The Complexity of Counting Surjective Homomorphisms and Compactions*

Jacob Focke, Leslie Ann Goldberg and Stanislav Živný†

October 6, 2017

Abstract

A homomorphism from a graph G to a graph H is a function from the vertices of G to the vertices of H that preserves edges. A homomorphism is *surjective* if it uses all of the vertices of H and it is a *compaction* if it uses all of the vertices of H and all of the non-loop edges of H. Hell and Nešetřil gave a complete characterisation of the complexity of deciding whether there is a homomorphism from an input graph G to a fixed graph H. A complete characterisation is not known for surjective homomorphisms or for compactions, though there are many interesting results. Dyer and Greenhill gave a complete characterisation of the complexity of counting homomorphisms from an input graph G to a fixed graph H. In this paper, we give a complete characterisation of the complexity of counting surjective homomorphisms from an input graph G to a fixed graph H and we also give a complete characterisation of the complexity of counting compactions from an input graph G to a fixed graph H.

The full version containing detailed proofs is available at http://arxiv.org/abs/1706.08786. The theorem numbering here matches the full version.

*The research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013) ERC grant agreement no. 334828 and under the European Union's Horizon 2020 research and innovation programme (grant agreement No 714532). Stanislav Živný was supported by a Royal Society University Research Fellowship. The paper reflects only the authors' views and not the views of the ERC or the European Commission. The European Union is not liable for any use that may be made of the information contained therein.

†University of Oxford, UK. jacob.focke@cs.ox.ac.uk, leslie.goldberg@cs.ox.ac.uk, standa.zivny@cs.ox.ac.uk

1 Introduction

A homomorphism from a graph G to a graph H is a function from $V(G)$ to $V(H)$ that preserves edges. That is, the function maps every edge of G to an edge of H. Many structures in graphs, such as proper colourings, independent sets, and generalisations of these, can be represented as homomorphisms, so the study of graph homomorphisms has a long history in combinatorics [2, 4, 18, 19, 22, 24].

Much of the work on this problem is algorithmic in nature. A very important early work is Hell and Nešetřil's paper [20], which gives a complete characterisation of the complexity of the following decision problem, parameterised by a fixed graph H: "Given an input graph G, determine whether there is a homomorphism from G to H." Hell and Nešetřil showed that this problem can be solved in polynomial time if H has a loop or is bipartite and that it is NP-complete otherwise. An important generalisation of the homomorphism decision problem is the list-homomorphism decision problem. Here, in addition to the graph G, the input specifies, for each vertex v of G, a list S_v of permissible vertices of H. The problem is to determine whether there is a homomorphism from G to H that maps each vertex v of G to a vertex in S_v. Feder, Hell and Huang [11] gave a complete characterisation of the complexity of this problem. This problem can be solved in polynomial time if H is a so-called bi-arc graph, and it is NP-complete otherwise.

More recent work has restricted attention to homomorphisms with certain properties. A function from $V(G)$ to $V(H)$ is *surjective* if every element of $V(H)$ is the image of at least one element of $V(G)$. So a homomorphism from G to H is surjective if every vertex of H is "used" by the homomorphism. There is still no complete characterisation of the complexity of determining whether there is a surjective homomorphism from an input graph G to a graph H, despite an impressive collection of results [1, 15, 16, 17, 25]. A homomorphism from $V(G)$ to $V(H)$ is a *compaction* if it uses every vertex of H and also every non-loop edge of H (so it is surjective both on $V(H)$ and on the non-loop edges in $E(H)$). Compactions have been studied under the name "homomorphic image" [18, 22] and even under the name "surjective homomorphism" [6, 24]. Once again, despite much work [1, 27, 28, 29, 30, 31], there is still no characterisation of the complexity of determining whether there is a compaction from an input graph G to a graph H.

Dyer and Greenhill [9] initiated the algorithmic study of *counting* homomorphisms. They gave a complete characterisation of the graph homomorphism counting problem, parameterised by a fixed graph H: "Given an input graph G, determine how many homomorphisms there are from G to H." Dyer and Greenhill showed that this problem can be solved in polynomial time if every component of H is a clique with all loops present or a biclique (complete bipartite graph) with no loops present. Otherwise, the counting problem is $\#$P-complete. Díaz, Serna and Thilikos [8] and Hell and Nešetřil [21] have shown that the same dichotomy characterisation holds for the problem of counting list homomorphisms.

The main contribution of this paper is to give complete dichotomy characterisations for the problems of counting compactions and surjective homomorphisms. Our main theorem, Theorem 2, shows that the characterisation for compactions is different from the characterisation for counting homomorphisms. If every component of H is (i) a star with no loops present, (ii) a single vertex with a loop, or (iii) a single edge with two loops then counting compactions to H is solvable in polynomial time. Otherwise, it is $\#$P-complete. We also obtain the same dichotomy for the problem of counting list compactions. Thus, even though the decision problem is still open for compactions, our theorem gives a complete classification of the complexity of the corresponding counting problem.

There is evidence that computational problems involving surjective homomorphisms are more difficult than those involving (unrestricted) homomorphisms. For example, suppose that H consists of a 3-vertex clique with no loops together with a single looped vertex. As [1] noted, the problem of deciding whether there is a homomorphism from an input graph G to H is trivial (the answer is yes, since all vertices of G may be mapped to the loop) but the problem of determining whether there is a surjective homomorphism from an input graph G to H is NP-complete. (To see this, recall the NP-hard problem of determining whether a connected graph G' that is not bipartite is 3-colourable. Given such a graph G', we may determine

whether it is 3-colourable by letting G consist of the disjoint union of G' and a loop-free clique of size 4, and then checking whether there is a surjective homomorphism from G to H.) There is also evidence that *counting* problems involving surjective homomorphisms are more difficult than those involving unrestricted homomorphisms. In the full version we consider a *uniform* homomorphism-counting problem where all connected components of G are cliques without loops and all connected components of H are cliques with loops, but both G and H are part of the input. It turns out (Theorem 31 in the full version) that in this uniform case, counting homomorphisms is in FP but counting surjective homomorphisms is $\#\mathrm{P}$-complete. Despite this evidence, we show (Theorem 3) that the problem of counting surjective homomorphisms to a fixed graph H has the same complexity characterisation as the problem of counting all homomorphisms to H: The problem is solvable in polynomial time if every component of H is a clique with loops or a biclique without loops. Otherwise, it is $\#\mathrm{P}$-complete. Once again, our dichotomy characterisation extends to the problem of counting surjective list homomorphisms. Even though the decision problem is still open for surjective homomorphisms, our theorem gives a complete complexity classification of the corresponding counting problem.

