A More General Theory of Static Approximations for Conjunctive Queries

Pablo Barceló[1], Miguel Romero[1], and Thomas Zeume[2]

1 **Center for Semantic Web Research & DCC, University of Chile**
{pbarcelo,mromero}@dcc.uchile.cl
2 **TU Dortmund University & University Warsaw**
thomas.zeume@cs.tu-dortmund.de

Abstract

Conjunctive query (CQ) evaluation is NP-complete, but becomes tractable for fragments of bounded hypertreewidth. If a CQ is hard to evaluate, it is thus useful to evaluate an approximation of it in such fragments. While underapproximations (i.e., those that return correct answers only) are well-understood, the dual notion of overapproximations that return complete (but not necessarily sound) answers, and also a more general notion of approximation based on the symmetric difference of query results, are almost unexplored. In fact, the decidability of the basic problems of evaluation, identification, and existence of those approximations, is open.

We develop a connection with existential pebble game tools that allows the systematic study of such problems. In particular, we show that the evaluation and identification of overapproximations can be solved in polynomial time. We also make progress in the problem of existence of overapproximations, showing it to be decidable in 2EXPTIME over the class of acyclic CQs. Furthermore, we look at when overapproximations do not exist, suggesting that this can be alleviated by using a more liberal notion of overapproximation. We also show how to extend our tools to study symmetric difference approximations. We observe that such approximations properly extend under- and over-approximations, settle the complexity of its associated identification problem, and provide several results on existence and evaluation.

1998 ACM Subject Classification H.2.3 Database Management - Query Languages

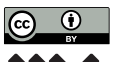
Keywords and phrases conjunctive queries; evaluation; hypertreewidth; approximations; existential pebble game

Digital Object Identifier 10.4230/LIPIcs...

1 Introduction

Context. Due to the growing number of scenarios in which exact query evaluation is infeasible – e.g., when the volume of the data being queried is very large, or when queries are inherently complex – approximate query answering has become an important area of study in databases (see, e.g. [12–14, 21, 25]). Here we focus on approximate query answering for the fundamental class of conjunctive queries (CQs), for which exact evaluation is NP-complete. (Recall that CQ evaluation is the problem of given a CQ q, a database $\mathcal{D}$, and a tuple $\bar{a}$ of constants in $\mathcal{D}$, check if $\bar{a}$ belongs to $q(\mathcal{D})$, the *result* of q over $\mathcal{D}$).

It is known that the complexity of evaluation of a CQ depends on its *degree of acyclicity*, which can be formalized using different notions. One of the most general and well-studied such a notion corresponds to *generalized hypertreewidth* [16]. Notably, the classes of CQs of bounded generalized hypertreewidth can be evaluated in polynomial time (see [15] for a survey). Following recent work on approximate query answering for CQs and some related



query languages [5, 6], we study the process of approximating a CQ as one of bounded generalized hypertreewidth. This provides us with a certificate of efficiency for the cost of evaluating such an approximation. It is worth noticing that our approximations are *static*, in the sense that they depend only on the CQ q and not on the underlying database $\mathcal{D}$. This has clear benefits in terms of the cost of the approximation process, as q is often orders of magnitude smaller than $\mathcal{D}$. Moreover, it allows us to construct a principled approach to CQ approximation based on the well-studied notion of CQ containment [8]. Recall that a CQ q is *contained* in a CQ q', written $q \subseteq q'$, if $q(\mathcal{D}) \subseteq q'(\mathcal{D})$ over each database $\mathcal{D}$. This notion constitutes the theoretical basis for the study of several CQ optimization problems [1].

We denote by $\mathsf{GHW}(k)$ the class of CQs of generalized hypertreewidth at most k, for $k \geq 1$. As mentioned above, we look for an approximation of a CQ q in $\mathsf{GHW}(k)$. A formalization of this notion was first introduced in [4], based on the following partial order $\sqsubseteq_q$ over the set of CQ in $\mathsf{GHW}(k)$: if $q', q'' \in \mathsf{GHW}(k)$, then $q' \sqsubseteq_q q''$ iff over every database $\mathcal{D}$ the symmetric difference between $q(\mathcal{D})$ and $q''(\mathcal{D})$ is contained in the symmetric difference between $q(\mathcal{D})$ and $q'(\mathcal{D})$. Intuitively, this states that q'' is a better $\mathsf{GHW}(k)$-approximation of q than q'. The $\mathsf{GHW}(k)$-approximations of q correspond then to maximal elements with respect to $\sqsubseteq_q$ among a distinguished class of CQs in $\mathsf{GHW}(k)$. Three notions of approximation were introduced in [4], by imposing different "reasonable" conditions on such a class. These are:

- *Underapproximations:* In this case we look for approximations in the set of CQs q' in $\mathsf{GHW}(k)$ that are contained in q, i.e., $q' \subseteq q$. This ensures that the evaluation of such approximations always produce correct (but not necessarily complete) answers to q. A $\mathsf{GHW}(k)$-underapproximation of q is then a CQ q' amongst these CQs that is *maximal* with respect to the partial order defined by $\sqsubseteq_q$. Noticeably, the latter coincides with being maximal with respect to the containment partial order $\subseteq$ among the CQs in $\mathsf{GHW}(k)$ that are contained in q; i.e., no other CQ in such a set strictly contains q'.
- *Overapproximations:* This is the dual notion of underapproximations, in which we look for minimal elements in the class of CQs q' in $\mathsf{GHW}(k)$ that contain q, i.e., $q \subseteq q'$. Hence, $\mathsf{GHW}(k)$-overapproximations produce complete (but not necessarily correct) answers to q.
- *Symmetric difference approximations:* While underappoximations must be contained in the original query, and overapproximations must contain it, this notion does not impose any constraint on approximations with respect to the partial order $\subseteq$. Then a symmetric difference $\mathsf{GHW}(k)$-approximation of q – or simply $\mathsf{GHW}(k)$-Δ-approximation from now on – is a maximal CQ in $\mathsf{GHW}(k)$ with respect to the partial order $\sqsubseteq_q$.

The approximations presented above provide "qualitative" guarantees for evaluation, as they are as close as possible to q among all CQs in $\mathsf{GHW}(k)$ of a certain kind. In particular, under and overapproximations are dual notions which provide lower and upper bounds for the exact evaluation of a CQ, while Δ-approximations can give us useful information when the quality of the result of the under- and overapproximations is poor. Then, in order to develop a robust theory of bounded hypertreewidth static approximations for CQs, it is necessary to have a good understanding of all three notions.

The notion of underapproximation is by now well-understood. Indeed, it is known that for each fixed $k \geq 1$ the $\mathsf{GHW}(k)$-underapproximations have good properties that justify its application: (a) they always exist, and (b) evaluating all $\mathsf{GHW}(k)$-underapproximations of a CQ q over a database $\mathcal{D}$ is a *fixed-parameter tractable* problem [5]. This is an improvement over general CQ evaluation for which the latter is believed not to hold [27]. On the other hand, while $\mathsf{GHW}(k)$-overapproximations and $\mathsf{GHW}(k)$-Δ-approximations were introduced in [4], its theoretical aspects were left almost unexplored since no tools were identified for studying the decidability of basic problems such as:

- *Existence:* Does CQ q have a $\mathsf{GHW}(k)$-overapproximation (or $\mathsf{GHW}(k)$-Δ-approximation)?
- *Identification:* Is q' a $\mathsf{GHW}(k)$-overapproximation (or $\mathsf{GHW}(k)$-Δ-approximation) of q?
- *Evaluation:* Given a CQ q, a database $\mathcal{D}$, and a tuple $\bar{a}$ in $\mathcal{D}$, is it the case that $\bar{a} \in q'(\mathcal{D})$, for some $\mathsf{GHW}(k)$-overapproximation (resp., $\mathsf{GHW}(k)$-Δ-approximation) q' of q?

Partial results were obtained in [4], but based on ad-hoc tools. Also, some CQs have no $\mathsf{GHW}(k)$-overapproximations (in contrast to underapproximations, that always exist), which was seen as a negative result.

