On Algorithms Based on Finitely Many Homomorphism Counts

Yijia Chen
Shanghai Jiao Tong University, Shanghai, China

Jörg Flum
Albert-Ludwigs-Universität, Freiburg i.Br., Germany

Mingjun Liu
Fudan University, Shanghai, China

Zhiyang Xun
Shanghai Jiao Tong University, Shanghai, China

Abstract

It is well known [16] that up to isomorphism a graph G is determined by the homomorphism counts $\hom(F,G)$, i.e., by the number of homomorphisms from F to G where F ranges over all graphs. Moreover, it suffices that F ranges over the graphs with at most as many vertices as G. Thus, in principle, we can answer any query concerning G with only accessing the $\hom(\cdot,G)$'s instead of G itself. In this paper, we deal with queries for which there is a *hom algorithm*, i.e., there are *finitely many* graphs $F_1,\dots,F_k$ such that for any graph G whether it is a YES-instance of the query is already determined by the vector $\overrightarrow{\hom}_{F_1,\dots,F_k}(G) := (\hom(F_1,G),\dots,\hom(F_k,G))$.

We observe that planarity of graphs and 3-colorability of graphs, properties expressible in monadic second-order logic, have no hom algorithm. On the other hand, queries expressible as a Boolean combination of universal sentences in first-order logic FO have a hom algorithm. Even though it is not easy to find FO definable queries without a hom algorithm, we succeed to show this for the non-existence of an isolated vertex, a property expressible by the FO sentence $\forall x\exists y Exy$, somehow the "simplest" graph property not definable by a Boolean combination of universal sentences. These results provide a characterization of the prefix classes of first-order logic with the property that each query definable by a sentence of the prefix class has a hom algorithm.

For *adaptive* hom algorithms, i.e., algorithms that might access a $\hom(F_{i+1},G)$ with F_{i+1} depending on $\hom(F_j,G)$ for $1 \le j \le i$ we show that *three* homomorphism counts $\hom(\cdot,G)$ are sufficient and in general necessary to determine the (isomorphism type of) G. In particular, by three adaptive queries we can answer any question on G. Moreover, adaptively accessing two $\hom(\cdot,G)$'s is already enough to detect an isolated vertex.

In 1993 Chaudhuri and Vardi [6] showed the analogue of the Lovász Isomorphism Theorem for the right homomorphism vector of a graph G, i.e, the vector of values $\hom(G,F)$ where F ranges over all graphs characterizes the isomorphism type of G. We study to what extent our results carry over to the right homomorphism vector.

2012 ACM Subject Classification Theory of computation → Graph algorithms analysis; Mathematics of computing → Graph theory

Keywords and phrases homomorphism numbers, hom algorithms, adaptive hom algorithms

Digital Object Identifier 10.4230/LIPIcs.MFCS.2022.32

Related Version *Previous Version*: https://arxiv.org/abs/2111.13269

Funding The collaboration of the first two authors is funded by the Sino-German Center for Research Promotion (GZ 1518). Yijia Chen is supported by National Natural Science Foundation of China (Project 61872092).

Acknowledgements Mingjun Liu is supervised by Yijia Chen in the Mathematical Logic Program for undergraduate students at Fudan University.

1 Introduction

In [16], one of the first papers on graph homomorphisms, Lovász proved that graphs G and H are isomorphic if and only if for all graphs F the number $\hom(F,G)$ of homomorphisms from F to G is equal to the number $\hom(F,H)$ of homomorphisms from F to H. Recently, this result has attracted a lot of attention in various contexts, e.g., algorithms and complexity [3, 9], machine learning [2, 14], and logic [1, 15]. Among others, it provides a powerful reduction of problems concerning graph structures to questions on the number of homomorphisms.

Lovász' result says that the infinite vector $\overrightarrow{\hom}(G) := (\hom(F,G))_{F \text{ a graph}}$ determines the graph G up to isomorphism. For a class C of graphs set $\overrightarrow{\hom}_{\mathsf{C}}(G) := (\hom(F,G))_{F\in\mathsf{C}}$. Using Lovász' *Cancellation Law* [17] (see Theorem 7.9) it is easy to see that for some classes C, including the class of 3-colorable graphs and the class of graphs that can be mapped homomorphically to an odd cycle, $\overrightarrow{\hom}_{\mathsf{C}}(G)$ already determines G up to isomorphisms. A further example: the class of 2-degenerate graphs has this property [12]. For other natural classes of graphs, $\overrightarrow{\hom}_{\mathsf{C}}(G)$ does not have the full power of distinguishing non-isomorphic graphs but characterizes interesting graph properties (e.g., see [10]).

We turn to results more relevant for the algorithmic problems we are interested in. Actually Lovász' proof shows that in order to determine the isomorphism type of G it is sufficient to consider the homomorphism counts $\hom(F,G)$ for the graphs F with at most as many vertices as G. As a consequence, given an oracle to $\overrightarrow{\hom}(G)$, we might answer any query by first recovering G and then computing the query on G. However, such a naive algorithm requires *exponentially* many entries in $\overrightarrow{\hom}(G)$, i.e., $\hom(F,G)$ for all isomorphism types of graphs F with at most as many vertices as G, rendering any practical implementation beyond reach.

There are queries that can be answered very easily using $\overrightarrow{\hom}(G)$, e.g., to decide whether G has a clique of size k, all we need to know is $\hom(K_k,G)$ where K_k is the complete graph on k vertices. So ideally, one would hope that to answer a query on G it suffices to access $\overrightarrow{\hom}(G)$ for finitely many fixed graphs independent of G.

The question of using $\overrightarrow{\hom}(G)$ to answer queries algorithmically has been raised before. In [9] Curticapean et al. observed that counting (induced) subgraphs isomorphic to a fixed graph F can be reduced to computing appropriate linear combinations of subvectors of $\overrightarrow{\hom}(G)$. Thereby they introduced the so-called graph motif parameters. Using this framework, they were able to design some algorithms faster than the known ones to count various specific subgraphs and induced subgraphs. These results can be understood as answering counting queries using $\hom(\cdot,G)$'s. More explicitly, Grohe [15] asked whether it is possible to answer any C^{k+1}-query in *polynomial time* by accessing $\hom(F,G)$ for graphs F of tree-width bounded by k. Here, C^{k+1} denotes counting first-order logic with $k+1$ variables [5]. Observe that without the polynomial time constraint such an algorithm exists because graphs G and H cannot be distinguished by C^{k+1} if and only if $\hom(F,G) = \hom(F,H)$ for finitely many graphs F of tree-width bounded by k [12] (see also [10]).

Our contributions

In this paper we study what Boolean queries (equivalently, graph properties) can be answered using a constant number of homomorphism counts. More precisely, let C be a class of graphs closed under isomorphism. We ask: are there graphs $F_1,\dots,F_k$ such that for any graph G whether $G\in\mathsf{C}$ can be decided knowing $\overrightarrow{\hom}_{F_1,\dots,F_k}(G) := (\hom(F_1,G),\dots,\hom(F_k,G))$.