In Section 1.2 we will introduce one more related counting problem — the problem of counting *retractions*. Informally, if G is a graph containing an induced copy of H then a retraction from G to H is a homomorphism from G to H that maps the induced copy to itself. Retractions are well-studied in combinatorics, often from an algorithmic perspective [1, 10, 11, 12, 28, 30]. A complexity classification is not known for the decision problem (determining whether there is a retraction from an input to H). Nevertheless, it is easy to give a complexity characterisation for the corresponding counting problem (Corollary 7). This characterisation, together with our main results, implies that a long-standing conjecture of Winkler about the complexity of the decision problems for compactions and retractions is false in the counting setting. See Section 1.2 for details.

1.1 Notation and Theorem Statements

In this paper graphs are undirected and may contain loops. A homomorphism from a graph G to a graph H is a function $h\colon V(G) \to V(H)$ such that, for all $\{u, v\} \in E(G)$, the image $\{h(u), h(v)\}$ is in $E(H)$. We use $N\big(G \to H\big)$ to denote the number of homomorphisms from G to H. A homomorphism h is said to "use" a vertex $v \in V(H)$ if there is a vertex $u \in V(G)$ such that $h(u) = v$. It is *surjective* if it uses every vertex of H. We use $N^{\text{sur}}\big(G \to H\big)$ to denote the number of surjective homomorphisms from G to H. A homomorphism h is said to use an edge $\{v_1, v_2\} \in E(H)$ if there is is an edge $\{u_1, u_2\} \in E(G)$ such that $h(u_1) = v_1$ and $h(u_2) = v_2$. It is a *compaction* if it uses every vertex of H and every non-loop edge of H. We use $N^{\text{comp}}\big(G \to H\big)$ to denote the number of compactions from G to H. H is said to be *reflexive* if every vertex has a loop. It is said to be *irreflexive* if no vertex has a loop. We study the following computational problems[1], which are parameterised by a graph H.

Name: #Hom(H).	**Name:** #Comp(H).	**Name:** #SHom(H).
Input: Irreflexive graph G.	**Input:** Irreflexive graph G.	**Input:** Irreflexive graph G.
Output: $N\big(G \to H\big)$.	**Output:** $N^{\text{comp}}\big(G \to H\big)$.	**Output:** $N^{\text{sur}}\big(G \to H\big)$.

A *list homomorphism* generalises a homomorphism in the same way that a list colouring of a graph generalises a (proper) colouring. Suppose that G is an irreflexive graph and that H is a graph. Consider a collection of sets $\mathbf{S} = \{S_v \subseteq V(H) : v \in V(G)\}$ A *list homomorphism* from $(G, \mathbf{S})$ to H is a homomorphism h from G to H such that, for every vertex v of G, $h(v) \in S_v$. The set S_v is referred to as a "list",

[1]The reason that the input graph G is restricted to be irreflexive in these problems, but that H is not restricted, is that this is the convention in the literature. Since our results will be complexity classifications, parameterised by H, we strengthen the results by avoiding restrictions on H. Different conventions are possible regarding G, but hardness results are typically the most difficult part of the complexity classifications in this area, so restricting G leads to technically-stronger results.

specifying the allowable targets of vertex v. We use $N\big((G,\mathbf{S}) \to H\big)$ to denote the number of list homomorphisms from $(G,\mathbf{S})$ to H, $N^{\text{sur}}\big((G,\mathbf{S}) \to H\big)$ to denote the number of surjective list homomorphisms from $(G,\mathbf{S})$ to H and $N^{\text{comp}}\big((G,\mathbf{S}) \to H\big)$ to denote the number of list homomorphisms from $(G,\mathbf{S})$ to H that are compactions. We study the following additional computational problems, again parameterised by a graph H.

Name: #LHom(H).	**Name:** #LComp(H).	**Name:** #LSHom(H).
Input: Irreflexive graph G and a collection of lists $\mathbf{S} = \{S_v \subseteq V(H) : v \in V(G)\}$.	**Input:** Irreflexive graph G and a collection of lists $\mathbf{S} = \{S_v \subseteq V(H) : v \in V(G)\}$.	**Input:** Irreflexive graph G and a collection of lists $\mathbf{S} = \{S_v \subseteq V(H) : v \in V(G)\}$.
Output: $N\big((G,\mathbf{S}) \to H\big)$.	**Output:** $N^{\text{comp}}\big((G,\mathbf{S}) \to H\big)$.	**Output:** $N^{\text{sur}}\big((G,\mathbf{S}) \to H\big)$.

In order to state our theorems, we define some classes of graphs. A graph H is a *clique* if, for every pair (u,v) of distinct vertices, $E(H)$ contains the edge $\{u,v\}$. (Like other graphs, cliques may contain loops but not all loops need to be present.) H is a *biclique* if it is bipartite and there is a partition of $V(H)$ into two disjoint sets U and V such that, for every $u \in U$ and $v \in V$, $E(H)$ contains the edge $\{u,v\}$. We sometimes use the notation $K_{a,b}$ to denote a biclique whose vertices can be partitioned into U and V with $|U| = a$ and $|V| = b$. A biclique is a *star* if $|U| = 1$ or $|V| = 1$ (or both). Note that a star may have only one vertex since, for example, we could have $|U| = 1$ and $|V| = 0$. The size of a graph is the number of vertices that it has. We can now state the theorem of Dyer and Greenhill [9], as extended to list homomorphisms by Díaz, Serna and Thilikoas [8] and Hell and Nešetřil [21].

Theorem 1 (Dyer, Greenhill)**.** *Let H be a graph. If every connected component of H is a reflexive clique or an irreflexive biclique, then #Hom(H) and #LHom(H) are in* FP*. Otherwise, #Hom(H) and #LHom(H) are* #P*-complete.*

We can also state the main results of this paper.