Contributions. We develop tools for the systematic study of overapproximations and Δ-approximations. While we mainly focus on the former, we provide a detailed account of how our techniques can be extended to deal with the latter. In the context of $\mathsf{GHW}(k)$-overapproximations, we apply our tools to pinpoint the complexity of evaluation and identification, and make progress in the problem of existence. We also study when overapproximations do not exist and suggest how this can be alleviated. Our contributions are as follows:

1. Link to existential pebble games. We establish a link between $\mathsf{GHW}(k)$-overapproximations and *existential pebble games* [23]. Such games have been used to show that CQs of bounded width can be evaluated efficiently [9, 11]. Using the fact that the existence of winning conditions in the existential pebble game can be checked in PTIME [9], we show that identification and evaluation for $\mathsf{GHW}(k)$-overapproximations are tractable problems.
2. A more liberal notion of overapproximation. We observe that non-existence of overapproximations is due to the fact that in some cases overapproximations require expressing conjunctions of infinitely many atoms. By relaxing our notion, we get that each CQ q has a (potentially infinite) $\mathsf{GHW}(k)$-overapproximation q'. This q' is unique (up to equivalence). Further, it can be evaluated efficiently – in spite of being potentially infinite – by checking a winning condition for the existential k-pebble game on q and $\mathcal{D}$.
3. Existence of overapproximations. It is still useful to check if a CQ q has a *finite* $\mathsf{GHW}(k)$-overapproximation q', and compute it if possible. This might allow to optimize q' before evaluating it. There is also a difference in complexity, as existential pebble game techniques are PTIME-complete in general [22], and thus inherently sequential, while evaluation of CQs in $\mathsf{GHW}(k)$ is highly parallelizable (Gottlob et al. [16]).
   By exploiting automata techniques, we show that checking if a CQ q has a (finite) $\mathsf{GHW}(1)$-overapproximation q' is in 2Exptime. Also, when such q' exists it can be computed in 3Exptime. This is important since $\mathsf{GHW}(1)$ coincides with the well-known class of *acyclic* CQs [29]. If the arity of the schema is fixed, these bounds become Exptime and 2Exptime, respectively. Also, we look at the case of binary schemas, such as the ones used in *graph databases* [3] and *description logics* [2]. In this case, $\mathsf{GHW}(1)$-overapproximations can be computed efficiently via a greedy algorithm. This is optimal, as over ternary schemas we prove an exponential lower bound for the size of $\mathsf{GHW}(1)$-overapproximations. We do not know if the existence problem is decidable for $k > 1$. However, we show that it can be recast as an unexplored boundedness condition for the existential pebble game. Understanding the decidability boundary for such conditions is often difficult [7, 26].

We then move to study $\mathsf{GHW}(k)$-Δ-approximations. We start by showing that they strictly generalize $\mathsf{GHW}(k)$-under and $\mathsf{GHW}(k)$-overapproximations. As for the case of $\mathsf{GHW}(k)$-overapproximations, we provide a link between $\mathsf{GHW}(k)$-Δ-overapproximations and the existential pebble game, and use it to characterize when a CQ q has at least one $\mathsf{GHW}(k)$-Δ-approximation that is neither a $\mathsf{GHW}(k)$-underapproximation nor a $\mathsf{GHW}(k)$-overapproximation (a so-called *incomparable* $\mathsf{GHW}(k)$-Δ-*approximation*). This allows us to

show that the identification problem for such Δ-approximations is coNP-complete. As for the problem of checking for the existence of incomparable $\mathsf{GHW}(k)$-Δ-approximations, we extend our automata techniques to prove that it is in 2EXPTIME for $k = 1$ (and in EXPTIME for fixed-arity schemas). In case such a $\mathsf{GHW}(1)$-Δ-approximation exists, we can evaluate it using a fixed-parameter tractable algorithm. We also provide results on existence and evaluation of infinite incomparable $\mathsf{GHW}(1)$-Δ-approximations.

Organization. Section 2 contains preliminaries. Basic properties of overapproximations are presented in Section 3, while the existence of overapproximations is studied in Section 4. In Section 5 we deal with Δ-approximations, and conclude in Section 6 with final remarks. Due to limited space, several proofs are in the appendix.

2 Preliminaries

Relational databases and homomorphisms. A *relational schema* σ is a finite set of relation symbols, each one of which has an arity $n > 0$. A *database* $\mathcal{D}$ over σ is a finite set of atoms of the form $R(\bar{a})$, where R is a relation symbol in σ of arity n and $\bar{a}$ is an n-tuple of constants. We often abuse notation and write $\mathcal{D}$ also for the set of elements in $\mathcal{D}$.

Let $\mathcal{D}$ and $\mathcal{D}'$ be databases over σ. A *homomorphism* from $\mathcal{D}$ to $\mathcal{D}'$ is a mapping h from $\mathcal{D}$ to $\mathcal{D}'$ such that for every atom $R(\bar{a})$ in $\mathcal{D}$ it is the case that $R(h(\bar{a})) \in \mathcal{D}'$. If $\bar{a}$ and $\bar{b}$ are n-ary tuples ($n \geq 0$) in $\mathcal{D}$ and $\mathcal{D}'$, respectively, we write $(\mathcal{D}, \bar{a}) \to (\mathcal{D}', \bar{b})$ if there is a homomorphism h from $\mathcal{D}$ to $\mathcal{D}'$ such that $h(\bar{a}) = \bar{b}$. Checking if $(\mathcal{D}, \bar{a}) \to (\mathcal{D}', \bar{b})$ is a well-known NP-complete problem.

Conjunctive queries. A *conjunctive query* (CQ) over schema σ is a formula q of the form $\exists \bar{y} \bigwedge_{1 \leq i \leq m} R_i(\bar{x}_i)$, where each $R_i(\bar{x}_i)$ is an atom over σ ($1 \leq i \leq m$). We often write this as $q(\bar{x})$ to denote that $\bar{x}$ are the *free variables* of q, i.e., the ones that are not existentially quantified in $\bar{y}$. If $\bar{x}$ is empty, then q is *boolean*. We define the evaluation of CQs in terms of homomorphisms. Recall that the *canonical database* $\mathcal{D}_q$ of a CQ $q = \exists \bar{y} \bigwedge_{1 \leq i \leq m} R_i(\bar{x}_i)$ consists precisely of the atoms $R_i(\bar{x}_i)$, for $1 \leq i \leq m$. We then define the *result of q over $\mathcal{D}$*, denoted $q(\mathcal{D})$, as the set of all tuples $\bar{a}$ such that $(\mathcal{D}_q, \bar{x}) \to (\mathcal{D}, \bar{a})$. We often do not distinguish between a CQ q and its canonical database $\mathcal{D}_q$ (i.e., we write q for $\mathcal{D}_q$).

Evaluation and tractable classes of CQs. The *evaluation problem for CQs* is as follows: Given a CQ q, a database $\mathcal{D}$, and a tuple $\bar{a}$ in $\mathcal{D}$, is $\bar{a} \in q(\mathcal{D})$? Since this problem corresponds to checking if $(q, \bar{x}) \to (\mathcal{D}, \bar{a})$, it is NP-complete [8]. This led to a flurry of activity for finding classes of CQs for which evaluation is tractable.

Here we deal with one of the most studied such classes: CQs of bounded *generalized hypertreewidth* [16], also called *coverwidth* [9]. We adopt the definition of [9] which is better suited for working with non-boolean queries. A *tree decomposition* of a CQ $q = \exists \bar{y} \bigwedge_{1 \leq i \leq m} R_i(\bar{x}_i)$ is a pair (T, χ), where T is a tree and χ is a mapping that assigns a subset of the existentially quantified variables in $\bar{y}$ to each node $t \in T$, such that:

1. For each $1 \leq i \leq m$, the variables in $\bar{x}_i \cap \bar{y}$ are contained in $\chi(t)$, for some $t \in T$.
2. For each variable y in $\bar{y}$, the set of nodes $t \in T$ for which y occurs in $\chi(t)$ is connected.