In Section 4 we observe that this is the case if C can be defined by a sentence of first-order logic (FO) that is a Boolean combination of *universal* sentences. For $d\geq 1$ this includes the class of graphs of maximum degree d, of tree-depth [7] exactly d, and the class of graphs of

SC-depth [8] exactly d (but also the classes where we replace "exactly d" by "at most d"). On the negative side, in Section 6 we show that for any $k \geq 1$ and any $F_1, \ldots, F_k$ there are graphs G and H such that

- $\overrightarrow{\hom}_{F_1,\ldots,F_k}(G) = \overrightarrow{\hom}_{F_1,\ldots,F_k}(H)$ and G contains an isolated vertex but H does not.

That is, any $\overrightarrow{\hom}_{F_1,\ldots,F_k}(G)$ is not sufficient to detect the existence of an isolated vertex in G. This is our technically most challenging result; it requires non-trivial arguments using linear algebra. We introduce some of the tools and construction methods needed in this proof already in Section 5, thereby showing the corresponding result for the class of planar graphs and the class of 3-colorable graphs.

A graph G has no isolated vertex if it satisfies the FO-sentence $\forall x \exists y Exy$. Thus we know for what quantifier-prefix classes of FO-sentences all queries definable by a sentence of the class can be answered by $\overrightarrow{\hom}_{F_1,\ldots,F_k}(G)$ for some $F_1, \ldots, F_k$ *independent* of G.

Answering a query using $\overrightarrow{\hom}_{F_1,\ldots,F_k}(\cdot)$ can be phrased as an algorithm checking this query with *non-adaptive* access to $\overrightarrow{\hom}(G)$ on entries $F_1, \ldots, F_k$. It is also very natural to allow access to $\overrightarrow{\hom}(G)$ to be *adaptive*. Informally, on input G an *adaptive hom algorithm* still queries some $\hom(F_1, G), \ldots, \hom(F_k, G)$, but for $i = 0, \ldots, k-1$ the choice of F_{i+1} might depend on $\hom(F_1, G), \ldots, \hom(F_i, G)$ (see Definition 7.2 for a precise description). It turns out that adaptive hom algorithms are extremely powerful. In Section 7 we first present an adaptive hom algorithm with *two* accesses to $\overrightarrow{\hom}(G)$ that can decide whether G has no isolated vertices. Even more, the algorithm is able to compute the number of vertices of G of degree k for each $k \geq 0$. So in particular, it can decide whether G is regular. Furthermore, in Section 7 we provide an adaptive hom algorithm that queries *three* entries in $\overrightarrow{\hom}(G)$ and determines (the isomorphism type of) G. Hence, it can answer any question on G. The downside of this algorithm is its superpolynomial running time, while all the aforementioned hom algorithms run in polynomial time (when provided with access to $\hom(G)$). For graph classes that are relevant in applications it is a challenging task to study whether there is an algorithm where the size of the corresponding F's are polynomial in the size of G. We conjecture that there is no polynomial time algorithm that can reconstruct a graph G only accessing $\overrightarrow{\hom}(G)$ (even without the requirement of a constant number of accesses).

Results in [1] may be interpreted as saying that often proper subvectors of the right homomorphism vector $\overrightarrow{\hom}(G) := (\hom(G, F))_{F \text{ a graph}}$ of a graph G are not so expressive as the corresponding subvectors of the (left) homomorphism vector. For our topic, the finite subvectors, in Section 8 we prove that our two "positive results" on hom algorithms (namely, Theorem 4.4 on Boolean combinations of universal sentences and Theorem 7.4 on the power of 3 adaptive hom algorithms) fail for the finite right subvectors. Even when the result is the same (e.g., there is no right hom algorithm showing the non-existence of isolated vertices) the proof and its complexity can be quite different.

Due to space limitations for some proofs we refer to the full version of the paper.

2 Preliminaries

We denote by $\mathbb{N}$ the set of natural numbers greater than or equal to 0. For $n \in \mathbb{N}$ let $[n] := \{1, 2, \ldots, n\}$.

For graphs we use the notation $G = (V(G), E(G))$ common in graph theory. Here $V(G)$ is the nonempty set of vertices of G and $E(G)$ is the set of edges. We only consider finite, simple, and undirected graphs and briefly speak of graphs. To express that there is an edge connecting the vertices u and v of the graph G, we use (depending on the context) one of the notations $uv \in E(G)$ and $\{u, v\} \in E(G)$. For graphs G and H with disjoint vertex sets we

denote by $G \dot\cup H$ the *disjoint union of G and H*, i.e., the graph with vertex set $V(G) \cup V(H)$ and edge set $E(G) \cup E(H)$. If the vertex sets are not disjoint, we tacitly pass to isomorphic copies with disjoint vertex sets. Similarly, $\dot\bigcup_{i\in I} G_i$ denotes the disjoint union of the graphs G_i with $i \in I$. For a graph G and $\ell \geq 1$ we denote by ℓG the disjoint union of ℓ copies of G.

For $n \geq 1$ we denote by K_n a clique with n vertices, by P_n a path of n vertices, and by C_n a cycle of n vertices.

For graphs G and H by $G \cong H$ we express that G and H are isomorphic. All classes of graphs considered in this paper are closed under isomorphism.

▶ **Definition 2.1.** Let G and H be graphs and $f : V(G) \to V(H)$. The function f is a *homomorphism* if $uv \in E(G)$ implies $f(u)f(v) \in E(H)$ for all $u, v \in V(G)$. It is an *embedding* (or *monomorphism*) if f is a homomorphism that is one-to-one. We call f an *epimorphism* if f is a homomorphism, the range of f is $V(H)$, and for every $u'v' \in E(H)$ there are $u, v \in V(G)$ such that $uv \in E(G)$ and $f(u) = u'$, $f(v) = v'$. We get the definitions of *strong homomorphism*, *strong embedding*, and *strong epimorphism* by additionally requiring in the previous definitions that $(uv \in E(G) \iff f(u)f(v) \in E(H))$ for all $u, v \in V(G)$.

We denote by $\textsc{Hom}(G, H)$ the set of homomorphisms from G to H, thus $\hom(G, H) := |\textsc{Hom}(G, H)|$ is the number of homomorphisms from G to H. Similarly, we define s-$\textsc{Hom}(G, H)$ and s-$\hom(G, H)$ for strong homomorphisms and use corresponding notations for the other notions of morphisms. Finally, $\textsc{Aut}(G)$ and $\mathrm{aut}(G)$ denote the set of automorphisms of G and their number, respectively.