Theorem 2. *Let H be a graph. If every connected component of H is an irreflexive star or a reflexive clique of size at most* 2 *then #Comp(H) and #LComp(H) are in* FP*. Otherwise, #Comp(H) and #LComp(H) are* #P*-complete.*

Theorem 3. *Let H be a graph. If every connected component of H is a reflexive clique or an irreflexive biclique, then #SHom(H) and #LSHom(H) are in* FP*. Otherwise, #SHom(H) and #LSHom(H) are* #P*-complete.*

The proofs of Theorems 2 and 3 are in the full version. The proof of Theorem 3 is not difficult. It is based on two reductions established via polynomial interpolation: #LSHom$(H) \leq$ #LHom(H) (Theorem 28 in the full version) and #Hom$(H) \leq$ #SHom(H) (Theorem 30 in the full version). The proof of Theorem 2 is significantly more complicated. We give the key technical and conceptual ideas of the proof in Section 3 and refer the reader to the full version for the omitted details. In the next subsection, we will discuss complexity consequences of Theorems 1, 2 and 3 to the problem of counting retractions.

1.2 Reductions and Retractions

In the context of two computational problems P_1 and P_2, we write $\mathrm{P}_1 \leq \mathrm{P}_2$ if there exists a polynomial-time Turing reduction from P_1 to P_2. If there exist such reductions in both directions, we write $\mathrm{P}_1 \equiv \mathrm{P}_2$. Theorems 1, 2 and 3 imply the following observation.

Observation 4. *Let H be a graph. Then*

$$\textit{\#Hom}(H) \equiv \textit{\#LHom}(H) \equiv \textit{\#SHom}(H) \equiv \textit{\#LSHom}(H) \leq \textit{\#Comp}(H) \equiv \textit{\#LComp}(H).$$

In order to see how Observation 4 contrasts with the situation concerning decision problems, it is useful to define decision versions of the computational problems that we study. Thus, Hom(H) is the problem of determining whether $N\big(G \to H\big) = 0$, given an input G of #Hom(H). The decision problems Comp(H), SHom(H) and LHom(H) are defined similarly.

It is also useful to define the notion of a *retraction*. Suppose that H is a graph with $V(H) = \{v_1, \ldots, v_c\}$ and that G is an irreflexive graph. We say that a tuple $(u_1, \ldots, u_c)$ of c distinct vertices of G induces a copy of H if, for every $1 \leq a < b \leq c$, $\{u_a, u_b\} \in E(G) \iff \{v_a, v_b\} \in E(H)$. A *retraction* from $(G; u_1, \ldots, u_c)$ to H is a homomorphism h from G to H such that, for all $i \in [c]$, $h(u_i) = v_i$. We use $N^{\text{ret}}\big((G; u_1, \ldots, u_c) \to H\big)$ to denote the number of retractions from $(G; u_1, \ldots, u_c)$ to H. We briefly consider the retraction counting and decision problems, which are parameterised by a graph H with $V(H) = \{v_1, \ldots, v_c\}$.[2]

Name: #Ret(H).
Input: Irreflexive graph G and a tuple $(u_1, \ldots, u_c)$ of distinct vertices of G that induces a copy of H.
Output: $N^{\text{ret}}\big((G; u_1, \ldots, u_c) \to H\big)$.

Name: Ret(H).
Input: Irreflexive graph G and a tuple $(u_1, \ldots, u_c)$ of distinct vertices of G that induces a copy of H.
Output: Does $N^{\text{ret}}\big((G; u_1, \ldots, u_c) \to H\big) = 0$?

The following observation appears as Proposition 1 of [1]. The proposition is stated for more general structures than graphs, but it applies equally to our setting.

Proposition 5. *(Bodirsky et al.) Let H be a graph. Then*

$$\textit{Hom}(H) \leq \textit{SHom}(H) \leq \textit{Comp}(H) \leq \textit{Ret}(H) \leq \textit{LHom}(H).$$

We have already mentioned the fact (pointed out by Bodirsky et al.) that if H is an irreflexive 3-vertex clique together with a single looped vertex, then Hom(H) is in P, but SHom(H) is NP-complete. There are no known graphs H separating SHom(H), Comp(H) and Ret(H). Moreover, Bodirsky et al. mention a conjecture [1, Conjecture 2], attributed to Peter Winkler, that, for all graphs H, Comp(H) and Ret(H) are polynomially Turing equivalent.

The following observation, together with our theorems, implies Corollary 8 (below), which shows that the generalisation of Winkler's conjecture to the counting setting is false unless $\text{FP} = \#\text{P}$, since #Comp(H) and #Ret(H) are not polynomially Turing equivalent for all H.

Observation 6. *Let H be a graph. Then #Ret(H) $\leq$ #LHom(H) and #Hom(H) $\leq$ #Ret(H)*

Observation 6 immediately implies the following dichotomy characterisation for the problem of counting retractions.

Corollary 7. *Let H be a graph. If every connected component of H is a reflexive clique or an irreflexive biclique, then #Ret(H) is in* FP*. Otherwise, #Ret(H) is $\#\text{P}$-complete.*

Corollary 8. *Let H be a graph. Then*

$$\textit{\#Hom}(H) \equiv \textit{\#LHom}(H) \equiv \textit{\#SHom}(H) \equiv \textit{\#LSHom}(H) \equiv \textit{\#Ret}(H) \leq \textit{\#Comp}(H) \equiv \textit{\#LComp}(H).$$

Furthermore, there is a graph H for which #Comp(H) and #LComp(H) are $\#\text{P}$-complete, but #Hom(H), #LHom(H), #SHom(H), #LSHom(H) and #Ret(H) are in FP*.*

[2] Once again, some works would allow G to have loops, and would insist that loops are preserved in the induced copy of H. We prefer to stick with the convention that G is irreflexive, but this does not make a difference to the complexity classifications that we describe.

1.3 Related Work

This section was added after the paper was written, in order to draw attention to some interesting subsequent work [7, 5].

Both our tractability results and our hardness results rely on the fact (see Theorem 17) that the number of compactions from G to H can be expressed as a linear combination of the number of homomorphisms from G to certain subgraphs J of H. A similar statement applies to surjective homomorphisms.