The *width* of node t in (T, χ) is the minimal size of an $I \subseteq \{1, \ldots, m\}$ such that $\bigcup_{i \in I} \bar{x}_i$ covers $\chi(t)$. The width of (T, χ) is the maximal width of the nodes of T. The *generalized hypertreewidth* of q is the minimum width of its tree decompositions.

For a fixed $k \geq 1$, we denote by $\mathsf{GHW}(k)$ the class of CQs of generalized hypertreewidth at most k. The CQs in $\mathsf{GHW}(k)$ can be evaluated in polynomial time; see [15].

Containment of CQs. A CQ q is *contained* in a CQ q', written as $q \subseteq q'$, if $q(\mathcal{D}) \subseteq q'(\mathcal{D})$ over every database $\mathcal{D}$. Two CQs q and q' are *equivalent*, denoted $q \equiv q'$, if $q \subseteq q'$ and $q' \subseteq q$.

It is known that CQ containment and CQ evaluation are, essentially, the same problem [8]. In particular, let $q(\bar{x})$ and $q'(\bar{x})$ be CQs. Then:

$$q \subseteq q' \iff \bar{x} \in q'(\mathcal{D}_q) \iff (\mathcal{D}_{q'}, \bar{x}) \to (\mathcal{D}_q, \bar{x}). \tag{1}$$

Thus, $q \subseteq q'$ and $(q', \bar{x}) \to (q, \bar{x})$ (i.e., $(\mathcal{D}_{q'}, \bar{x}) \to (\mathcal{D}_q, \bar{x})$) are used interchangeably.

Approximations of CQs. Fix $k \geq 1$. Let q be a CQ. The approximations of q in $\mathsf{GHW}(k)$ are defined with respect to a partial order $\sqsubseteq_q$ over the set of CQs in $\mathsf{GHW}(k)$. Formally, for any two CQs q', q'' in $\mathsf{GHW}(k)$ we have:

$$q' \sqsubseteq_q q'' \iff \Delta(q(\mathcal{D}), q''(\mathcal{D})) \subseteq \Delta(q(\mathcal{D}), q'(\mathcal{D})), \text{ for every database } \mathcal{D},$$

where $\Delta(A, B)$ denotes the symmetric difference between sets A and B. Thus, $q' \sqsubseteq_q q''$, whenever the "error" of q'' with respect to q – measured in terms of the symmetric difference between $q''(\mathcal{D})$ and $q(\mathcal{D})$ – is contained in that of q' for each database $\mathcal{D}$. As usual, we write $q' \sqsubset_q q''$ if $q' \sqsubseteq_q q''$ but $q'' \not\sqsubseteq_q q'$.

The approximations of q in $\mathsf{GHW}(k)$ always correspond to maximal elements, with respect to the partial order $\sqsubseteq_q$, over a class of CQs in $\mathsf{GHW}(k)$ that satisfies certain conditions. The following three basic notions of approximation were identified in [4]:

- Underapproximations: Let q, q' be CQs such that $q' \in \mathsf{GHW}(k)$. Then q' is a $\mathsf{GHW}(k)$-*underapproximation* of q if it is maximal, with respect to $\sqsubseteq_q$, among all CQs in $\mathsf{GHW}(k)$ that are contained in q. That is:

  $$q' \subseteq q, \text{ and there is no CQ } q'' \in \mathsf{GHW}(k) \text{ such that } q'' \subseteq q \text{ and } q' \sqsubset_q q''.$$

  In particular, the $\mathsf{GHW}(k)$-underapproximations of q produce correct (but not necessarily complete) answers with respect to q over every database $\mathcal{D}$.
- Overapproximations: Analogously, q' is a $\mathsf{GHW}(k)$-*overapproximation* of q if it is maximal, with respect to $\sqsubseteq_q$, among all CQs in $\mathsf{GHW}(k)$ that contain q. That is, the $\mathsf{GHW}(k)$-overapproximations of q produce complete (but not necessarily correct) answers with respect to q over every database $\mathcal{D}$.
- Δ-approximations: In this case we impose no restriction on q'. That is, q' is a $\mathsf{GHW}(k)$-Δ-*approximation* of q if it is maximal with respect to the partial order $\sqsubseteq_q$, i.e., there is no $q'' \in \mathsf{GHW}(k)$ such that $q' \sqsubset_q q''$.

Underapproximations and overapproximations admit an equivalent, but arguably simpler characterization as maximal (resp., minimal) elements, with respect to the containment partial order $\subseteq$, among all CQs in $\mathsf{GHW}(k)$ that are contained in q (resp., contain q):

▶ **Proposition 1.** [4] *Fix $k \geq 1$. Let q, q' be CQs such that $q' \in \mathsf{GHW}(k)$. Then:*

- *q' is a $\mathsf{GHW}(k)$-underapproximation of q iff $q' \subseteq q$ and there is no CQ $q'' \in \mathsf{GHW}(k)$ such that $q' \subset q'' \subseteq q$.*
- *q' is a $\mathsf{GHW}(k)$-overapproximation of q iff $q \subseteq q'$ and there is no CQ $q'' \in \mathsf{GHW}(k)$ such that $q \subseteq q'' \subset q'$.*

As mentioned before, $\mathsf{GHW}(k)$-underapproximations are by now well-understood. We concentrate on $\mathsf{GHW}(k)$-overapproximations and $\mathsf{GHW}(k)$-Δ-approximations in this paper. We start by studying the former.

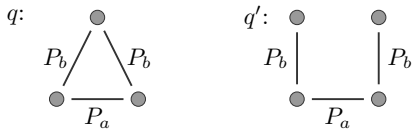


Figure 1 The CQ q and its $\mathsf{GHW}(1)$-overapproximation q' from Example 1.

3 Overapproximations

Recall that $\mathsf{GHW}(k)$-overapproximations are minimal elements (in terms of $\subseteq$) in the set of CQs in $\mathsf{GHW}(k)$ that contain q. We show an example of a $\mathsf{GHW}(1)$-approximation below:

▶ **Example 1.** Figure 1 shows a CQ q and its $\mathsf{GHW}(1)$-overapproximation q'. The schema consists of binary symbols P_a and P_b. Dots represent variables, and an edge labeled P_a between x and y represents the presence of atoms $P_a(x, y)$ and $P_a(y, x)$. (Same for P_b). All variables are existentially quantified. Clearly, $q \subseteq q'$ (as $q' \to q$). In addition, there is no CQ $q'' \in \mathsf{GHW}(1)$ such that $q \subseteq q'' \subset q'$. We provide an explanation for this later. ◀

We start in Section 3.1 by stating some basic properties on existence and uniqueness of $\mathsf{GHW}(k)$-overapproximations. Later in Section 3.2 we establish a connection between $\mathsf{GHW}(k)$-overapproximations and the existential pebble game, which allows us to show that both the identification and evaluation problems for $\mathsf{GHW}(k)$-overapproximations are tractable. Finally, in Section 3.4 we look at the case when $\mathsf{GHW}(k)$-overapproximations do not exist, and suggest how this can be alleviated by allowing infinite overapproximations.

3.1 Existence and uniqueness of overapproximations

Existence of overapproximations is not a general phenomenon as shown in [4]. In fact, for every $k > 1$ there is a boolean CQ q in $\mathsf{GHW}(k)$ that has no $\mathsf{GHW}(1)$-overapproximation. Using the characterization given later in Theorem 13, we can strengthen this further:

▶ **Proposition 2.** *For each $k > 1$, there is a Boolean CQ $q \in \mathsf{GHW}(k)$ without $\mathsf{GHW}(\ell)$-overapproximations for any $1 \leq \ell < k$.*

Figure 2 depicts examples of CQs in $\mathsf{GHW}(k)$, for $k = 2$ and $k = 3$, respectively, without $\mathsf{GHW}(\ell)$-overapproximations for any $1 \leq \ell < k$.