The equalities (1) and (2) are easy to verify and will often be used tacitly. For graphs F_1, F_2, and G and $\mathrm{xyz} \in \{\hom, \mathrm{emb}\}$,

$$\hom(F_1 \dot\cup F_2, G) = \hom(F_1, G) \cdot \hom(F_2, G), \tag{1}$$

$$\text{if } G \text{ is connected, then } \mathrm{xyz}(G, F_1 \dot\cup F_2) = \mathrm{xyz}(G, F_1) + \mathrm{xyz}(G, F_2). \tag{2}$$

Once and for all we fix an enumeration

$$F_1^0, F_2^0, F_3^0, \ldots \tag{3}$$

of graphs such that each graph is isomorphic to exactly one graph in the list and such that $i \leq j$ implies $F_i^0 \leq F_j^0$. Here for graphs F and G by $F \leq G$ we mean that

$$|V(F)| < |V(G)| \quad \text{or} \quad (|V(F)| = |V(G)| \text{ and } |E(F)| \leq |E(G)|).$$

In particular, F_1^0 is a graph whose vertex set is a singleton. We repeatedly use:

▶ **Theorem 2.2** (Lovász Isomorphism Theorem [16])**.** *Let G and H be graphs. If $\hom(F, G) = \hom(F, H)$ for all graphs F with $|V(F)| \leq \min\{|V(G)|, |V(H)|\}$, then G and H are isomorphic, i.e., the finite vector $\hom(F, G))_{F \text{ a graph with } |V(F)| \leq |V(G)|}$ determines G up to isomorphism.*

3 Algorithms accessing morphism counts

For what classes C of graphs is there a finite set F of graphs such that the membership of any graph G in C is determined by the values $\hom(F, G)$ where F ranges over F? This question leads to the following definition.

▶ **Definition 3.1.** Let C be a class of graphs. *A hom algorithm for* C *(with a constant number of non-adaptive accesses to homomorphism counts)* consists of a $k \geq 1$, graphs $F_1, \ldots, F_k$, and an $X \subseteq \mathbb{N}^k$ such that for all G,

$$G \in \mathsf{C} \iff (\hom(F_1, G), \ldots, \hom(F_k, G)) \in X.$$

We then say that the *hom algorithm decides* C. Analogously we define the notions of *emb algorithm*, *s-hom algorithm*, and *s-emb algorithm*.

Often we use the following fact, whose proof is immediate: A class C can be decided by a hom algorithm if and only if there is a finite set $\mathsf{F} = \{F_1, \ldots, F_k\}$ of graphs such that for all G and H (recall that $\overrightarrow{\hom}_{\mathsf{F}}(G) = (\hom(F_1, G), \ldots, \hom(F_k, G))$),

$$\overrightarrow{\hom}_{\mathsf{F}}(G) = \overrightarrow{\hom}_{\mathsf{F}}(H) \text{ implies } (G \in \mathsf{C} \iff H \in \mathsf{C}).$$

▶ **Remark 3.2.** If the set X in Definition 3.1 is decidable, then we easily extract an actual algorithm for C with an oracle to $\overrightarrow{\hom}(G)$. However the previous equivalence only holds for arbitrary X. Nevertheless, all our positive results have decidable X's. We use the current definition to ease presentation, and also to make our negative result, namely Theorem 6.1, stronger. Let C have a hom algorithm as in Definition 3.1. Then the set X can be chosen to be decidable if and only if C is decidable.

▶ **Examples 3.3.**

(a) By taking $k = 1$, a graph F whose vertex set is a singleton, and an arbitrary set $X \subseteq \mathbb{N}$, we get a hom algorithm for the class of graphs whose number of vertices is in X. In particular, for an undecidable X we get an undecidable class of graphs with a hom algorithm.

(b) Theorem 2.2 shows that every class that only contains finitely many graphs up to isomorphism can be decided by a hom algorithm.

(c) By passing from $k \geq 1$, $F_1, \ldots, F_k$, and $X \subseteq \mathbb{N}^k$ to $k \geq 1$, $F_1, \ldots, F_k$, and $\mathbb{N}^k \setminus X$, we see that with every class C also the class $\mathsf{C}^{\text{comp}} := \{G \mid G \notin \mathsf{C}\}$ has a hom algorithm.

By Definition 3.1 we have four types of algorithms (hom, emb, s-hom, s-emb). The following proposition shows that a class has an algorithm of one type if and only if it has an algorithm of any other type. This allows us to speak of a *query algorithm accessing morphism counts* (or *query algorithm* for short) if in the given context it is irrelevant to what type we refer.

▶ **Proposition 3.4.** *For a class* C *of graphs the following five statements are equivalent.*

(a) *There is a hom algorithm for* C.

(b) *There is an emb algorithm for* C.

(c) *There is an s-hom algorithm for* C.

(d) *There is an s-emb algorithm* C.

(e) *There is a hom algorithm for* C^c*, the class of graphs that are complements of graphs in* C *(the complement of a graph* G *is the graph* $G^c = \big(V(G), \{uv \mid u \neq v \text{ and } uv \notin E(G)\}\big)$.

The equivalence (a) ⇔ (b) and (c) ⇔ (d) are well known (e.g., see the proof of the Lovász Isomorphism Theorem in [15]). It uses the fact that every $h \in \text{Hom}(F, G)$ can be written as $h = f \circ g$, where for some graph F' we have $g \in \text{Epi}(F, F')$ and $f \in \text{Emb}(F', G)$. Clearly, $F' \leq F$ (as otherwise $\text{Epi}(F, F') = \emptyset$). Hence.

$$\hom(F, G) = \sum_{F' \leq F} \frac{1}{\text{aut}(F')} \cdot \text{epi}(F, F') \cdot \text{emb}(F', G), \tag{4}$$

where the sum ranges over all isomorphism types of graphs F' with $F' \leq F$ and the corresponding equation for s-hom and s-emb.

As in the literature we did not find a proof of the equivalence (a) ⇔ (c), the main tool in our proof of Theorem 4.4, we present proofs of the equivalences (a) ⇔ (c) and its consequence (a) ⇔ (e) in the full version of the paper.

▶ **Remark 3.5.** The proof of the equivalences of (a) to (e) show: If for C we have a query algorithm of one type based on graphs $F_1, \dots, F_k$ and $m := \max\{|V(F_i)| \mid i \in [k]\}$, then for any other type we can compute finitely many graphs, all with at most m vertices, that are the graphs of a query algorithm for C of this other type.

4 FO-definable classes with query algorithms

We start by showing that every class of graphs that excludes a finite set of graphs as induced subgraphs has a query algorithm. Of course, the complement and the union of such classes again have such an algorithm. In terms of first-order logic this means that every class axiomatizable by a Boolean combination of universal sentences has a query algorithm.

For a finite set F of graphs we define the class $\textsc{Forb}(\mathsf{F})$ by

$$\textsc{Forb}(\mathsf{F}) := \{G \mid G \text{ has no induced subgraph isomorphic to a graph in } \mathsf{F}\}.$$

We say that *a class C of graphs is definable by a set of forbidden induced subgraphs* if there is a finite set F with $\mathsf{C} = \textsc{Forb}(\mathsf{F})$.