As we note in the paper, these kinds of linear combinations have been noticed in related contexts before, for example in [3, Lemma 4.2] and in [24]. We use the linear combination of Theorem 17, together with interpolation, to prove hardness. Although it is standard to restrict the input graph G to be irreflexive (and this restriction makes the results stronger) the fact that G is required to be irreflexive causes severe difficulties.

In fact, Dell's note about our paper [7] shows that, if you weaken the theorem statements by allowing the input G to have loops, then a simpler interpolation based on a very recent paper by Curticapean, Dell and Marx [6] can be used to make the proofs very elegant! The exact same idea, written more generally, was also discovered by Chen [5].

2 Preliminaries

It will often be technically convenient to restrict the problems that we study by requiring the input graph G to be connected. In each case, we do this by adding a superscript "C" to the name of the problem. For example, the problem #Hom$^{\mathrm{C}}(H)$ is defined as follows.

Name. #Hom$^{\mathrm{C}}(H)$.

Input. A *connected* irreflexive graph G.

Output. $N\big(G \to H\big)$.

It is well known and easy to see (See, e.g., [24, (5.28)]) that if G is an irreflexive graph with components $G_1, \ldots, G_t$ then $N\big(G \to H\big) = \prod_{i\in[t]} N\big(G_i \to H\big)$. Thus, Dyer and Greenhill's theorem (Theorem 1) can be re-stated in the following convenient form.

Theorem 9 (Dyer, Greenhill)**.** *Let H be a graph. If every connected component of H is a reflexive clique or an irreflexive biclique, then #Hom$^{C}(H)$, #Hom(H), #LHom$^{C}(H)$ and #LHom(H) are all in* FP*. Otherwise, #Hom$^{C}(H)$, #Hom(H), #LHom$^{C}(H)$ and #LHom(H) are all* #P*-complete.*

Finally, we introduce some frequently used notation. For every positive integer n, we define $[n] = \{1, \ldots, n\}$. A subgraph H' of H is said to be *loop-hereditary* with respect to H if for every $v \in V(H')$ that is contained in a loop in $E(H)$, v is also contained in a loop in $E(H')$. We indicate that two graphs G_1 and G_2 are isomorphic by writing $G_1 \cong G_2$. Given sets S_1 and S_2, we write $S_1 \oplus S_2$ for the disjoint union of S_1 and S_2. Given graphs G_1 and G_2, we write $G_1 \oplus G_2$ for the graph $(V(G_1) \oplus V(G_2), E(G_1) \oplus E(G_2))$.

3 Counting Compactions

The tractability result in Theorem 2 follows from the fact (see Theorem 17) that the number of compactions from G to H can be expressed as a linear combination of the number of homomorphisms from G to certain subgraphs J of H. A shorter explicit tractability result is also given in the full version. Here we concentrate on the intractability results. We consider a graph H that has a connected component that is not an irreflexive star or a reflexive clique of size at most 2. The objective is to show that #Comp(H) and #LComp(H) are #P-hard (this is the hardness content of Theorem 2).

We start with a brief proof sketch. The easy case is when H contains a component that is not a reflexive clique or an irreflexive biclique. In this case, Dyer and Greenhill's Theorem 1 shows that #Hom(H) is $\#$P-hard. We obtain the desired hardness by giving (in Theorem 13 in the full version) a polynomial-time Turing reduction from #Hom(H) to #Comp(H). The result is finished off with a trivial reduction from #Comp(H) to #LComp(H). The proof of Theorem 13 is not difficult — given an input G to #Hom(H), we add isolated vertices and edges to G and recover the desired quantity $N\big(G \to H\big)$ using an oracle for #Comp(H) and polynomial interpolation. There are small technical issues related to size-1 components in H, and these are dealt with in Lemma 11 in the full version.

The more interesting case is when every component of H is a reflexive clique or an irreflexive biclique, but some component is either a reflexive clique of size at least 3 or an irreflexive biclique that is not a star. The first milestone is Lemma 23, which shows $\#$P-hardness in the special case where H is connected. We prove Lemma 23 in a slightly stronger setting where the input graph G is connected. This allows us, in the remainder of the section, to generalise the connected case to the case in which H is not connected.

The main difficulty, then, is Lemma 23. The goal is to show that #Comp(H) is $\#$P-hard when H is a reflexive clique of size at least 3 or an irreflexive biclique that is not a star. Our main method for solving this problem is a technique (Theorem 17) that lets us compute the number of compactions from a connected graph G to a connected graph H using a weighted sum of homomorphism counts, say $N\big(G \to J_1\big), \ldots, N\big(G \to J_k\big)$. An important feature is that some of the weights might be negative.

Our basic approach will be to find a constituent J_i such that #Hom$^{\mathrm{C}}(J_i)$ is $\#$P-hard and to reduce #Hom$^{\mathrm{C}}(J_i)$ to the problem of computing the weighted sum. Of course, if computing $N\big(G \to J_1\big)$ is $\#$P-hard and computing $N\big(G \to J_2\big)$ is $\#$P-hard, it does not follow that computing a weighted sum of these is $\#$P-hard.

In order to solve this problem, in Lemmas 19 and 20 we use an argument similar to that of Lovász [23, Theorem 3.6] to prove the existence of input instances that help us to distinguish between the problems #Hom$^{\mathrm{C}}(J_1), \ldots,$ #Hom$^{\mathrm{C}}(J_k)$. Theorem 21 then provides the desired reduction from a chosen #Hom$^{\mathrm{C}}(J_i)$ to the problem of computing the weighted sum. Theorem 21 is proved by a more complicated interpolation construction, in which we use the instances from Lemma 20 to modify the input.

Having sketched the proof at a high level, we now give some more details of the difficult case. Towards this end, we introduce some definitions which we will use repeatedly in the remainder of this section.

Definition 14. A *weighted graph set* is a tuple $(\mathcal{H}, \lambda)$, where $\mathcal{H}$ is a set of *non-empty, pairwise non-isomorphic, connected* graphs and λ is a function $\lambda\colon \mathcal{H} \to \mathbb{Z}$.