Interestingly, when $\mathsf{GHW}(k)$-overapproximations do exist, they are unique (up to equivalence). This follows since, in this case, $\mathsf{GHW}(k)$-overapproximations are not only the minimal elements, but also the lower bounds of the set of CQs in $\mathsf{GHW}(k)$ that contain q:

▶ **Proposition 3.** *Let q, q' be CQs such that $q' \in \mathsf{GHW}(k)$. The following are equivalent:*

1. *q' is a $\mathsf{GHW}(k)$-overapproximation of q.*
2. *(i) $q \subseteq q'$, and (ii) for every CQ $q'' \in \mathsf{GHW}(k)$, it is the case that $q \subseteq q''$ implies $q' \subseteq q''$.*

Proof. We only prove the nontrivial direction (1) $\Rightarrow$ (2). By contradiction, suppose there is a CQ $q'' \in \mathsf{GHW}(k)$ such that $q \subseteq q''$ but $q' \not\subseteq q''$. Let $(q' \wedge q'')$ be the *conjunction* of q' and q'', i.e., the CQ which is obtained by first renaming each existentially quantified variable in q' and q'' with a different fresh variable, and then taking the conjunction of the atoms in q' and q''. It is easy to see that $(q' \wedge q'')$ is in $\mathsf{GHW}(k)$. Also, by the definition of $(q' \wedge q'')$ we have that $q \subseteq (q' \wedge q'') \subseteq q'$. But q' is a $\mathsf{GHW}(k)$-overapproximation of q, and thus $q' \subseteq (q' \wedge q'')$. Clearly, on the other hand, $(q' \wedge q'') \subseteq q''$, and hence $q' \subseteq q''$. This is a contradiction. ◀

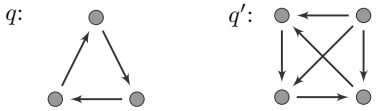


Figure 2 The CQ q is in GHW(2) but has no GHW(1)-overapproximations, while q' is in GHW(3) but has no GHW(ℓ)-overapproximations for $\ell \in \{1, 2\}$.

As a corollary, we immediately obtain the following:

▶ **Corollary 2.** *If a CQ q has* GHW(k)*-overapproximations q_1 and q_2, then $q_1 \equiv q_2$.*

The previous results show the stark difference between GHW(k)-overapproximations and GHW(k)-underapproximations: While GHW(k)-overapproximations do not necessarily exist, but when they do they are unique, GHW(k)-underapproximations always exist but there can be exponentially many incomparable ones [5].

3.2 A link with the existential pebble game

We characterize GHW(k)-overaproximations in terms of the existential pebble game. We use a version of such a game, known as *existential cover game*, that is tailored for CQs of bounded generalized hypertreewidth [9]. Let $k \geq 1$. The existential k-cover game is played by *Spoiler* and *Duplicator* on pairs $(\mathcal{D}, \bar{a})$ and $(\mathcal{D}', \bar{b})$, where $\mathcal{D}$ and $\mathcal{D}'$ are databases and $\bar{a}$ and $\bar{b}$ are n-ary ($n \geq 0$) tuples over $\mathcal{D}$ and $\mathcal{D}'$, respectively. The game proceeds in rounds. In each round, Spoiler places (resp., removes) a pebble on (resp., from) an element of $\mathcal{D}$, and Duplicator responds by placing (resp., removing) its corresponding pebble on an element of (resp., from) $\mathcal{D}'$. The number of pebbles is not bounded, but Spoiler is constrained as follows: At any round p of the game, if $c_1, \ldots, c_l$ ($l \leq p$) are the elements marked by Spoiler pebbles in $\mathcal{D}$, there must be at most k atoms in $\mathcal{D}$ that contain all such elements (this is why the game is called k-cover, as pebbled elements are *covered* by such k atoms).

Duplicator wins if she has a *winning strategy*, i.e., she can indefinitely continue playing the game in such way that after each round, if $c_1, \ldots, c_\ell$ are the elements that are marked by Spoiler's pebbles in $\mathcal{D}$ and $d_1, \ldots, d_\ell$ are the elements marked by the corresponding pebbles of Duplicator in $\mathcal{D}'$, then $\big((c_1, \ldots, c_k, \bar{a}), (d_1, \ldots, d_k, \bar{b})\big)$ is a *partial homomorphism* from $\mathcal{D}$ to $\mathcal{D}'$. That is, for every atom $R(\bar{c}) \in \mathcal{D}$, where each element c of $\bar{c}$ appears in $(c_1, \ldots, c_k, \bar{a})$, it is the case that $R(\bar{d}) \in \mathcal{D}'$, where $\bar{d}$ is the tuple obtained from $\bar{c}$ by replacing each element c of $\bar{c}$ by its corresponding element d in $(d_1, \ldots, d_k, \bar{b})$. In such case, we write $(\mathcal{D}, \bar{a}) \to_k (\mathcal{D}', \bar{b})$.

Notice that $\to_k$ "approximates" $\to$ as follows: $\to \subset \ \ldots \ \subset \to_{k+1} \subset \to_k \subset \ \ldots \ \subset \to_1$. These approximations are convenient complexity-wise: Checking if $(\mathcal{D}, \bar{a}) \to (\mathcal{D}', \bar{b})$ is NP-complete, but $(\mathcal{D}, \bar{a}) \to_k (\mathcal{D}', \bar{b})$ can be solved efficiently.

▶ **Proposition 4.** [9] *Fix $k \geq 1$. Checking $(\mathcal{D}, \bar{a}) \to_k (\mathcal{D}', \bar{b})$ is solvable in polynomial time.*

Moreover, there is a connection between $\to_k$ and the evaluation of CQs in GHW(k) that we heavily exploit in our work:

▶ **Proposition 5.** [9] *Fix $k \geq 1$. Then $(\mathcal{D}, \bar{a}) \to_k (\mathcal{D}', \bar{b})$ iff for each CQ $q(\bar{x})$ in* GHW(k) *we have that if $(q, \bar{x}) \to (\mathcal{D}, \bar{a})$ then $(q, \bar{x}) \to (\mathcal{D}', \bar{b})$.*

In particular, if $q(\bar{x}) \in$ GHW(k) then for every $\mathcal{D}$ and $\bar{a}$:

$$\bar{a} \in q(\mathcal{D}) \quad \Longleftrightarrow \quad (q, \bar{x}) \to (\mathcal{D}, \bar{a}) \quad \Longleftrightarrow \quad (q, \bar{x}) \to_k (\mathcal{D}, \bar{a}). \tag{2}$$

That is, the "approximation" of $\to$ provided by $\to_k$ is sufficient for evaluating CQs in $\mathsf{GHW}(k)$. Together with Proposition 4, this proves that CQs in $\mathsf{GHW}(k)$ can be evaluated efficiently.

The characterization. Existential cover games can be applied to obtain a semantic characterization of $\mathsf{GHW}(k)$-overapproximations:

▶ **Theorem 3.** *Fix $k \geq 1$. Let q, q' be CQs with $q' \in \mathsf{GHW}(k)$. Then $q'(\bar{x})$ is the $\mathsf{GHW}(k)$-overapproximation of $q(\bar{x})$ iff $(q', \bar{x}) \to_k (q, \bar{x})$ and $(q, \bar{x}) \to_k (q', \bar{x})$.*

Proof. Assume that $q'(\bar{x})$ is the $\mathsf{GHW}(k)$-overapproximation of $q(\bar{x})$. Then $(q', \bar{x}) \to (q, \bar{x})$, and thus $(q', \bar{x}) \to_k (q, \bar{x})$ from Equation (2). We prove now that $(q, \bar{x}) \to_k (q', \bar{x})$. From Proposition 5, we need to prove that if $q''(\bar{x})$ is a CQ in $\mathsf{GHW}(k)$ such that $(q'', \bar{x}) \to (q, \bar{x})$, then also $(q'', \bar{x}) \to (q', \bar{x})$. This follows directly from Proposition 3.

Assume now that $(q', \bar{x}) \to_k (q, \bar{x})$ and $(q, \bar{x}) \to_k (q', \bar{x})$. Since q' is in $\mathsf{GHW}(k)$, we have that $q \subseteq q'$ from Equation (2). From Proposition 5, if $q \subseteq q''$ and $q'' \in \mathsf{GHW}(k)$ then $q' \subseteq q''$, i.e., there is no q'' in $\mathsf{GHW}(k)$ such that $q \subseteq q'' \subset q'$. ◀

▶ **Example 4.** (Example 1 cont.) It is now easy to see that the CQ q' in Figure 1 is a $\mathsf{GHW}(1)$-overapproximation of q. In fact, since $q' \to q$, we only need to show that $q \to_1 q'$. The latter is simple and left to the reader. ◀

Next we show that this characterization allows us to show that the identification and evaluation problems for $\mathsf{GHW}(k)$-overapproximations can be solved in polynomial time.

