recursively enumerable.

We are now able to prove the second item of Lemma A.2: Using Lemma A.3 and the properties of the previous reductions, we can, on input G and k, compute in time $f(k) \cdot |V(G)|^{O(1)}$ (for some computable function f), a pair $(H_k, \hat{G})$ of graphs such that

(a) the graph $\hat{G}$ is connected,
(b) the graph $\hat{G}$ is H_k-colorable by some coloring c, and
(c) the number of k-cliques in G is precisely $\#\mathsf{cp\text{-}Hom}(H_k \to \hat{G}) \cdot (k!)^{-1}$.

Now recall that the condition of the second item of Lemma A.2 states that the graph H_k is a core. Let h be a (not necessarily color-prescribed) homomorphism from H_k to $\hat{G}$. Then, the composition of h and the coloring c is an endomorphism of H_k. As the graph H_k is a core, it has no homomorphism to a proper subgraph. Hence, the homomorphism $h \circ c$ is an automorphism and, in particular, h is color-prescribed if and only if $h \circ c$ is the identity. It is then straightforward[10] to show that

$$\#\mathsf{cp\text{-}Hom}(H_k \to \hat{G}) = \#\mathsf{Hom}(H_k \to \hat{G}) \cdot \#\mathsf{Aut}(H_k)^{-1}.$$

This concludes the proof of the second item of Lemma A.2.

For the first item, we cannot assume that the graph $\hat{G}$ is connected, as the graph H_k might be disconnected. However, we still obtain an algorithm that, on input G and k, computes in time $f(k) \cdot |V(G)|^{O(1)}$ (for some computable function f), a pair $(H_k, \hat{G})$ of graphs such that

[9] Note that Curticapean uses the problem #PartitionedSub($\mathcal{H}$) which, however, is equivalent to the problem #cp-Hom($\mathcal{H} \to \top$) [15, Definition 5.2 and Remark 5.3].

[10] A reader familiar with group actions will find a very easy proof based on the observation that the automorphism group of H_k acts on the set $\mathsf{Hom}(H_k \to \hat{G})$.

(i) the graph $\hat{G}$ is H_k-colorable by some coloring c, and
(ii) the number of k-cliques in G is precisely $\#\mathsf{cp\text{-}Hom}(H_k \to \hat{G}) \cdot (k!)^{-1}$.

Now, using the principle of inclusion and exclusion, it is possible to compute the number N of homomorphisms h from H_k to $\hat{G}$ such that $h \circ c$ is an automorphism in time $O(2^k) \cdot |V(\hat{G})|^{O(1)}$ if an oracle to $\#\mathsf{cp\text{-}Hom}(H_k \to \star)$ is provided. More precisely, we have that

$$N = \sum_{J \subseteq V(H_k)} (-1)^{|J|} \cdot \#\mathsf{Hom}(H_k \to \hat{G} \setminus J),$$

where $\hat{G} \setminus J$ is the graph obtained from $\hat{G}$ by deleting all vertices v for which $c(v) \in J$. Having obtained N, for the same reasons as in the previous case, the Turing-reduction can output

$$N \cdot \#\mathsf{Aut}(H_k)^{-1}.$$

The first item of Lemma A.2 now holds as the graph $\hat{G}$ is H_k-colorable and hence every subgraph $\hat{G} \setminus J$ is H_k-colorable as well. ◀

Note that the proofs of Lemma A.3 and the second item of Lemma A.2 immediately show the following consequence for the decision version and parameterized many-one reductions [29, Definition 2.1]:

▶ **Corollary A.4.** *Let $\mathcal{H}$ be a recursively enumerable class of connected cores of unbounded treewidth. Then there exists a parameterized many-one reduction from* Clique *to* Hom$(\mathcal{H} \to \top)$ *such that every pair (H, G) in the image of the reduction satisfies that G is connected and H-colorable.*

B Proofs of Section 4

B.1 Proof of Theorem 4.1

We prove Theorem 4.1 in two steps. First, in Lemma B.2, we show a polynomial-time algorithm for graph classes $\mathcal{H}_{easy}$ that do not contain F-colorable graphs of arbitrarily large treewidth. After that, we prove $\#W[1]$-hardness for all other graph classes.