Examples of classes definable by a set of forbidden induced subgraphs are classes of bounded vertex cover number (attributed to Lovász), of bounded tree-depth [11], or even of bounded shrub-depth [13]. All these classes have a query algorithm:

▶ **Lemma 4.1.** *Every class of graphs definable by a set of forbidden induced subgraphs can be decided by a query algorithm.*

Proof. If $\mathsf{C} = \textsc{Forb}(\emptyset)$, we set $k = 1$, let F be an arbitrary graph, and take $X := \mathbb{N}$. Assume now that $\mathsf{C} = \textsc{Forb}(\mathsf{F})$ with $\mathsf{F} = \{F_1, \dots, F_k\}$ and $k \geq 1$. Then, for $X = \{(0, 0, \dots, 0)\} \subseteq \mathbb{N}^k$,

$$G \in \mathsf{C} \iff \big(\text{s-emb}(F_1, G), \dots, \text{s-emb}(F_k, G)\big) \in X.$$

Hence, k, $F_1, \dots, F_k$, and X constitute an s-emb algorithm for C. ◀

The following lemma shows that the universe of classes with query algorithms is closed under the Boolean operations. Part (a) was already mentioned as Examples 3.3 (c). We omit the straightforward proof.

▶ **Lemma 4.2.**
(a) *If C has a query algorithm, then so does $\{G \mid G \notin \mathsf{C}\}$.*
(b) *If C and C' have query algorithms, then $\mathsf{C} \cap \mathsf{C}'$ and $\mathsf{C} \cup \mathsf{C}'$ have query algorithms.*

Recall that *formulas φ* of *first-order logic* FO for graphs are built up from *atomic formulas* $x = y$ and Exy (where x, y are variables) using the Boolean connectives $\neg$, $\wedge$, and $\vee$ and the universal $\forall$ and existential $\exists$ quantifiers. A *sentence* is a formula without free variables (i.e., all variables of φ are in the scope of a corresponding quantifier). If φ is a sentence, we denote by $\mathsf{C}(\varphi)$ the class of graphs that are models of φ.

An FO-formula is *universal* if it is built up from atomic and negated atomic formulas by means of the connectives $\wedge$ and $\vee$ and the universal quantifier $\forall$. If in this definition we replace the universal quantifier by the existential one, we get the definition of an *existential formula*. The following result is well known (e.g., see [18]).

▶ **Lemma 4.3.** *Let* C *be a class of graphs. Then* C *is the class of graphs definable by a universal sentence if and only if* C *is definable by a set of forbidden induced subgraphs.*

By Lemma 4.1 – Lemma 4.3 we get:

▶ **Theorem 4.4.** *If the* FO*-sentence* φ *is a Boolean combination of universal sentences, then there is a query algorithm for* $\mathsf{C}(\varphi)$*.*

▶ **Remark 4.5.** The class $\mathsf{C}(3)$ of 3-regular graphs (each vertex has exactly 3 neighbors) is an example of a class decidable by a query algorithm, definable in FO but not by a Boolean combination of universal sentences. Indeed, using the following steps we get a query algorithm deciding whether a graph G belongs to $\mathsf{C}(3)$.

- We check whether $G \in \mathsf{C}(\leq 3)$, i.e., whether each vertex has at most 3 neighbors. Note that $\mathsf{C}(\leq 3)$ is definable by a universal sentence. Hence there is a hom algorithm for $\mathsf{C}(\leq 3)$, say consisting of $k \geq 1$, $F_1, \ldots, F_k$, and $X_{\leq 3}$ ($\subseteq \mathbb{N}^k$).
- We query $\hom(K_1, G)$ in order to get $n := |V(G)|$.
- We query $\hom(P_2, G)$, i.e., the number of homomorphisms from the path P_2 of two vertices to G. Then, G is 3-regular if and only if $\hom(P_2, G) = 3 \cdot n$.

Hence, we have a hom algorithm for $\mathsf{C}(3)$ consisting of $k+2$, $F_1, \ldots, F_k, K_1, P_2$, and X where X consists of all tuples $(n_1, \ldots, n_k, n, 3n)$ with $(n_1, \ldots, n_k) \in X_{\leq 3}$ and $n \geq 1$.

The class $\mathsf{C}(3)$ is definable in FO by

$$\forall x \forall y_1 \ldots \forall y_4 \exists z_1 \ldots \exists z_3 \Big(\neg \big(\bigwedge_{1 \leq i < j \leq 4} y_i \neq y_j \wedge \bigwedge_{i \in [4]} Exy_i \big) \wedge \big(\bigwedge_{1 \leq i < j \leq 3} z_i \neq z_j \wedge \bigwedge_{j \in [3]} Exz_j \big) \Big).$$

If it would be definable by a Boolean combination of universal sentences, then it would be also definable by a sentence φ of the form $\varphi = \exists x_1 \ldots \exists x_m \forall y_1 \ldots \forall y_\ell \psi$ with $m, \ell \in \mathbb{N}$ and with quantifier-free ψ. Let G be a graph with more than $m+1$ vertices that is the disjoint union of copies of the clique K_4. Of course, G is 3-regular. Hence, G is a model of φ. In particular, there are vertices $u_1, \ldots, u_m$ that satisfy in G the formula $\forall y_1 \ldots \forall y_\ell \psi(x_1, \ldots, x_m)$ if we interpret x_1 by u_1, …, x_m by u_m. Choose a vertex $u \in V(G) \setminus \{u_1, \ldots, u_m\}$. Then, $G \setminus u$, the graph induced by G on $V(G) \setminus \{u\}$, is still a model of φ but not 3-regular.

By the previous remark the question arises whether every class $\mathsf{C}(\varphi)$ for an FO-sentence φ of the form$\forall x_1 \ldots \forall x_m \exists y_1 \ldots \exists y_\ell \psi$ with quantifier free ψ can be decided by a query algorithm. We will see that already for the simple formula $\forall x \exists y Exy$ of this type, the class $\mathsf{C}(\forall x \exists y Exy)$, i.e., the class of graphs not containing isolated vertices, has no query algorithm.

5 Planarity and 3-colorability

As just mentioned we want to show that no query algorithm detects the existence of an isolated vertex. In this section we prove the corresponding result for planarity, where some easy tools and construction methods relevant in the much more involved proof for isolated vertices are used. Essentially by a similar proof one can show the same result for 3-colorability. Note that the class of planar graphs and the class of 3-colorable graphs are definable in monadic second-order logic but not in first-order logic.

By the following lemma a class has no query algorithm if there is no emb algorithm for this class that only uses *connected graphs*:

▶ **Lemma 5.1.** *Let* C *be a class of graphs. Assume that for every finite set* K' *of connected graphs there are graphs* G *and* H *such that (a) and (b) hold.*

(a) $G \in \mathsf{C}$ *and* $H \notin \mathsf{C}$.

(b) *For all* $F' \in \mathsf{K}'$ *we have* $\mathrm{emb}(F', G) = \mathrm{emb}(F', H)$.

Then there is no hom algorithm for C*, i.e., for every finite set* K *of graphs there are graphs* G *and* H *such that (c) and (d) hold*

(c) $G \in \mathsf{C}$ *and* $H \notin \mathsf{C}$.

(d) *For all* $F \in \mathsf{K}$ *we have* $\hom(F, G) = \hom(F, H)$.