3.3 Identification and evaluation of $\mathsf{GHW}(k)$-overapproximations

A direct corollary of Proposition 4 and Theorem 3 is that the *identification* problem for $\mathsf{GHW}(k)$-overapproximations is in polynomial time:

▶ **Corollary 5.** *Fix $k \geq 1$. Given CQs q, q' such that $q' \in \mathsf{GHW}(k)$, checking if q' is the $\mathsf{GHW}(k)$-overapproximation of q can be solved in polynomial time.*

This corresponds to a *promise version* of the problem, as it is given to us that q' is in fact in $\mathsf{GHW}(k)$. Checking the latter is NP-complete [17].

Assume now that we are given the *promise* that q has a $\mathsf{GHW}(k)$-overapproximation q' (but q' itself is not given). How hard is it to evaluate q' over a database $\mathcal{D}$? We could try to compute q', but so far we have no techniques to do that. Notably, we can use existential cover games to show that $\mathsf{GHW}(k)$-overapproximations can be evaluated efficiently, without even computing them. This is based on the next result, which states that evaluating q' over $\mathcal{D}$ boils down to checking $(q, \bar{x}) \to_k (\mathcal{D}, \bar{a})$ for the tuples $\bar{a}$ over $\mathcal{D}$.

▶ **Theorem 6.** *Fix $k \geq 1$. Let $q(\bar{x})$ be a CQ with a $\mathsf{GHW}(k)$-overapproximation $q'(\bar{x})$. Then for every $\mathcal{D}$ and $\bar{a}$:*

$$\bar{a} \in q'(\mathcal{D}) \iff (q', \bar{x}) \to (\mathcal{D}, \bar{a}) \iff (q, \bar{x}) \to_k (\mathcal{D}, \bar{a}).$$

Proof. Assume first that $(q, \bar{x}) \to_k (\mathcal{D}, \bar{a})$. Since q' is a $\mathsf{GHW}(k)$-overapproximation of q, we have that $(q', \bar{x}) \to (q, \bar{x})$. By composition then, $(q', \bar{x}) \to_k (\mathcal{D}, \bar{a})$. But $q' \in \mathsf{GHW}(k)$, and thus $(q', \bar{x}) \to (\mathcal{D}, \bar{a})$ from Equation (2). Assume now that $(q', \bar{x}) \to (\mathcal{D}, \bar{a})$. From Theorem 3, we have that $(q, \bar{x}) \to_k (q', \bar{x})$, and by composition, $(q, \bar{x}) \to_k (\mathcal{D}, \bar{a})$ holds. ◀

As a corollary to Theorem 6 and Proposition 4 we obtain:

▶ **Corollary 7.** *Fix $k \geq 1$. Checking if $\bar{a} \in q'(\mathcal{D})$, given a CQ q that has a $\mathsf{GHW}(k)$-overapproximation q', a database $\mathcal{D}$, and a tuple $\bar{a}$ in $\mathcal{D}$, can be solved in polynomial time by checking if $(q, \bar{x}) \to_k (\mathcal{D}, \bar{a})$. Moreover, this can be done without even computing q'.*

3.4 More liberal GHW(k)-overapproximations

CQs may not have GHW(k)-overapproximations, for some $k \geq 1$. We observe in this section that this anomaly can be solved by extending the language of queries over which overapproximations are to be found.

An *infinite* CQ is as a finite one, save that now the number of atoms can be countable. We assume that there are finitely many free variables in an infinite CQ. The evaluation of an infinite CQ $q(\bar{x})$ over a database $\mathcal{D}$ is defined analogously to the evaluation of a finite one. We write $\mathsf{GHW}(k)^\infty$ for the class of all CQs, finite and countably infinite ones, of generalized hypertreewidth at most k. The next result states a crucial relationship between the existential k-cover game and the class $\mathsf{GHW}(k)^\infty$:

▶ **Lemma 8.** *Fix $k \geq 1$. For every CQ q there is a q' in $\mathsf{GHW}(k)^\infty$ such that for every database $\mathcal{D}$ and tuple $\bar{a}$ of constants in $\mathcal{D}$:*

$$\bar{a} \in q'(\mathcal{D}) \quad \iff \quad (q', \bar{x}) \to (\mathcal{D}, \bar{a}) \quad \iff \quad (q, \bar{x}) \to_k (\mathcal{D}, \bar{a}).$$

This holds even for countably infinite databases $\mathcal{D}$.

We omit the proof of this result, as it follows from techniques in [23]. The basic idea is that q' has an (infinite) generalized hypertree decomposition of width k that represents all possible strategies of Spoiler in the existential k-cover game played from q.

Since we now deal with infinite CQs and databases, we cannot apply Proposition 5 directly in our analysis of $\mathsf{GHW}(k)^\infty$-overapproximations. Instead, we use the following suitable reformulation of it, which we obtain by a straightforward inspection of its proof:

▶ **Proposition 6.** *Fix $k \geq 1$. Consider countably infinite databases $\mathcal{D}$ and $\mathcal{D}'$. Then $(\mathcal{D}, \bar{a}) \to_k (\mathcal{D}', \bar{b})$ iff for each CQ $q(\bar{x})$ in $\mathsf{GHW}(k)^\infty$, if $(q, \bar{x}) \to (\mathcal{D}, \bar{a})$ then $(q, \bar{x}) \to (\mathcal{D}', \bar{b})$.*

GHW(k)$^\infty$-overapproximations. We expand the notion of overapproximation by allowing infinite CQs. Let $q' \in \mathsf{GHW}(k)^\infty$. Then q' is a $\mathsf{GHW}(k)^\infty$-overapproximation of CQ q, if $q \subseteq q'$ and there is no $q'' \in \mathsf{GHW}(k)^\infty$ such that $q \subseteq q'' \subset q'$. (Here, $\subseteq$ is still defined with respect to finite databases only). In $\mathsf{GHW}(k)^\infty$, we can provide each CQ q an overapproximation:

▶ **Theorem 9.** *Fix $k \geq 1$. For every CQ q there is a CQ in $\mathsf{GHW}(k)^\infty$ that is a $\mathsf{GHW}(k)^\infty$-overapproximation of q.*

Proof. We prove that q', as given in Lemma 8, is a $\mathsf{GHW}(k)^\infty$-overapproximation of q. Notice that $(q', \bar{x}) \to (q, \bar{x})$ (by choosing $(\mathcal{D}, \bar{a})$ as $(q, \bar{x})$ in Lemma 8). Therefore, $q \subseteq q'$ since this direction of Equation (1) continues to hold for countably infinite CQs. Observe now that $(q, \bar{x}) \to_k (q', \bar{x})$ (by choosing $(\mathcal{D}, \bar{a})$ as $(q', \bar{x})$ in Lemma 8). Proposition 6 tells us that for every $q''(\bar{x})$ in $\mathsf{GHW}(k)^\infty$, if $(q'', \bar{x}) \to (q, \bar{x})$ then $(q'', \bar{x}) \to (q', \bar{x})$. But then $q \subseteq q''$ implies that $q' \subseteq q''$, since Equation (1) continues to hold if q (but not necessarily q') is finite. Thus, q' is a $\mathsf{GHW}(k)^\infty$-overapproximation of q. ◀

Despite the non-computable nature of $\mathsf{GHW}(k)^\infty$-overapproximations, we get from Proposition 4 and the proof of Theorem 9 that they can be evaluated efficiently:

▶ **Corollary 10.** *Fix $k \geq 1$. Checking whether $\bar{a} \in q'(\mathcal{D})$, given a CQ q with $\mathsf{GHW}(k)^\infty$-overapproximation q', a database $\mathcal{D}$, and a tuple $\bar{a}$ in $\mathcal{D}$, boils down to checking if $(q, \bar{x}) \to_k (\mathcal{D}, \bar{a})$, and thus it can be solved in polynomial time.*

On the power of the existential cover game. Some works "overestimate" the evaluation of CQs, weakening the quality of the answer in favor of efficiency. A paradigmatic

example is [12], which redefines the semantics of CQs in terms of *simulations* (a well-known overestimation of the notion of homomorphism [28]). For schemas of node-labeled graphs, as the ones considered in [12], the notion of q being *simulated* in $\mathcal{D}$ coincides with $q \to_1 \mathcal{D}$. Together with Corollary 10, this provides a theoretical justification for such a simulation-based semantics: It corresponds to evaluating a CQ in $\mathsf{GHW}(1)^\infty$, but not an arbitrary one. Indeed, it corresponds to its $\mathsf{GHW}(1)^\infty$-overapproximation.