Polynomial-Time Algorithm for the Tractable Cases

Let $\mathcal{H}_{easy}$ denote any graph class such that for any graph $H \in \mathcal{H}_{easy}$, either H has a treewidth of at most c, or H is not F-colorable; where $c = c(\mathcal{H}_{easy})$ is a constant only depending on $\mathcal{H}_{easy}$. We obtain a polynomial-time algorithm for the (PPC) problem $\#\mathsf{Hom}(\mathcal{H}_{easy} \to \mathcal{G}_F)$ as follows. Given graphs $H \in \mathcal{H}_{easy}$ and $G \in \mathcal{G}_F$, we check, using Bodlaender's Algorithm [7], whether H has a treewidth $\mathsf{tw}(H)$ of at most $c(\mathcal{H}_{easy})$. Next, if $\mathsf{tw}(H) \leq c(\mathcal{H}_{easy})$, we use the standard dynamic programming algorithm due to Díaz et. al [21] to compute $\#\mathsf{Hom}(H \to G)$. Otherwise, that is if $\mathsf{tw}(H) > c(\mathcal{H}_{easy})$, we output 0, as H is not F-colorable by definition of $\mathcal{H}_{easy}$. This last step is justified by the following observation.

▶ **Observation B.1.** Let graphs F, G, and H be given. If there is no homomorphism from H to F, but a homomorphism from G to F, then there is no homomorphism from H to G.

Proof. Choose any homomorphism g from G to F and suppose there was a homomorphism h from H to G. As the concatenation of two homomorphisms is again a homomorphism, in particular $f \circ g : H \to F$ is again a homomorphism, which is a contradiction to the assumption that there is no homomorphism from H to F. ◀

In total, we obtain the following algorithm.

▶ **Lemma B.2.** *For any graph classes $\mathcal{H}_{easy}$ and $\mathcal{G}_F$, the (PPC) problem $\#\mathsf{Hom}(\mathcal{H}_{easy} \to \mathcal{G}_F)$ can be solved in polynomial time.*

Proof. The correctness follows directly from the definition of $\mathcal{H}_{easy}$ and Observation B.1.

For the running time, set $c = c(\mathcal{H}_{easy})$, $k = |V(H)|$, and $n := |V(G)|$ Checking whether the given graph has a treewidth of at most c takes time $c^{O(c^3)} \cdot k$ using Bodlaender's algorithm[7]. Next, computing the number of homomorphisms from a graph with treewidth at most c takes time $\text{poly}(k, c) \cdot n^{c+O(1)}$ due to Díaz et. al [21]. Hence in total, our algorithm has a running time of $\text{poly}(k, c) \cdot n^{c+O(1)}$, which is polynomial, thus completing the proof. ◀

$\#W[1]$-Hardness for the Intractable Cases

It remains to demonstrate $\#\mathsf{W}[1]$-hardness of $\#\mathsf{Hom}(H \to \mathcal{G}_F)$ whenever the treewidth of the intersection $\mathcal{H} \cap \mathcal{G}_F$ is unbounded. However, as we have seen in Lemma A.2, the existing hardness proof already shows the desired stronger result.

▶ **Observation B.3.** Let $\mathcal{H}$ be a recursively enumerable class of graphs such that the treewidth of $\mathcal{H} \cap \mathcal{G}_F$ is unbounded. Then we have that

$$\#\textsc{Clique} \leq_{\mathrm{T}}^{\mathrm{fpt}} \#\textsc{Hom}(\mathcal{H} \to \mathcal{G}_F).$$

Proof. We use Lemma A.2 (1) for the class $\mathcal{H} \cap \mathcal{G}_F$: As every oracle query (H, G) satisfies that G is H-colorable by some coloring c and every graph $H \in \mathcal{H} \cap \mathcal{G}_F$ is F-colorable by some coloring c', we obtain that the composition $c \circ c'$ is an F-coloring of G. ◀

▶ **Theorem 4.1** (repeated). *Let F be a graph, and let $\mathcal{H}$ be a recursively enumerable class of graphs.*

(1) *If the treewidth of $\mathcal{H} \cap \mathcal{G}_F$ is bounded then the PPC problem $\#\textsc{Hom}(\mathcal{H} \to \mathcal{G}_F)$ is polynomial-time solvable.*

(2) *Otherwise, the problem $\#\textsc{Hom}(\mathcal{H} \to \mathcal{G}_F)$ is $\#\mathsf{W}[1]$-hard.*

Proof. Holds by Lemma B.2 and Observation B.3. ◀

B.2 Proof of Theorem 4.3

We start with the proof of the following lemma.