Proof. By (1), $\hom(F^1 \dot\cup F^2, F^3) = \hom(F^1, F^3) \cdot \hom(F^2, F^3)$ holds for arbitrary graphs F^1, F^2, F^3. Thus, if the class of connected components of graphs in a finite set K satisfies (d) for some G and H, then K satisfies (d), too. Hence, we can assume that the graphs in K are connected. Let $n := \max\{|V(F)| \mid F \in \mathsf{K}\}$ and $\mathsf{K}' := \{F_i^0 \mid i \geq 1, |V(F_i^0)| \leq n, F_i^0 \text{ connected}\}$. By assumption we know that there are graphs G and H such that (a) and (b) hold for K'. Now we recall (4), i.e.,

$$\hom(F, G) = \sum_{F' \leq F} \frac{1}{\mathrm{aut}(F')} \cdot \mathrm{epi}(F, F') \cdot \mathrm{emb}(F', G).$$

If F is connected, then $\mathrm{epi}(F, F') > 0$ implies that F' is connected, too. That is, the values $\hom(F, G)$ for $F \in \mathsf{K}$ are determined by the values of $\mathrm{emb}(F', G)$ for $F' \in \mathsf{K}'$. Therefore, (d) holds by (b). ◀

▶ **Theorem 5.2.** *The class* P *of planar graphs and the class* C(*3-col*) *of 3-colorable graphs have no query algorithm.*

Proof. Here we only present the proof for P. For a contradiction, by Lemma 5.1 we can assume that there is an emb algorithm for P that uses the finite set F of connected graphs. By induction on $n := |\mathsf{F}|$, we show that this cannot be the case. Let $|\mathsf{F}| = 1$, i.e., $\mathsf{F} = \{F\}$ for some connected F. Set $k := |V(F)| + 4$. Clearly, $K_k \notin \mathsf{P}$ and both, $\mathrm{emb}(F, F)$ and $\mathrm{emb}(F, K_k)$, are nonzero. If F is planar, we set (recall that for a graph G and $p \in \mathbb{N}$ by pG we denote the disjoint union of p copies of G),

$$G := \mathrm{emb}(F, K_k)F \quad \text{and} \quad H := \mathrm{emb}(F, F)K_k$$

By (2), $\mathrm{emb}(F, G) = \mathrm{emb}(F, K_k) \cdot \mathrm{emb}(F, F) = \hom(F, H)$ and $G \in \mathsf{P}$, $H \notin \mathsf{P}$, a contradiction. If F is not planar, we set $G := K_1$ and take as H a topological minor of K_k in which every edge in K_k is subdivided into $1 + |(V(F)|$ edges. Then $H \notin \mathsf{P}$ but $\mathrm{emb}(F, G) = 0 = \mathrm{emb}(F, H)$, again a contradiction.

Now assume $|\mathsf{F}| \geq 2$. If F contains no planar graphs, we essentially can proceed as in the preceding case. So assume that F contains a planar graph. Choose a "minimal" (w.r.t. $\leq$) planar graph $F \in \mathsf{F}$ and set $\mathsf{F}' := \mathsf{F} \setminus \{F\}$. By minimality, $\mathrm{emb}(F', F) = 0$ for all planar $F' \in \mathsf{F}'$. As there is no embedding from a non-planar graph to a planar graph, we have $\mathrm{emb}(F', F) = 0$ for all $F' \in \mathsf{F}'$. By induction hypothesis, there are G_0 and H_0 satisfying the desired properties with respect to F'. If $\mathrm{emb}(F, G_0) = \mathrm{emb}(F, H_0)$, then we can simply take $G := G_0$ and $H := H_0$. Otherwise, assume first that $\mathrm{emb}(F, G_0) < \mathrm{emb}(F, H_0)$. We set

$$G := \mathrm{aut}(F)G_0 \dot\cup (\mathrm{emb}(F, H_0) - \mathrm{emb}(F, G_0))F \quad \text{and} \quad H := \mathrm{aut}(F)H_0.$$

Hence, $G \in \mathsf{P}$, $H \notin \mathsf{P}$, and $\mathrm{emb}(F, G) = \mathrm{emb}(F, H)$ (by (2)). In case $\mathrm{emb}(F, G_0) > \mathrm{emb}(F, H_0)$ we argue similarly. ◀

6 No query algorithm detects isolated vertices

▶ **Theorem 6.1.** *The class* $\mathsf{C}(\forall x \exists y Exy)$ *of graphs without isolated vertices has no query algorithm.*

In the proof given in the full version of the paper we use some tools already used in the preceding section:

- By Lemma 5.1 it suffices to show that $\mathsf{C}(\forall x \exists y Exy)$ has no emb algorithm that only uses connected graphs.
- If C is one of the classes P or $\mathsf{C}(\forall x \exists y Exy)$, then the disjoint union $\dot{\bigcup}_{i \in I} G_i$ of a family $(G_i)_{i \in I}$ of graphs is in C if and only if each G_i is in C.
- By (2), for graphs F_1 and F_2, $p, q \geq 1$, and a connected graph G, we have

$$\operatorname{emb}(G, pF_1 \dot{\cup} qF_2) = p \operatorname{emb}(G, F_1) + q \operatorname{emb}(G, F_2),$$

  i.e., $\operatorname{emb}(G, pF_1 \dot{\cup} qF_2)$ is a linear combination of $\operatorname{emb}(G, F_1)$ and $\operatorname{emb}(G, F_2)$.

Furthermore note:

- Let G be a connected graph with more than one vertex. If the graph F' is obtained from the graph F by adding a set of isolated vertices, then $\operatorname{emb}(G, F') = \operatorname{emb}(G, F)$.

Recall that in (3) we fixed the enumeration $F_1^0, F_2^0, \ldots$ of graphs that contains a copy of every isomorphism type and respects the relation $\leq$. Here we let $H_1, H_2, \ldots$ be the subsequence of $F_1^0, F_2^0, \ldots$ consisting of the connected graphs.

For $i \geq 1$ let $\alpha_i := (\operatorname{emb}(H_i, H_j))_{j \geq 1}$ be the vector containing the emb-values of H_i for connected graphs. We sketch the idea underlying the proof of Theorem 6.1. The central idea can be vaguely expressed by saying that for each $n \geq 1$ there is an $r_n \in \mathbb{N}$ such that appropriate "subvectors" of length r_n of r_n-many of the $\alpha_1, \ldots, \alpha_n$ are linearly independent vectors of the vector space $\mathbb{Q}^{r_n}$ and hence, a basis of $\mathbb{Q}^{r_n}$. In particular, every further vector of $\mathbb{Q}^{r_n}$ is a linear combination of these vectors. Furthermore, r_n tends to infinity when n increases.

For an arbitrary emb algorithm with connected graphs we must show the existence of graphs G and H, one with isolated vertices the other one without, that cannot be distinguished by this emb algorithm. We use the tools mentioned above to construct such graphs using the knowledge about the linear independence or linear dependence of some tuples of vectors obtained in the first steps of the proof.

7 Adaptive hom algorithms

For $r \geq 1$ let S_r denote the star of r vertices, i.e., a graph that consists of a vertex of degree $r-1$ (the center of the star) and $r-1$ vertices of degree 1, all neighbors of the center. For a vertex u of a graph we denote by $\deg(u)$ its degree. Note that $\deg(u) = 0$ means that u is isolated. The proof in the full paper of the following result is built on the well-known equality (see e.g., [4, 15]) obtained by looking at the value of the center of a star under a homomorphism: $\hom(S_r, G) = \sum_{v \in V(G)} \deg(v)^{r-1}$.