Definition 15. Let H be a connected graph. By $Sub(H)$ we denote the set of non-empty, loop-hereditary, connected subgraphs of H. Let $\mathcal{S}_H$ be a set which contains exactly one representative of each isomorphism class of the graphs in $Sub(H)$. Finally, for $H' \in \mathcal{S}_H$, we define $\mu_H(H')$ to be the number of graphs in $Sub(H)$ that are isomorphic to H'.

Note that for a connected graph H, we have $\mu_H(H) = 1$.

Definition 16. For each non-empty connected graph H, we define a weight function λ_H which assigns an integer weight to each non-empty connected graph J.

- If J is not isomorphic to any graph in $\mathcal{S}_H$, then $\lambda_H(J) = 0$.
- If $J \cong H$, then $\lambda_H(J) = 1$.
- Finally, if J is isomorphic to some graph in $\mathcal{S}_H$ but $J \ncong H$, we define $\lambda_H(J)$ inductively as follows.

$$\lambda_H(J) = -\sum_{\substack{H' \in \mathcal{S}_H \\ \text{s.t. } H' \ncong H}} \mu_H(H')\lambda_{H'}(J).$$

Note that λ_H is well-defined as all graphs $H' \in \mathcal{S}_H$ with $H' \not\cong H$ are smaller than H either in the sense of having fewer vertices or in the sense of having the same number of vertices but fewer edges.

The following theorem is the key to our approach for computing the number of compactions from a connected graph G to a connected graph H using a weighted sum of homomorphism counts.

Theorem 17. *Let H be a non-empty connected graph. Then for every non-empty, irreflexive and connected graph G we have $N^{comp}\big(G \to H\big) = \sum_{J \in \mathcal{S}_H} \lambda_H(J) N\big(G \to J\big)$.*

Proof. Let $H_1, H_2, \dots$ be the set of non-empty connected graphs sorted by some fixed ordering that ensures that if H_i is isomorphic to a subgraph of H_j, then $i \leq j$. We verify the statement of the theorem by induction over the graph index with respect to this ordering. Let G be non-empty, irreflexive and connected.

For the base case, H_1 is K_1, which is the graph with one vertex and no edges. In this case, $\mathcal{S}_{H_1} = \{K_1\}$ and $\lambda_{K_1}(K_1) = 1$. Also, $N^{\text{comp}}\big(G \to K_1\big) = N\big(G \to K_1\big)$. So the theorem holds in this case.

Now assume that the statement holds for all graphs up to index i and consider the graph H_{i+1}. For ease of notation we set $H = H_{i+1}$. We use the fact that every homomorphism from a connected graph G to H_{i+1} is a compaction onto some non-empty, loop-hereditary and connected subgraph of H_{i+1} and vice versa. Thus, it holds that

$$\begin{aligned} N\big(G \to H\big) &= \sum_{H' \in \mathcal{S}_H} \mu_H(H') \cdot N^{\text{comp}}\big(G \to H'\big) \\ &= N^{\text{comp}}\big(G \to H\big) + \sum_{\substack{H' \in \mathcal{S}_H \\ \text{s.t. } H' \not\cong H}} \mu_H(H') \cdot N^{\text{comp}}\big(G \to H'\big). \end{aligned}$$

Thus, we can rearrange and use the induction hypothesis to obtain

$$\begin{aligned} N^{\text{comp}}\big(G \to H\big) &= N\big(G \to H\big) - \sum_{\substack{H' \in \mathcal{S}_H \\ \text{s.t. } H' \not\cong H}} \mu_H(H') \cdot N^{\text{comp}}\big(G \to H'\big) \\ &= N\big(G \to H\big) - \sum_{\substack{H' \in \mathcal{S}_H \\ \text{s.t. } H' \not\cong H}} \mu_H(H') \cdot \sum_{J \in \mathcal{S}_{H'}} \lambda_{H'}(J) N\big(G \to J\big). \end{aligned}$$

Then we change the order of summation and use that $\lambda_{H'}(J) = 0$ if J is not isomorphic to any graph in $\mathcal{S}_{H'}$ to collect all coefficients that belong to a particular term $N\big(G \to J\big)$. We obtain

$$\begin{aligned} N^{\text{comp}}\big(G \to H\big) &= N\big(G \to H\big) - \sum_{\substack{J \in \mathcal{S}_H \\ \text{s.t. } J \not\cong H}} \Big(\sum_{\substack{H' \in \mathcal{S}_H \\ \text{s.t. } H' \not\cong H}} \mu_H(H') \lambda_{H'}(J) \Big) N\big(G \to J\big) \\ &= \sum_{J \in \mathcal{S}_H} \lambda_H(J) N\big(G \to J\big). \end{aligned}$$

□

We remark that Theorem 17 can be generalised to graphs H and G with multiple connected components by looking at all subgraphs of H, rather than just at the connected ones. However, within this work, the version for connected graphs suffices.

Let $(\mathcal{H}, \lambda)$ be a weighted graph set. The following parameterised problem is not natural in its own right, but it helps us to analyse the complexity of $\#\text{Comp}^{\text{C}}(H)$:

Name. #GraphSetHom$^{\mathrm{C}}((\mathcal{H}, \lambda))$.

Input. An irreflexive, connected graph G.

Output. $Z_{\mathcal{H},\lambda}(G) = \begin{cases} 0 & \text{if } G \text{ is empty} \\ \sum_{J \in \mathcal{H}} \lambda(J) N\big(G \to J\big) & \text{otherwise.} \end{cases}$

Corollary 18. *Let H be a non-empty connected graph. Then*

$$\#\mathit{Comp}^{C}(H) \equiv \#\mathit{GraphSetHom}^{C}((\mathcal{S}_H, \lambda_H)).$$

Corollary 18 gives us the desired connection between weighted graph sets and compactions. We will use this in the proof of Lemma 23 to establish the #P-hardness of #Comp$^{\mathrm{C}}(H)$ when H is either a reflexive clique of size at least 3 or an irreflexive biclique that is not a star.

Our next goal is to prove Theorem 21, which states that, for certain weighted graph sets $(\mathcal{H}, \lambda)$, determining $Z_{\mathcal{H},\lambda}(G)$ is at least as hard as computing $N\big(G \to J\big)$ for some graph J from the set $\mathcal{H}$ with $\lambda(J) \neq 0$. To this end, we first introduce two lemmas that help us to distinguish between different graphs J in the interpolation that we use in the full version to prove Theorem 21.