▶ **Lemma B.4.** *Let F be a fixed graph and let Q be a quantum graph such that $\mathsf{supp}(Q) \subseteq \mathcal{G}_F$. There exists a deterministic algorithm $\mathbb{A}$ that is given oracle access to the problem $\#\textsc{Hom}(Q \to \star)$ and that, on input an F-colorable graph G, computes the number $\#\mathsf{Hom}(H \to G)$ for every constituent H of Q. Furthermore, there are computable functions f and s such that the running time of the algorithm $\mathbb{A}$ is bounded by $f(|Q|) \cdot |V(G)|^{O(1)}$ and every graph G' for which the oracle is queried is F-colorable and has at most $s(|Q|) \cdot |V(G)|$ vertices.*

Proof. We follow the lines of the proof of Lemma 3.6 in [16]: Given an F-colorable graph G, we wish to query the oracle for $(Q, G \times H')$ for certain graphs H'. Recall that the tensor product satisfies

$$\#\mathsf{Hom}(A \to B \times C) = \#\mathsf{Hom}(A \to B) \cdot \#\mathsf{Hom}(A \to C),$$

for all graphs A, B and C. Curticapean, Dell and Marx [16] discovered that a deep result of Lovász implies the existence of graphs $H_1, \ldots, H_\ell$ such that the following system of linear equations has a unique solution.

$$\#\mathsf{Hom}(Q \to G \times H_i) = \#\sum_H \lambda_H \cdot \#\mathsf{Hom}(H \to G \times H_i) = \#\sum_H c_H \cdot \#\mathsf{Hom}(H \to H_i),$$

where $c_H := \lambda_H \cdot \#\mathsf{Hom}(H \to G)$. In particular, the graphs $H_1, \ldots, H_\ell$ can be computed in time only depending on H. Consequently the number $\#\mathsf{Hom}(H \to G)$ can be computed whenever $\lambda_H \neq 0$ by standard Gaussian elimination.

Now, the only catch is the fact that we are only allowed to query the oracle for F-colorable graphs. However, by Fact 2.3, the tensor product of an F-colorable graph with another (not necessarily F-colorable) graph is always F-colorable. Consequently, the original proof of Curticapean, Dell and Marx [16] transfers without modification to the F-colored setting. ◀

▶ **Theorem 4.3** (repeated). *Let F be a fixed graph and let $\mathcal{H}$ be a recursively enumerable class of graphs.*

(1) *If the matching number of $\mathcal{H} \cap \mathcal{G}_F$ is bounded then the problem $\#\textsc{Sub}(\mathcal{H} \to \mathcal{G}_F)$ is polynomial-time solvable.*

(2) *Otherwise, the problem $\#\textsc{Sub}(\mathcal{H} \to \mathcal{G}_F)$ is $\#\mathsf{W}[1]$-hard.*

Proof. Proving (1) is easy: Let c be the constant upper bound on the matching number of graphs in $\mathcal{H} \cap \mathcal{G}_F$. Now, given $H \in \mathcal{H}$ and $G \in \mathcal{G}_F$, we first compute the matching number of H in polynomial time by the Blossom-Algorithm [25]. If the result is greater than c, the promise tells us that $H \notin \mathcal{G}_F$, in which case we can output 0 as H would be F-colorable if it was isomorphic to a subgraph of $G \in \mathcal{G}_F$. Otherwise, we use the algorithm given by the first item of Theorem 4.2.

For $\#\mathsf{W}[1]$-hardness, we construct a reduction from the problem

$$\#\textsc{Hom}(\mathsf{spasms}(\mathcal{H}) \cap \mathcal{G}_F \to \mathcal{G}_F),$$

where $\mathsf{spasms}(\mathcal{H})$ is the set of all spasms of graphs in $\mathcal{H}$. We start with the following observation.