▶ **Proposition 7.1.** *Let* G *be a graph and* $d_i := |\{u \in V(G) \mid \deg(u) = i\}|$ *for* $i \geq 0$. *If* $n := |V(G)|$, *then* $\hom(S_{n \cdot \log n}, G)$ *determines* $d_0, \ldots, d_{n-1}$.

In Section 6 we showed that there is no query algorithm that decides whether a graph G is in $\mathsf{C}(\forall x \exists y Exy)$, i.e., whether $d_0 = 0$ for G. However, Proposition 7.1 shows that for a graph G with n vertices this can be decided by querying $\hom(S_{n \cdot \log n}, G)$. Thus, we have an algorithm for $\mathsf{C}(\forall x \exists y Exy)$ consisting of two homomorphism counts:

- query $n := \hom(F_1^0, G)$ $(= |V(G)|)$,
- query $\hom(S_{n \cdot \log n}, G)$.

That is, the selection of the graph for the second homomorphism count, in our case $S_{n \cdot \log n}$, depends on the answer to the first query. This leads to the notion of adaptive hom algorithm. Recall that $F_1^0, F_2^0, \ldots$ is an enumeration of all graphs (up to isomorphism) respecting $\leq$.

▶ **Definition 7.2.** Let C be a class of graphs and $k \geq 1$. A *k adaptive hom algorithm for* C consists of a function $g : \{\emptyset\} \cup \bigcup_{i \in [k-1]} \mathbb{N}^i \to \mathbb{N}$ and a subset X of $\mathbb{N}^k$ such that for all G,

$$G \in \mathsf{C} \iff (n_1, \ldots, n_k) \in X,$$

where $n_1 := \hom(F^0_{g(\emptyset)}, G)$, $n_2 := \hom(F^0_{g(n_1)}, G)$, $\ldots$, $n_k := \hom(F^0_{g(n_1, n_2, \ldots, n_{k-1})}, G)$. We then say that C *can be decided by a k adaptive hom algorithm.*

The main result of this section:

▶ **Theorem 7.3.** *Every class C can be decided by a 3 adaptive hom algorithm.*

To get this result we show:

▶ **Theorem 7.4.** *For $n \geq 1$ there exist graphs F_1 $(= F_1(n))$ and F_2 $(= F_2(n))$ such that for all graphs G and H with n vertices,*

$$\hom(F_1, G) = \hom(F_1, H) \quad \textit{and} \quad \hom(F_2, G) = \hom(F_2, H) \quad \textit{imply} \quad G \cong H.$$

In fact, by this result for any class C of graphs, we get the 3 adaptive hom algorithm that for a graph G queries

- $\hom(F_1^0, G)$; set $n := \hom(F_1^0, G)$
- $\hom(F_1(n), G)$ and $\hom(F_2(n), G)$

(where $F_1(n)$ and $F_2(n)$ are the graphs of Theorem 7.4) and has as set X the set

$$X := \{(n, \hom(F_1(n), H), \hom(F_2(n), H)) \mid n \geq 1,\ H \in \mathsf{C}, \text{ and } |V(H)| = n\}.$$

▶ **Corollary 7.5.** *For graphs G and H, if $n_0 := \hom(F_1^0, G) = \hom(F_1^0, H)$, $\hom(F_1(n_0), G) = \hom(F_1(n_0), H)$, and $\hom(F_2(n_0), G) = \hom(F_2(n_0), H)$, then $G \cong H$.*

Hence, by "3 adaptive hom counts" we can characterize the isomorphism type of any graph. It is not possible to do this by two queries, more precisely (for a proof see the full paper):

▶ **Theorem 7.6.** *There is no $s_0 \in \mathbb{N}$ such that for some function $g : \mathbb{N} \to \mathbb{N}$ and all graphs G and H, if $n_0 := \hom(F^0_{s_0}, G) = \hom(F^0_{s_0}, H)$ and $\hom(F^0_{g(n_0)}, G) = \hom(F^0_{g(n_0)}, H)$, then $G \cong H$.*

We turn to a proof of Theorem 7.4. An important tool will be the following lemma.

▶ **Lemma 7.7.** *Let $n \geq 1$ and K be a finite set of graphs. We can construct a graph F_{K} such that for all G and H with exactly n vertices we have $\hom(F_{\mathsf{K}}, G) = \hom(F_{\mathsf{K}}, H)$ if and only if G and H satisfy at least one of the conditions (a) and (b).*

(a) *There exist $F, F' \in \mathsf{K}$ such that $\hom(F, G) = 0$ and $\hom(F', H) = 0$.*

(b) *For all $F \in \mathsf{K}$, $\hom(F, G) = \hom(F, H)$.*

Proof. The idea of the construction is best seen by assuming $\mathsf{K} = \{F_1, F_2\}$ (iterating the following process one gets the general case). We set $r := n^{|V(F_1)|}$. As $|V(G)| = n$, we know that $\hom(F_1, G) \leq r$. We set $F_{\mathsf{K}} := F_1 \dot\cup rF_2$. By (1) for every graph F,

$$\hom(F_{\mathsf{K}}, F) = \hom(F_1, F) \cdot \hom(F_2, F)^r. \tag{5}$$

Hence, if (a) or (b) hold, then $\hom(F_{\mathsf{K}}, G) = \hom(F_{\mathsf{K}}, H)$. Conversely, assume $z := \hom(F_{\mathsf{K}}, G) = \hom(F_{\mathsf{K}}, H)$. If $z = 0$, then (a) must hold by (5). If $z = 1$, then $\hom(F_i, G) = \hom(F_i, H) = 1$ for $i \in [2]$ and (b) holds. Otherwise, $z \geq 2$. Let $F := G$ or $F := H$ and set $x := \hom(F_1, F)$ and $y := \hom(F_2, F)$. Let p be a prime number with $p|z$ (i.e., p divides z). Choose the maximum k such that $p^k|z$. Write k in the form $k = \ell \cdot r + m$ with $0 \leq m < r$. As $z = x \cdot y^r$ and $x \leq r$, the factor p^m appears in x and the factor p^ℓ in y. This determines x and y; they do not depend on the $F \in \{G, H\}$ chosen. ◀

If the alternative (b) holds, we can replace in a hom algorithm the set K of graphs by the single graph F_{K}. The previous lemma doesn't help too much if the alternative (a) holds. To overcome this alternative essentially we consider the bipartite graphs and the non-bipartite graphs separately. For this purpose we recall the definition and some simple facts on bipartite graphs.

A graph G is *bipartite* if there is a partition $V(G) = X \dot\cup Y$ such that each edge has one end in X and one end in Y. The next lemma contains some simple facts on bipartite graphs.