Our results also highlight an important property of the existential cover game in terms of its power for evaluating CQs. As expressed in Equation (2), for each CQ q in $\mathsf{GHW}(k)$ we have that computing $q(\mathcal{D})$ boils down to finding the tuples $\bar{a} \in \mathcal{D}$ such that $(q, \bar{x}) \to_k (\mathcal{D}, \bar{a})$. For other CQs, on the other hand, determining such $\bar{a}$'s provides an "overestimation" of $q(\mathcal{D})$. Interestingly, Corollary 10 states that such an overstimation is not an arbitrary one; in fact, it corresponds to the evaluation of the $\mathsf{GHW}(k)^\infty$-approximation q' of q over $\mathcal{D}$.

4 Deciding existence of $\mathsf{GHW}(k)$-overapproximations

CQs always have $\mathsf{GHW}(k)^\infty$-overapproximations, but not necessarily finite ones. Here we study when a CQ q has one such a finite overapproximation. We start with the case $k = 1$, which we show to be decidable in 2Exptime. For $k > 1$ we leave the decidability open, but provide some explanation to where the difficulty lies.

4.1 The acyclic case

We start with the case of $\mathsf{GHW}(1)$-overapproximations. Recall that $\mathsf{GHW}(1)$ is an important class, as it consists precisely of the well-known acyclic CQs. Our main result is the following:

▶ **Theorem 11.** *There is a* 2Exptime *algorithm that checks if a CQ q has a $\mathsf{GHW}(1)$-overapproximation and, if one exists, it computes one in triple-exponential time.*

If the arity of the schema is fixed, there is an Exptime *algorithm that does this and computes a $\mathsf{GHW}(1)$-overapproximation of q in double-exponential time.*

We sketch the proof for nonfixed arities. From a CQ q we build a *two-way alternating tree automaton* [10], or 2ATA, $\mathcal{A}_q$, such that the language $L(\mathcal{A}_q)$ of trees accepted by $\mathcal{A}_q$ is nonempty iff q has a $\mathsf{GHW}(1)$-overapproximation. Intuitively, $\mathcal{A}_q$ accepts those trees that encode a $\mathsf{GHW}(1)$-overapproximation q' of q. Formally:

▶ **Proposition 7.** *There exists a double-exponential time algorithm that takes as input a CQ q and returns a 2ATA $\mathcal{A}_q$ with exponentially many states, such that q has a $\mathsf{GHW}(1)$-overapproximation iff $L(\mathcal{A}_q) \neq \emptyset$. Furthermore, from every tree T in $L(\mathcal{A}_q)$ one can construct in polynomial time a $\mathsf{GHW}(1)$-overapproximation of q.*

Proof sketch. For simplicity we assume that q is Boolean. Before describing the construction of $\mathcal{A}_q$, we explain how input trees for $\mathcal{A}_q$ encode CQs in $\mathsf{GHW}(1)$. To this end let q' be a CQ in $\mathsf{GHW}(1)$ and $(T_{q'}, \chi)$ a tree decomposition of q'. The CQ q' can have unbounded many variables. Yet, in each node of $T_{q'}$ at most r variables appear, where r is the maximum arity of an atom in q. Thus, by reusing variables, $(T_{q'}, \chi)$ can be encoded by using $2r$ variables in such a way that it can then be decoded, i.e. a variable name u_i is used in two neighboring nodes v and v' of the encoding iff the corresponding variables of the tree decomposition also occur in neighboring nodes. The encoding $\mathsf{Enc}(T_{q'}, \chi)$ of $(T_{q'}, \chi)$ is thus a tree labeled by (a) the variables $\{u_1, \ldots, u_{2r}\}$ as described, and (b) the atoms of q' covered by those variables.

The 2ATA $\mathcal{A}_q$ checks that the CQ q' encoded by $T' = \mathsf{Enc}(T_{q'}, \chi)$ is a $\mathsf{GHW}(1)$-overapproximation of q. From Theorem 3, we need to check: (1) $q' \to_1 q$, and (2) $q \to_1 q'$.

The 2ATA $\mathcal{A}_q$ will be defined as the intersection of 2ATAs $\mathcal{A}_1$ and $\mathcal{A}_2$, that check conditions (1) and (2), respectively. Condition (1) is equivalent to $q' \to q$ (since $q' \in \mathsf{GHW}(1)$). A simple standard construction yields a 2ATA $\mathcal{A}_1$ that checks that such a condition holds. In particular, $\mathcal{A}_1$ requires no alternation and has at most exponentially many states.

We now sketch how the automaton $\mathcal{A}_2$ works. First, $q \to_1 q'$ can be restated as Duplicator having a *compact winning strategy* [9] as follows. A 1-*union* of q is a set of variables that appears exactly in an atom of q. Then $q \to_1 q'$ iff there is a family $\mathcal{F}$ of partial homomorphisms from q to q' such that: (a) The domain of each $f \in \mathcal{F}$ is a 1-union of q, and (b) If U and U' are 1-unions of q, then each $f \in \mathcal{F}$ with domain U can be *extended* to U', i.e., there is $f' \in \mathcal{F}$ with domain U' such that $f(x) = f'(x)$ for every $x \in U \cap U'$.

The 2ATA $\mathcal{A}_2$ assumes an annotation of $T' = \mathsf{Enc}(T_{q'}, \chi)$ that encodes the intended strategy $\mathcal{F}$. This annotation labels each node t' of T' by the set of partial mappings from q to q' whose domain is a 1-union of q, and whose range is contained in the variables from $\{u_1, \ldots, u_{2r}\}$ labeling t'. It can be easily checked from the labelings of T' if each mapping in this annotation is a partial homomorphism. To check condition (2), the 2ATA $\mathcal{A}_2$ makes a universal transition for each pair (U, U') of 1-unions and partial mapping g with domain U annotating a node t' of T'. Then it checks the existence of a node t' in T' that is annotated with a mapping g' that extends g to U'. The latter means that, for each $x \in U \cap U'$, both $g(x)$ and $g'(x)$ are the same variable of q', that is, $g(x)$ and $g(x')$ are connected occurrences of the same variable in $\{u_1, \ldots, u_{2r}\}$. Thus to check the consistency of g and g', the automaton can store the variables in $\{g(x) \mid x \in U \cap U'\}$, and check that these are present in the label of each node guessed before reaching t'. As this is a polynomial amount of information, $\mathcal{A}_2$ can be implemented using exponentially many states. ◀

It is easy to see how Theorem 11 follows from Proposition 7. Checking if a CQ q has a $\mathsf{GHW}(1)$-overapproximation amounts to checking if $L(\mathcal{A}_q) \neq \emptyset$. The latter can be done in exponential time in the number of states of $\mathcal{A}_q$ [10], and thus in double-exponential time in the size of q. If $L(\mathcal{A}_q) \neq \emptyset$, one can construct a tree $T \in L(\mathcal{A}_q)$ in double-exponential time in the size of $\mathcal{A}_q$, and thus in triple-exponential time in the size of q. From T one then gets in polynomial time (i.e., in 3EXPTIME in the size of q) a $\mathsf{GHW}(1)$-overapproximation of q.