For the following lemmas, we introduce some new notation. For a graph G with distinguished vertex $v \in V(G)$ and a graph H with distinguished vertex $w \in V(H)$, the quantity $N\big((G, v) \to (H, w)\big)$ denotes the number of homomorphisms h from G to H with $h(v) = w$. If there exists an isomorphism from G to H that maps v onto w, we write $(G, v) \cong (H, w)$, otherwise we write $(G, v) \ncong (H, w)$. In the following lemma, we show that for two such target entities (H_1, w_1) and (H_2, w_2) that are non-isomorphic, there exists an input which separates them. To this end, we use an argument very similar to that presented in [14, Lemma 3.6] and in the textbook by Hell and Nešetřil [22, Theorem 2.11], which goes back to the works of Lovász [23, Theorem 3.6].

Lemma 19. *Let H_1 and H_2 be connected graphs with distinguished vertices $w_1 \in V(H_1)$ and $w_2 \in V(H_2)$ such that $(H_1, w_1) \ncong (H_2, w_2)$. Suppose that one of the following cases holds:*

Case 1. H_1 and H_2 are reflexive graphs.

Case 2. H_1 and H_2 are irreflexive bipartite graphs, each of which contains at least one edge.

Then

i) There exists a connected irreflexive graph G with distinguished vertex $v \in V(G)$ for which $N\big((G, v) \to (H_1, w_1)\big) \neq N\big((G, v) \to (H_2, w_2)\big)$.

ii) In Case 2 we can assume that G contains at least one edge and is bipartite.

In the following lemma, we generalise the pairwise property from Lemma 19. The result and the proof are adapted versions of [13, Lemma 6]. For ease of notation let $\binom{[k]}{2}$ denote the set of all pairs $\{i, j\}$ with $i, j \in [k]$ and $i \neq j$. The proof is in the full version.

Lemma 20. *Let $H_1, \ldots, H_k$ be connected graphs with distinguished vertices $w_1, \ldots, w_k$ where $w_i \in V(H_i)$ for all $i \in [k]$ and, for every pair $\{i, j\} \in \binom{[k]}{2}$, we have $(H_i, w_i) \ncong (H_j, w_j)$. Suppose that one of the following cases holds:*

Case 1. $\forall i \in [k]$, H_i is a reflexive graph.

Case 2. $\forall i \in [k]$, H_i is an irreflexive bipartite graph that contains at least one edge.

Then

i) There exists a connected irreflexive graph G with a distinguished vertex $v \in V(G)$ such that, for every $\{i,j\} \in \binom{[k]}{2}$, it holds that $N\big((G,v) \to (H_i,w_i)\big) \neq N\big((G,v) \to (H_j,w_j)\big)$.

ii) In Case 2 we can assume that G contains at least one edge and is bipartite.

In the following theorem, we use the separating instances that we obtain from Lemma 20 for interpolation-based reductions to #GraphSetHom$^{\mathrm{C}}((\mathcal{H},\lambda))$. The proof is in the full version.

Theorem 21. *Let $(\mathcal{H},\lambda)$ be a weighted graph set for which one of two cases holds:*

Case 1. All graphs in $\mathcal{H}$ are reflexive.

Case 2. All graphs in $\mathcal{H}$ are irreflexive and bipartite.

Then, for all $H \in \mathcal{H}$ with $\lambda(H) \neq 0$, it holds that #Hom$^{C}(H) \leq$ #GraphSetHom$^{C}((\mathcal{H},\lambda))$.

Next, we give a short technical lemma which follows from Definition 16 and is used in Lemma 23 to show that Theorem 21 gives hardness results for #Comp$^{\mathrm{C}}(H)$. The proof is in the full version.

Lemma 22. *Let H be a connected graph with at least one non-loop edge. Let H^- be the graph obtained from H by deleting exactly one non-loop edge (but keeping all vertices). If H^- is connected, then $\lambda_H(H^-) \neq 0$.*

We now have most of the tools at hand to classify the complexity of #Comp(H). Tractability results come from Lemma 10 in the full version. If H has a component that is not a reflexive clique or an irreflexive biclique then hardness will be lifted from Dyer and Greenhill's Theorem 1 via Theorem 13 from the full version. The most difficult case is when all components of H are reflexive cliques or irreflexive bicliques, but some component is not a star or a reflexive biclique of size at most 2. If H is connected then hardness will come from the following key lemma.

Lemma 23. *If H is a reflexive clique of size at least 3 then #Comp$^{C}(H)$ is $\#\mathrm{P}$-hard. If H is an irreflexive biclique that is not a star then #Comp$^{C}(H)$ is $\#\mathrm{P}$-hard.*

Proof. Suppose that H is a reflexive clique of size at least 3 or an irreflexive biclique that is not a star. Note that $(\mathcal{S}_H,\lambda_H)$ is a weighted graph set. Let H^- be a graph obtained from H by deleting a non-loop edge. Note that H^- is connected and it is not a reflexive clique or an irreflexive biclique. Thus we can apply Lemma 22 to show that $\lambda(H^-) > 0$ and Theorem 9 to show that #Hom$^{\mathrm{C}}(H^-)$ is $\#\mathrm{P}$-complete. We will complete the proof of the Lemma by showing #Hom$^{\mathrm{C}}(H^-) \leq$ #Comp$^{\mathrm{C}}(H)$.

If H is a reflexive graph then the definition of $\mathcal{S}_H$ ensures that all graphs in $\mathcal{S}_H$ are reflexive. If H is an irreflexive bipartite graph, then the definition ensures that all graphs in $\mathcal{S}_H$ are irreflexive and bipartite. We can apply Theorem 21 to the weighted graph set $(\mathcal{S}_H,\lambda_H)$ with $H^- \in \mathcal{S}_H$ to obtain #Hom$^{\mathrm{C}}(H^-) \leq$ #GraphSetHom$^{\mathrm{C}}((\mathcal{S}_H,\lambda_H))$. By Corollary 18, #GraphSetHom$^{\mathrm{C}}((\mathcal{S}_H,\lambda_H)) \equiv$ #Comp$^{\mathrm{C}}(H)$. The lemma follows. □

The remainder of the section generalises the connected case to the case in which H is not connected. We use the following two definitions in Lemmas 26 and 27 and in the proof of Theorem 2.