▷ Claim B.5. The class $\mathsf{spasms}(\mathcal{H}) \cap \mathcal{G}_F$ has unbounded treewidth if the class $\mathcal{H} \cap \mathcal{G}_F$ has unbounded matching number.

Proof. Let $b \in \mathbb{N}$ be a positive integer. We show that there exists a graph of treewidth at least b in the class $\mathsf{spasms}(\mathcal{H}) \cap \mathcal{G}_F$. By assumption, there is a graph $H \in \mathcal{H} \cap \mathcal{G}_F$ such that H contains a matching M of size at least $|E(F)| \cdot b^2$; recall that $|E(F)|$ is a constant as the graph F is fixed.

Now let c be an F-coloring of H. For every edge $e = \{u, v\} \in M$, we let $c(e) \in E(F)$ be the image of e under the coloring. As $|M| \geq |E(F)| \cdot b^2$, we have that there is an edge $\hat{e} = \{\hat{u}, \hat{v}\} \in E(F)$ such that $c(e) = \hat{e}$ for at least b^2 many edges of M. More precisely, there are edges $\{u_1, v_1\}, \ldots, \{u_{b^2}, v_{b^2}\} \in M$ such that $c(u_i) = \hat{u}$ and $c(v_i) = \hat{v}$ for all $i \in \{1, \ldots, b^2\}$. As c is a coloring (and thus a homomorphism), we have that the sets $U := \{u_1, \ldots, u_{b^2}\}$ and $V := \{v_1, \ldots, v_{b^2}\}$ are independent. Consequently, we can contract vertices in U and V such that the matching M becomes a complete bipartite graph $K_{b,b}$ with b vertices on each side.

Let ρ be the induced partition on $V(H)$. Then the quotient graph H/ρ contains as subgraph $K_{b,b}$, and H/ρ is still F-colorable as we only identified vertices that are contained in the same pre-image of the F-coloring c. Furthermore, H/ρ is a spasm as we did not create self-loops. Thus, we have that $H/\rho \in \mathsf{spasms}(\mathcal{H}) \cap \mathcal{G}_F$. As the treewidth of $K_{b,b}$ is b and the

treewidth of a graph cannot increase by taking subgraphs, we obtain that H/ρ has treewidth at least b. ◁

Consequently, a reduction from $\#\textsc{Hom}(\mathsf{spasms}(\mathcal{H}) \cap \mathcal{G}_F \to \mathcal{G}_F)$ to $\#\textsc{Sub}(\mathcal{H} \to \mathcal{G}_F)$ shows $\#\mathsf{W}[1]$-hardness of the latter problem by Theorem 4.1. For the construction of the reduction, we use the known fact that, given a graph H, there exists a quantum graph $Q[H]$ with $\mathsf{supp}(Q) = \mathsf{spasms}(H)$ and $\#\mathsf{Hom}(Q[H] \to G) = \#\mathsf{Sub}(H \to G)$ for every graph G [43, Equation (6.2)], [16]. Therefore, given a graph $\hat{H} \in \mathsf{spasms}(\mathcal{H}) \cap \mathcal{G}_F$ we can find (in time only depending on $\hat{H}$) a graph $H \in \mathcal{H}$ such that we have for all graphs G that

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

and $\hat{H} \in \mathsf{supp}(Q[H])$. Now let $\hat{Q}[H]$ be the quantum graph obtained from $Q[H]$ by deleting all constituents that are not F-colorable. If G is F-colorable, we obtain from the previous equation that

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

as $\#\mathsf{Hom}(H' \to G) = 0$ for all graphs H' that are not F-colorable: Assuming otherwise, there exists a homomorphism h from H' to G which yields an F-coloring of H' when composed with the F-coloring of G, contradicting the assumption that H' is not F-colorable.

It follows that an oracle query (H, G) for $\#\textsc{Sub}(\mathcal{H} \to \mathcal{G}_F)$ computes $\#\mathsf{Hom}(\hat{Q}[H] \to G)$. As $\mathsf{supp}(\hat{Q}[H]) = \mathsf{spasms}(H) \cap \mathcal{G}_F$, we can use Lemma B.4, which concludes the proof. ◀