The case of binary schemas. For schemas of arity two the existence and computation of $\mathsf{GHW}(1)$-overapproximations can be solved in polynomial time. This is of practical importance since data models such as *graph databases* [3] and description logic *ABoxes* [2] can be represented using schemas of this kind. Note that in this context $\mathsf{GHW}(1)$ coincides with the class of CQs of treewidth one [11]. Then:

▶ **Theorem 12.** *There is a* PTIME *algorithm that checks if a CQ q over a schema of maximum arity two has a $\mathsf{GHW}(1)$-overapproximation q', and computes such q' if it exists.*

We relegate the proof of this result to the appendix, as it is quite involved. It requires, in particular, the application of tools different from the ones based on tree automata used in the proof of Theorem 11.

Size of overapproximations. Over binary schemas $\mathsf{GHW}(1)$-overapproximations are of polynomial size. This is optimal as over schemas of arity three there is an exponential lower bound for the size of $\mathsf{GHW}(1)$-overapproximations:

▶ **Proposition 8.** *There is a schema σ with a single ternary relation symbol and a family $(q_n)_{n\geq 1}$ of Boolean CQs over σ, such that (1) q_n is of size $O(n)$, and (2) the size of every $\mathsf{GHW}(1)$-overapproximation of q_n is $\Omega(2^n)$.*

Proof. The CQ q_n contains the atoms $R(x_0, x_1^1, x_1^2)$, $R(x_0, x_1^2, x_1^1)$, as well as $R(x_i^j, x_{i+1}^1, x_{i+1}^2)$ and $R(x_i^j, x_{i+1}^2, x_{i+1}^1)$, for each $1 \leq i \leq n-1$ and $j \in \{1, 2\}$. Consider now the CQ q_n' with the

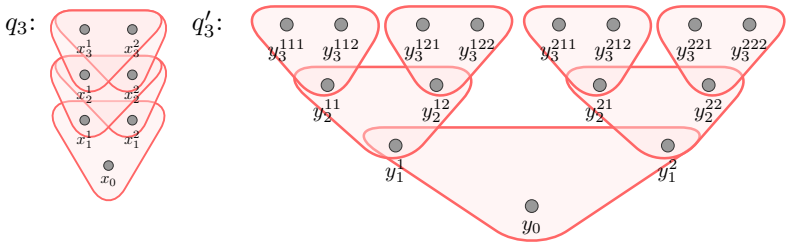


Figure 3 Illustration of the CQs q_3 and q'_3 from Proposition 8. Each triple of variables represents two atoms in the query; e.g., $\{y_0, y_1^1, y_1^2\}$ represents atoms $R(y_0, y_1^1, y_1^2)$ and $R(y_0, y_1^2, y_1^1)$ in q'_3.

atoms $R(y_0, y_1^1, y_1^2)$, $R(y_0, y_1^2, y_1^1)$, as well as $R(y^w_{|w|}, y^{w1}_{|w|+1}, y^{w2}_{|w|+1})$ and $R(y^w_{|w|}, y^{w2}_{|w|+1}, y^{w1}_{|w|+1})$, for each word w over $\{1,2\}$ of length $1 \leq |w| \leq n-1$. Figure 3 depicts q_3 and q'_3.

Clearly, the mapping $h : q'_n \to q_n$ defined as $h(y_0) = x_0$ and $h(y^{wj}_{|w|+1}) = x^j_{|w|+1}$, for each word w over $\{1,2\}$ of length $0 \leq |w| \leq n-1$ and $j \in \{1,2\}$, is a homomorphism. We now show that $q_n \to_1 q'_n$ by building a compact winning strategy for Duplicator (see the proof of Proposition 7) which basically "inverts" the homomorphism h. It contains: (a) Partial homomorphisms $(x_0, x_1^1, x_1^2) \to (y_0, y_1^1, y_1^2)$ and $(x_0, x_1^1, x_1^2) \to (y_0, y_1^2, y_1^1)$, and (b) for each word w over $\{1,2\}$ of length $1 \leq |w| \leq n-1$ and $j \in \{1,2\}$, the partial homomorphisms $(x^j_{|w|}, x^1_{|w|+1}, x^2_{|w|+1}) \to (y^w_{|w|}, y^{w1}_{|w|+1}, y^{w2}_{|w|+1})$ and $(x^j_{|w|}, x^1_{|w|+1}, x^2_{|w|+1}) \to (y^w_{|w|}, y^{w2}_{|w|+1}, y^{w1}_{|w|+1})$. It can be seen that this is a winning strategy for Duplicator.

Observe that the size of q'_n is $\Omega(2^n)$. A straightforward case-by-case analysis shows that q'_n is a *core* [8,19], i.e., there is no homomorphism from q'_n to a *proper* subset of its atoms. We claim that q'_n is the smallest $\mathsf{GHW}(1)$-overapproximation of q_n, from which the proposition follows. Assume, towards a contradiction, that q' is a $\mathsf{GHW}(1)$-overapproximation of q_n with fewer atoms than q'_n. Then, by Corollary 2, we have that $q'_n \equiv q'$. Composing the homomorphisms $h_1 : q'_n \to q'$ and $h_2 : q' \to q'_n$ yields a homomorphism from q'_n to a proper subset of the atoms of q'_n. This is a contradiction since q'_n is a core. ◀

4.2 Beyond acyclicity

Theorem 6 characterizes when a CQ has a $\mathsf{GHW}(k)$-overapproximation. We provide an alternative characterization in terms of a *boundedness* condition for the existential cover game. This helps understanding where lies the difficulty of determining the decidability status of the problem of existence of $\mathsf{GHW}(k)$-overapproximations, for $k > 1$.

We write $(\mathcal{D}, \bar{a}) \to^c_k (\mathcal{D}, \bar{b})$, for $k \geq 1$ and $c \geq 0$, if Duplicator has a winning strategy *in the first c rounds* of the existential k-cover game on $(\mathcal{D}, \bar{a})$ and $(\mathcal{D}, \bar{b})$. The next result establishes that a CQ q has a $\mathsf{GHW}(k)$-overapproximation iff the existential k-cover game played from q is "bounded", i.e., there is a constant $c \geq 0$ that bounds the number of rounds this game needs to be played in order to determine if Duplicator wins.

▶ **Theorem 13.** *Fix $k \geq 1$. The CQ $q(\bar{x})$ has a $\mathsf{GHW}(k)$-overapproximation iff there is an integer $c \geq 0$ such that $(q, \bar{x}) \to_k (\mathcal{D}, \bar{a})$ iff $(q, \bar{x}) \to^c_k (\mathcal{D}, \bar{a})$, for each database $\mathcal{D}$ and $\bar{a} \in \mathcal{D}$.*

Boundedness conditions are a difficult area of study, with a delicate decidability boundary. For *least fixed point logic* (LFP), undecidability results for boundedness abound with the exception of a few restricted fragments [7,26]. Although the existence of winning Duplicator strategies in existential pebble games is expressible in LFP [24], no result obtained in such context seems to be directly applicable to determining the decidability status of the boundedness condition in Theorem 13.

5 Beyond under and overapproximations: Δ-approximations

We now turn to $\mathsf{GHW}(k)$-Δ-approximations. Recall that a $\mathsf{GHW}(k)$-Δ-approximation of q is a maximal element in $\mathsf{GHW}(k)$ with respect to the order $\sqsubseteq_q$, where $q' \sqsubseteq_q q''$, for CQs $q', q'' \in \mathsf{GHW}(k)$, iff $\Delta(q(\mathcal{D}), q''(\mathcal{D})) \subseteq \Delta(q(\mathcal{D}), q'(\mathcal{D}))$ for all databases $\mathcal{D}$. It is not surprising that $\mathsf{GHW}(k)$-Δ-approximations generalize over- and underapproximations.