Definition 24. Let H be a graph. Suppose that every non-star connected component of H has at most j vertices and that some non-star component of H has j vertices. Let $\mathcal{A}(H)$ be the set of reflexive components of H with j vertices and let $\mathcal{B}(H)$ be the set of irreflexive non-star components of H with j vertices.

Definition 25. Let $L(H)$ denote the set of loops of a graph H. We define the graph $H^0 = (V(H), E(H) \setminus L(H))$.

Lemma 26. *Let H be a graph in which every component is a reflexive clique or an irreflexive biclique. If $J \in \mathcal{A}(H)$ then #Comp$^{\mathsf{C}}(J) \leq$ #Comp(H).*

Proof. Let H be a graph in which every component is a reflexive clique or an irreflexive biclique. Let $\mathcal{A}(H) = \{A_1, \ldots, A_k\}$. It follows from the definition of $\mathcal{A}(H)$ that all elements of $\mathcal{A}(H)$ are reflexive cliques of some size j (the same j for all graphs in $\mathcal{A}(H)$).

If $j \leq 2$, the statement of the lemma is trivially true, since Lemma 10 in the full version shows that #Comp(A_i) is in FP, so the restricted problem #Comp$^{\mathsf{C}}(A_i)$ is also in FP.

Now assume $j \geq 3$. Suppose without loss of generality that $J = A_1$. Let G be a (connected) input to #Comp$^{\mathsf{C}}(J)$. For all $i \in [k]$, let $H \setminus A_i$ be the graph constructed from H by deleting the connected component A_i. Using Definition 25 we define the (irreflexive) graph $G' = (H \setminus J \oplus G)^0$ as an input to #Comp(H). Intuitively, to form G' from H we replace the connected component J with the graph G, then we delete all loops. We will prove the following claim.

Claim: Let $h\colon V(G') \to V(H)$ be a compaction from G' to H. Then the restriction $h|_{V(G)}$ is a compaction from G onto an element of $\mathcal{A}(H)$.

Proof of the claim: As h is a homomorphism, it maps each connected component of G' to a connected component of H. As, furthermore, h is a compaction and G' and H have the same number of connected components, it follows that there exist connected components $C_1, \ldots, C_k$ of G' such that for $i \in [k]$, $h|_{V(C_i)}$ is a compaction from C_i onto A_i. To prove the claim, we show that G is an element of $\mathcal{C} = \{C_1, \ldots, C_k\}$. In order to use all vertices of a graph in $\mathcal{A}(H)$, i.e. a reflexive size-j clique, a graph in $\mathcal{C}$ has to have at least j vertices itself. Therefore and by the construction of G', an element of $\mathcal{C}$ can only be one of the following: a clique with j vertices, a biclique with j vertices, a star with at least j vertices or the copy of G. Since $j \geq 3$, it is easy to see that there is no compaction from a star onto a clique with j vertices. In order to compact onto a reflexive clique of size j, an element of $\mathcal{C}$ also has to have at least $j(j-1)/2$ edges. Thus, $\mathcal{C}$ does not contain any bicliques. Finally, there are only $k-1$ connected components in G' that are j-vertex cliques other than (possibly) G. Therefore, G has to be an element of $\mathcal{C}$, which proves the claim.

Using the notation from Definition 25, the claim implies

$$N^{\text{comp}}\big(G' \to H\big) = \sum_{i=1}^{k} N^{\text{comp}}\big(G \to A_i\big) \cdot N^{\text{comp}}\big((H \setminus A_i)^0 \to H \setminus A_i\big). \tag{1}$$

We can simplify the expression (1) using the fact that all elements of $\mathcal{A}(H)$ are reflexive size-j cliques:

$$N^{\text{comp}}\big(G' \to H\big) = k \cdot N^{\text{comp}}\big(G \to J\big) \cdot N^{\text{comp}}\big((H \setminus J)^0 \to H \setminus J\big).$$

As $N^{\text{comp}}\big((H \setminus J)^0 \to H \setminus J\big)$ does not depend on G, it can be computed in constant time. Thus, using a single #Comp(H) oracle call we can compute $N^{\text{comp}}\big(G \to J\big)$ in polynomial time as required. □

The proof of the following lemma is similar to that of Lemma 26 and can be found in the full version.

Lemma 27. *Let H be a graph in which every component is a reflexive clique or an irreflexive biclique. If $\mathcal{A}(H)$ is empty but $\mathcal{B}(H)$ is non-empty, then there exists a component $J \in \mathcal{B}(H)$ such that #Comp$^{\mathsf{C}}(J) \leq$ #Comp(H).*

Finally, we prove the main theorem of this paper, which we restate at this point.

Theorem 2. *Let H be a graph. If every connected component of H is an irreflexive star or a reflexive clique of size at most 2 then #Comp(H) and #LComp(H) are in* FP*. Otherwise, #Comp(H) and #LComp(H) are* #P*-complete.*

Proof. The membership of #Comp(H) in $\#\mathrm{P}$ is straightforward. We distinguish between a number of cases depending on the graph H.

Case 1: Suppose that every connected component of H is an irreflexive star or a reflexive clique of size at most 2. Then #LComp(H) is in FP by Lemma 10 in the full version.

Case 2: Suppose that H contains a component that is not a reflexive clique or an irreflexive biclique. Then the hardness of #Hom(H) (from Theorem 1) together with the reduction #Hom(H) $\leq$ #Comp(H) (from Theorem 13 in the full version) implies that #Comp(H) is $\#\mathrm{P}$-hard. The hardness of #LComp(H) follows from the trivial reduction from #Comp(H) to #LComp(H).