▶ **Lemma 7.8.** *Let G be a graph. Then:*

(a) *G is bipartite $\iff$ $\hom(G, P_2) \neq 0$.*
(b) *If G is connected and bipartite, then $\hom(G, P_2) = 2$.*
(c) *G is bipartite $\iff$ $\hom(G, H) \neq 0$ for all graphs H with at least one edge.*
(d) *G is bipartite $\iff$ G does not contain a cycle of odd length.*
(e) *If G is bipartite and F is not, then $\hom(F, G) = 0$.*
(f) *If G is bipartite, then G is determined (up to isomorphism) by the values $\hom(F, G)$ for the bipartite graphs F with $F \leq G$ (by the Lovász Isomorphism Theorem and part (e)).*

For graphs G and H the *product $G \times H$ of G and H* is the graph with $V(G \times H) := \{(u, v) \mid u \in V(G),\ v \in V(H)\}$ and $E(G \times H) := \{\{(u, v), (u', v')\} \mid \{u, u'\} \in E(G) \text{ and } \{v, v'\} \in E(H)\}$. One easily verifies that for any graph H,

$$\hom(F, G \times H) = \hom(F, G) \cdot \hom(F, H). \tag{6}$$

Besides the simple facts on bipartite graphs mentioned above, we also need a deep result:

▶ **Theorem 7.9** (Lovász Cancellation Law [17])**.** *A graph H is not bipartite if and only if $F \times H \cong G \times H$ implies $F \cong G$ for all graphs F and G.*

The following lemma contains a further step for the proof of Theorem 7.4.

▶ **Lemma 7.10.** *Let $n \geq 2$ and t be the smallest natural number with $n \leq 2t$. We set (recall that C_m denotes a cycle of length m)*

$$\mathsf{K}_t := \{F \mid \hom(F, C_{2t+1}) > 0 \text{ and } |V(F)| \leq (2t+1)^2\}. \tag{7}$$

For graphs G and H with $|V(G)| = |V(H)| = n$, if $\hom(F, G) = \hom(F, H)$ for all $F \in \mathsf{K}_t$, then $G \cong H$.

Proof. Assume $G \ncong H$. By Lemma 7.8 (d), C_{2t+1} is not bipartite. Thus, by the Lovász Cancellation Law $G \times C_{2t+1} \ncong H \times C_{2t+1}$ As $G \times C_{2t+1}$ has $n \cdot (2t+1)$ vertices, by the Lovász Isomorphism Theorem (see Theorem 2.2) there is a graph F with $|V(F)| \leq n \cdot (2t+1) \leq (2t+1)^2$ such that

$$\hom(F, G \times C_{2t+1}) \neq \hom(F, H \times C_{2t+1}). \tag{8}$$

If F is not in K_t, then $\hom(F, C_{2t+1}) = 0$ and thus by (6) we get a contradiction to (8). Hence, $F \in \mathsf{K}_t$ and so by assumption, $\hom(F, G) = \hom(F, H)$. However, this contradicts (8) (use again (6)). ◀

Proof of Theorem 7.4. The case $n = 1$ is trivial. Assume $n \geq 2$. Let again t be the smallest natural number with $n \leq 2t$ and K_t be defined by (7). For the class K_{bip} of bipartite graphs in K_t let F_1 be the graph constructed in Lemma 7.7 for K_{bip} (i.e., $F_1 = F_{\mathsf{K}_{\text{bip}}}$). As the disjoint union of bipartite graphs is bipartite, the proof of Lemma 7.7 shows that F_1 is bipartite. Let F_2 be the graph constructed in Lemma 7.7 for the class $\mathsf{K}_t \setminus \mathsf{K}_{\text{bip}}$. We show that for graphs G and H with $|V(G)| = |V(H)| = n$,

$$\hom(F_i, G) = \hom(F_i, H) \text{ for } i \in [2] \text{ implies } G \cong H.$$

If G and H both have no edge, then clearly $G \cong H$. If, say G has an edge but $E(H) = \emptyset$, then, as F_1 is bipartite, $\hom(F_1, G) \neq 0 = \hom(F_1, H)$ by Lemma 7.8 (c) and $E(F_1) \neq \emptyset$. Hence we can assume that both graphs contain at least one edge. For a contradiction assume that $\hom(F_i, G) = \hom(F_i, H)$ for $i \in [2]$ and that $G \ncong H$. Then, by Lemma 7.10 there is a graph $F_0 \in \mathsf{K}_t$ with

$$\hom(F_0, G) \neq \hom(F_0, H). \tag{9}$$

Assume first that $F_0 \in \mathsf{K}_{\text{bip}}$. As G and H contain at least one edge, by Lemma 7.8 (c)

$$\hom(F, G) > 0 \quad \text{and} \quad \hom(F, H) > 0,$$

for every bipartite graph F. In particular, this holds for all graphs F in K_{bip}. Thus, $\hom(F_1, G) = \hom(F_1, H)$ implies the second case in Lemma 7.7, i.e., for every $F \in \mathsf{K}_{\text{bip}}$,

$$\hom(F, G) = \hom(F, H).$$

In particular, this holds for $F = F_0$ contradicting (9).

Thus $F_0 \in \mathsf{K}_t \setminus \mathsf{K}_{\text{bip}}$. Then $\hom(F_0, F) = 0$ for all bipartite graphs F $\big($by Lemma 7.8 (e)$\big)$. Hence, by (9) at least one of G and H must be non-bipartite. By $\hom(F_2, G) = \hom(F_2, H)$, Lemma 7.7, and (9) there exist graphs F^G and F^H in $\mathsf{K}_t \setminus \mathsf{K}_{\text{bip}}$ with

$$\hom(F^G, G) = \hom(F^H, H) = 0.$$

W.l.o.g. suppose that G is not bipartite and thus contains an odd cycle, say of length ℓ. Since $\ell \leq n \leq 2t+1$, we have $\hom(C_{2t+1}, C_\ell) > 0$. In fact, one easily verifies that $\hom(C_k, C_m) > 0$ for odd m and k with $m < k$. As $F^G \in \mathsf{K}_t$, $\hom(F^G, C_{2t+1}) > 0$. Therefore, $\hom(F^G, C_\ell) > 0$, which implies $\hom(F^G, G) > 0$, a contradiction. ◀

8 Right hom algorithms

In 1993 Chaudhuri and Vardi [6] (see also [1]) showed the analogue of the Lovász Isomorphism Theorem for the right homomorphism vector of a graph G. More precisely, the vector of values $\hom(G, F)$, where F ranges over all graphs, characterizes the isomorphism type of G. In this section we will see that our main "positive" results on hom algorithms fail for right hom algorithms (see Proposition 8.2) and that various results that survive use a completely different proof technique. Note that a graph G is 3-colorable if $\hom(G, K_3) > 0$ in contrast to Theorem 5.2.

▶ **Definition 8.1.** A class C of graphs can be *decided by a right hom algorithm* if and only if there is a $k \geq 1$ and graphs $F_1, \ldots, F_k$ such that for all graphs G and H,

$$\hom(G, F_1) = \hom(G, F_k), \ldots, \hom(G, F_k) = \hom(G, F_k) \text{ imply } (G \in \mathsf{C} \iff H \in \mathsf{C}),$$

or equivalently, if in addition to $F_1, \ldots, F_k$ there is a set $X \subseteq \mathbb{N}^k$ such that for any graph G, $\big(G \in \mathsf{C} \iff (\hom(G, F_1), \ldots, \hom(G, F_k)) \in X\big)$.