Case 3: Suppose that the components of H are reflexive cliques or irreflexive bicliques and that H contains at least one component that is not a star or a reflexive clique of size at most 2. Every graph $J \in \mathcal{A}(H) \cup \mathcal{B}(H)$ is a reflexive clique of size at least 3 or an irreflexive biclique that is not a star. By Lemma 23, #Comp$^{\mathrm{C}}(J)$ is $\#\mathrm{P}$-complete. Finally, as $\mathcal{A}(H) \cup \mathcal{B}(H)$ is non-empty, we can use either Lemma 26 or Lemma 27 to obtain the existence of $J \in \mathcal{A}(H) \cup \mathcal{B}(H)$ with #Comp$^{\mathrm{C}}(J) \leq$ #Comp(H). This implies that #Comp(H) is #P-hard. As in Case 2, the hardness of #LComp(H) follows from the trivial reduction from #Comp(H) to #LComp(H). □

References

[1] Manuel Bodirsky, Jan Kára, and Barnaby Martin. The complexity of surjective homomorphism problems – a survey. *Discrete Applied Mathematics*, 160(12):1680 – 1690, 2012.

[2] Christian Borgs, Jennifer Chayes, László Lovász, Vera T. Sós, and Katalin Vesztergombi. Counting graph homomorphisms. In *Topics in discrete mathematics*, volume 26 of *Algorithms Combin.*, pages 315–371. Springer, Berlin, 2006.

[3] Christian Borgs, Jennifer T. Chayes, Jeff Kahn, and László Lovász. Left and right convergence of graphs with bounded degree. *Random Struct. Algorithms*, 42(1):1–28, 2013.

[4] Graham R. Brightwell and Peter Winkler. Graph homomorphisms and phase transitions. *J. Comb. Theory, Ser. B*, 77(2):221–262, 1999.

[5] Hubie Chen. Homomorphisms are indeed a good basis for counting: Three fixed-template dichotomy theorems, for the price of one. *CoRR*, abs/1710.00234, 2017.

[6] Radu Curticapean, Holger Dell, and Dániel Marx. Homomorphisms are a good basis for counting small subgraphs. In *Proceedings of the 49th Annual ACM Symposium on Theory of Computing*, pages 210–213, 2017.

[7] Holger Dell. Note on "The Complexity of Counting Surjective Homomorphisms and Compactions". *CoRR*, abs/1710.01712, 2017.

[8] Josep Díaz, Maria Serna, and Dimitrios Thilikos. Recent results on parameterized H-coloring. In J. Nešetřil and P. Winkler, editors, *Graphs, Morphisms and Statistical Physics*, volume 63 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, 2004.

[9] Martin Dyer and Catherine Greenhill. The complexity of counting graph homomorphisms. *Random Structures & Algorithms*, 17(3-4):260–289, 2000.

[10] Tomás Feder and Pavol Hell. List homomorphisms to reflexive graphs. *J. Combin. Theory Ser. B*, 72(2):236–250, 1998.

[11] Tomás Feder, Pavol Hell, and Jing Huang. List homomorphisms and circular arc graphs. *Combinatorica*, 19(4):487–505, 1999.

[12] Tomás Feder, Pavol Hell, Peter Jonsson, Andrei Krokhin, and Gustav Nordh. Retractions to pseudoforests. *SIAM J. Discrete Math.*, 24(1):101–112, 2010.

[13] Andreas Galanis, Leslie Ann Goldberg, and Mark Jerrum. Approximately counting H-colorings is #BIS-hard. *SIAM Journal on Computing*, 45(3):680–711, 2016.

[14] Andreas Göbel, Leslie Ann Goldberg, and David Richerby. Counting homomorphisms to square-free graphs, modulo 2. *ACM Trans. Comput. Theory*, 8(3):12:1–12:29, May 2016.

[15] Petr A. Golovach, Matthew Johnson, Barnaby Martin, Daniël Paulusma, and Anthony Stewart. Surjective H-colouring: New hardness results. *CoRR*, abs/1701.02188, 2017.

[16] Petr A. Golovach, Bernard Lidický, Barnaby Martin, and Daniël Paulusma. Finding vertex-surjective graph homomorphisms. *Acta Inf.*, 49(6):381–394, 2012.

[17] Petr A. Golovach, Daniël Paulusma, and Jian Song. Computing vertex-surjective homomorphisms to partially reflexive trees. *Theoretical Computer Science*, 457:86 – 100, 2012.

[18] P. Hell and D. J. Miller. Graphs with forbidden homomorphic images. *Annals of the New York Academy of Sciences*, 319(1):270–280, 1979.

[19] P. Hell and J. Nešetřil. Homomorphisms of graphs and of their orientations. *Monatsh. Math.*, 85(1):39–48, 1978.

[20] P. Hell and J. Nešetřil. On the complexity of H-coloring. *Journal of Combinatorial Theory, Series B*, 48(1):92 – 110, 1990.

[21] P. Hell and J. Nešetřil. Counting list homomorphisms for graphs with bounded degrees. In J. Nešetřil and P. Winkler, editors, *Graphs, Morphisms and Statistical Physics*, volume 63 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 105–112, 2004.

[22] P. Hell and J. Nešetřil. *Graphs and Homomorphisms*. Oxford Lecture Series in Mathematics and Its Applications. OUP Oxford, 2004.

[23] L. Lovász. Operations with structures. *Acta Mathematica Academiae Scientiarum Hungaricae*, 1967.

[24] L. Lovász. *Large Networks and Graph Limits*. American Mathematical Society colloquium publications. American Mathematical Society, 2012.

[25] Barnaby Martin and Daniël Paulusma. The computational complexity of disconnected cut and 2k2-partition. *Journal of Combinatorial Theory, Series B*, 111:17 – 37, 2015.

[26] Christos M. Papadimitriou. *Computational complexity*. Addison-Wesley, Reading, Massachusetts, 1994.

[27] Narayan Vikas. Computational complexity of compaction to reflexive cycles. *SIAM J. Comput.*, 32(1):253–280, January 2003.

[28] Narayan Vikas. Compaction, retraction, and constraint satisfaction. *SIAM Journal on Computing*, 33(4):761–782, 2004.

[29] Narayan Vikas. Computational complexity of compaction to irreflexive cycles. *Journal of Computer and System Sciences*, 68(3):473 – 496, 2004.

[30] Narayan Vikas. A complete and equal computational complexity classification of compaction and retraction to all graphs with at most four vertices and some general results. *Journal of Computer and System Sciences*, 71(4):406 – 439, 2005.

[31] Narayan Vikas. Algorithms for partition of some class of graphs under compaction and vertex-compaction. *Algorithmica*, 67(2):180–206, 2013.