Again one can show that C is decidable if and only if as X can be chosen a decidable set.

It should be clear how we define *k adaptive right hom algorithms* for a class C of graphs.

The failure in the "right world" of the "positive" results Theorem 4.4 and Theorem 7.3 is shown by (it should be clear how we define *k adaptive right hom algorithms* for a class C of graphs):

▶ **Proposition 8.2.** *let $k \geq 1$. The class $\mathsf{K}(3)$ of graphs containing a clique of size 3 (expressible by the existential sentence $\exists x \exists y \exists z(Exy \wedge Eyz \wedge Exz)$) cannot be decided by a k adaptive right hom algorithm (and hence not by a right hom algorithm).*

Proof. For a graph G we denote by $\chi(G)$ the chromatic number of G, i.e., the least s such that G is s-colorable. Clearly, $\big(m < \chi(G) \iff \hom(G, K_m) = 0\big)$ for the clique K_m with m elements, and hence, for every graph F,

$$\text{if } |V(F)| < \chi(G), \text{ then } \hom(G, F) = 0. \tag{10}$$

For a contradiction, assume that g and X (compare Definition 7.2) witness the existence of a k adaptive right hom algorithm for C. Then set

$$n_1 := g(\emptyset),\ n_2 := g(0),\ n_3 := g(0,0), \ldots, n_k := g(\underbrace{0, \ldots, 0}_{k-1 \text{ times}}).$$

Let $s > 3$ be bigger than any of the $|V(F^0_{n_i})|$'s. According to [19] there is a $G_0 \notin \mathsf{K}(3)$ such that $\chi(G_0) = s$. Thus by (10), we have $\hom(G_0, F^0_{n_1}) = 0, \ldots, \hom(G_0, F^0_{n_k}) = 0$ and hence, $(0, 0, \ldots, 0) \notin X$. However, $K_s \in \mathsf{K}(3)$ and $\hom(K_s, F^0_{n_1}) = 0, \ldots, \hom(K_s, F^0_{n_k}) = 0$, thus $(0, 0, \ldots, 0) \in X$, a contradiction. ◀

Note that Theorem 10 in [1] cannot be applied to show (directly) the existence of an FO-sentence φ with no right hom algorithm. To apply this theorem to such a φ we should have for all graphs G and H, $(G \models \varphi \iff H \models \varphi)$ implies $|V(G)| = |V(H)|$. One easily verifies that this condition is not satisfied by any FO-sentence φ.

We mention a positive result on right hom algorithms (proven in the full version of the paper).

▶ **Theorem 8.3.** *Every class* C *of graphs with the property that there is a bound on the number of edges of graphs in* C *has a right hom algorithm.*

Finally we remark that some results with nontrivial proofs for hom algorithms are trivial for right hom algorithms. For example, the following simple proof shows that there is no right hom algorithm for the class $\mathsf{C}(\forall x \exists y Exy)$ of graphs with no isolated vertices.

Let $F_1, \dots, F_k$ be any finite set of graphs and set $m := 1 + \max\{|V(F_i)| \mid i \in [k]\}$. Then, $\hom(K_m, F_i) = 0 = \hom(K_m \dot\cup K_1)$ for every $i \in [k]$. Thus there is no right hom algorithm that detects an isolated vertex.

References

1 A. Atserias, P. Kolaitis, and W. Wu. On the expressive power of homomorphism counts. In *36th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2021, Rome, Italy, June 29 - July 2, 2021*, pages 1–13. IEEE, 2021.

2 J. Böker. Graph similarity and homomorphism densities. In *48th International Colloquium on Automata, Languages, and Programming, ICALP 2021, July 12-16, 2021, Glasgow, Scotland (Virtual Conference)*, volume 198 of *LIPIcs*, pages 32:1–32:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021.

3 J. Böker, Y. Chen, M. Grohe, and G. Rattan. The complexity of homomorphism indistinguishability. In *44th International Symposium on Mathematical Foundations of Computer Science, MFCS 2019*, pages 54:1–54:13, 2019.

4 C. Borgs, J. Chayes, L. Lovász, V.T. Sós, and K. Vesztergombi. Counting graph homomorphisms. In *Topics in Discrete Mathematics*, pages 315–371, 2006.

5 J. Cai, M. Fürer, and N. Immerman. An optimal lower bound on the number of variables for graph identifications. *Combinatorica*, 12(4):389–410, 1992.

6 S. Chaudhuri and M. Y. Vardi. Optimization of *Real* conjunctive queries. In *Proceedings of the Twelfth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, 1993*, pages 59–70. ACM Press, 1993.

7 Y. Chen and J. Flum. Tree-depth, quantifier elimination, and quantifier rank. In *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2018, Oxford, UK, July 09-12, 2018*, pages 225–234, 2018.

8 Y. Chen and J. Flum. FO-definability of shrub-depth. In *28th EACSL Annual Conference on Computer Science Logic, CSL 2020, January 13-16, 2020, Barcelona, Spain*, pages 15:1–15:16, 2020.

9 R. Curticapean, H. Dell, and D. Marx. Homomorphisms are a good basis for counting small subgraphs. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC*, pages 210–223. ACM, 2017.

10 H. Dell, M. Grohe, and G. Rattan. Lovász meets Weisfeiler and Leman. In *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018*, volume 107, pages 40:1–40:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018.

11 G. Ding. Subgraphs and well-quasi-ordering. *Journal of Graph Theory*, 16(5):489–502, 1992.

12 Z. Dvorák. On recognizing graphs by numbers of homomorphisms. *Journal of Graph Theory*, 64(4):330–342, 2010.

13 R. Ganian, P. Hlinený, J. Nesetril, J. Obdrzálek, P. Ossona de Mendez, and R. Ramadurai. When trees grow low: Shrubs and fast MSO_1. In *Mathematical Foundations of Computer Science 2012 - 37th International Symposium, MFCS 2012, Bratislava, Slovakia, August 27-31, 2012. Proceedings*, pages 419–430, 2012.

14 M. Grohe. Counting bounded tree depth homomorphisms. In *LICS '20: 35th Annual ACM/IEEE Symposium on Logic in Computer Science, 2020*, pages 507–520. ACM, 2020.

15 M. Grohe. word2vec, node2vec, graph2vec, x2vec: Towards a theory of vector embeddings of structured data. In *Proceedings of the 39th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS 2020*, pages 1–16. ACM, 2020.

16 L. Lovász. Operations with structures. *Acta Mathematica Academiae Scientiarum Hungarica*, 18:321–328, 1967.

17 L. Lovász. On the cancellation law among finite relational structures. *Periodica Mathematica Hungarica*, 1:145–156, 1971.

18 T. A. McKee. Forbidden subgraphs in terms of forbidden quantifiers. *Notre Dame Journal of Formal Log.*, 19:186–188, 1978.

19 J. Mycielski. Sur le coloriage des graphes. *Information Processing Letters*, 108(6):412–417, 2008.