▶ **Proposition 9.** *Fix $k \geq 1$. Let q, q' be CQs such that $q' \in \mathsf{GHW}(k)$. If $q \subseteq q'$ (resp., $q' \subseteq q$), then q' is a $\mathsf{GHW}(k)$-Δ-approximation of q if and only if q' is a $\mathsf{GHW}(k)$-overapproximation (resp., $\mathsf{GHW}(k)$-underapproximation) of q.*

Thus, we concentrate on the study of $\mathsf{GHW}(k)$-Δ-approximations that are neither $\mathsf{GHW}(k)$-under nor $\mathsf{GHW}(k)$-overapproximations. Evaluating such Δ-approximations can give us useful information when the quality of $\mathsf{GHW}(k)$-under- and $\mathsf{GHW}(k)$-overapproximations is poor. But, are there $\mathsf{GHW}(k)$-Δ-approximations that are neither $\mathsf{GHW}(k)$-under- nor $\mathsf{GHW}(k)$-overapproximations? In the rest of this section, we settle this question and study complexity questions associated with such $\mathsf{GHW}(k)$-Δ-approximations.

5.1 Incomparable $\mathsf{GHW}(k)$-Δ-approximations

Let q be a CQ. In view of Proposition 9, the $\mathsf{GHW}(k)$-Δ-approximations q' of q that are neither $\mathsf{GHW}(k)$-overapproximations nor $\mathsf{GHW}(k)$-underapproximations must be *incomparable* with q in terms of containment; i.e., both $q \not\subseteq q'$ and $q' \not\subseteq q$ must hold. Incomparable $\mathsf{GHW}(k)$-Δ-approximations do not necessarily exist, even when approximating in the set of infinite CQs $\mathsf{GHW}(k)^\infty$. A trivial example is any CQ q in $\mathsf{GHW}(k)$, as its only $\mathsf{GHW}(k)$-Δ-approximation (up to equivalence) is q itself. The following characterization will help us to find CQs with incomparable $\mathsf{GHW}(k)$-Δ-approximations.

▶ **Theorem 14.** *Fix $k \geq 1$. Let $q(\bar{x}), q'(\bar{x})$ be CQs such that $q' \in \mathsf{GHW}(k)$. Then q' is an incomparable $\mathsf{GHW}(k)$-Δ-approximation of q iff $(q, \bar{x}) \to_k (q', \bar{x})$, and both $q \not\subseteq q'$ and $q' \not\subseteq q$.*

Proof. Suppose first that q' is an incomparable $\mathsf{GHW}(k)$-Δ-approximation of q and assume, towards a contradiction, that $(q, \bar{x}) \not\to_k (q', \bar{x})$. By Proposition 5, there is a $q'' \in \mathsf{GHW}(k)$ such that $(q'', \bar{x}) \to (q, \bar{x})$ and $(q'', \bar{x}) \not\to (q', \bar{x})$. In particular, $q \subseteq q''$ and $q' \not\subseteq q''$. We claim that $q' \sqsubset_q (q'' \wedge q')$, which contradicts our hypothesis since $(q'' \wedge q') \in \mathsf{GHW}(k)$. Indeed, suppose that $\bar{a} \in \Delta(q(\mathcal{D}), (q'' \wedge q')(\mathcal{D}))$, for a database $\mathcal{D}$ and a tuple $\bar{a} \in \mathcal{D}$. If $\bar{a} \notin q(\mathcal{D})$, then $\bar{a} \in (q'' \wedge q')(\mathcal{D}) \subseteq q'(\mathcal{D})$, and consequently, $\bar{a} \in \Delta(q(\mathcal{D}), q'(\mathcal{D}))$. Otherwise, $\bar{a} \in q(\mathcal{D})$ and $\bar{a} \notin (q'' \wedge q')(\mathcal{D})$. Since $q \subseteq q''$, the tuple $\bar{a}$ is not in $q'(\mathcal{D})$, and then $\bar{a} \in \Delta(q(\mathcal{D}), q'(\mathcal{D}))$. Hence $q' \sqsubseteq_q (q'' \wedge q')$. On the other hand, since $q' \not\subseteq q''$, there is a database $\mathcal{D}^*$ such that $q'(\mathcal{D}^*) \not\subseteq q''(\mathcal{D}^*)$, i.e., $\bar{a} \in q'(\mathcal{D}^*)$ but $\bar{a} \notin q''(\mathcal{D}^*)$, for some tuple $\bar{a}$ in $\mathcal{D}^*$. In particular $\bar{a} \in \Delta(q(\mathcal{D}^*), q'(\mathcal{D}^*))$ and $\bar{a} \notin \Delta(q(\mathcal{D}^*), (q'' \wedge q')(\mathcal{D}^*))$, and thus $(q'' \wedge q') \not\sqsubseteq_q q'$.

For the other implication, we need the following lemma, whose proof is in the appendix:

▶ **Lemma 15.** *Fix $k \geq 1$. Let $q(\bar{x}), q'(\bar{x}), q''(\bar{x})$ be CQs such that $q'' \in \mathsf{GHW}(k)$. Suppose that $(q, \bar{x}) \to_k (q', \bar{x})$. Then $(q'', \bar{x}) \to (q' \wedge q, \bar{x})$ implies $(q'', \bar{x}) \to (q', \bar{x})$.*

Assume that $q \not\subseteq q'$, $q' \not\subseteq q$, and $(q, \bar{x}) \to_k (q', \bar{x})$ hold. By contradiction, suppose that there is a CQ $q'' \in \mathsf{GHW}(k)$ such that $q' \sqsubset_q q''$. We show that $q' \equiv q''$, which is a contradiction. Recall that $\mathcal{D}_{(q' \wedge q)}$ denotes the canonical database of $(q' \wedge q)$. Clearly, $\bar{x} \in q(\mathcal{D}_{(q' \wedge q)})$ and $\bar{x} \in q'(\mathcal{D}_{(q' \wedge q)})$. It follows that $\bar{x} \notin \Delta(q(\mathcal{D}_{(q' \wedge q)}), q'(\mathcal{D}_{(q' \wedge q)}))$, and thus by hypothesis, $\bar{x} \notin \Delta(q(\mathcal{D}_{(q' \wedge q)}), q''(\mathcal{D}_{(q' \wedge q)}))$. Therefore, $\bar{x} \in q''(\mathcal{D}_{(q' \wedge q)})$. Applying Lemma 15, we have that $(q'', \bar{x}) \to (q', \bar{x})$, that is, $q' \subseteq q''$. We prove next that also $q'' \subseteq q'$.

Notice that $\bar{x} \notin q(\mathcal{D}_{q''})$; otherwise, $q'' \subseteq q$ would hold, implying that $q' \subseteq q$, which is a contradiction. Thus, $\bar{x} \in q''(\mathcal{D}_{q''})$, and hence $\bar{x} \in \Delta(q(\mathcal{D}_{q''}), q''(\mathcal{D}_{q''}))$. This implies that $\bar{x} \in \Delta(q(\mathcal{D}_{q''}), q'(\mathcal{D}_{q''}))$. It follows that $\bar{x} \in q'(\mathcal{D}_{q''})$, i.e., $q'' \subseteq q'$. Hence, $q' \equiv q''$. ◀

▶ **Example 16.** Consider the Boolean CQ $q = \exists x \exists y \exists z (E(x,y) \wedge E(y,z) \wedge E(z,x))$ from Figure 2, which asks for the existence of a directed cycle of length 3. The query q has a unique $\mathsf{GHW}(1)$-underapproximation $q' = \exists x E(x,x)$. As mentioned in Section 3.1, q has no $\mathsf{GHW}(1)$-overapproximations. Does q have incomparable $\mathsf{GHW}(1)$-Δ-approximations? By applying Theorem 14, we can give a positive answer to this question: the CQ $q'' = \exists x \exists y (E(x,y) \wedge E(y,x))$ is an incomparable $\mathsf{GHW}(1)$-Δ-approximation of q. ◀

Therefore, as Example 16 shows, incomparable $\mathsf{GHW}(k)$-Δ-approximations may exist for some CQs. However, in contrast with overapproximations, they are not unique in general:

▶ **Proposition 10.** *There is a CQ with infinitely many (non-equivalent) incomparable* $\mathsf{GHW}(1)$-Δ-*approximations. In fact, this holds for the CQ q in Figure 1.*

Identification, existence and evaluation. A direct consequence of Theorem 14 is that the identification problem, i.e., checking if $q' \in \mathsf{GHW}(k)$ is an incomparable $\mathsf{GHW}(k)$-Δ-approximation of a CQ q, is in coNP. In fact, we need to check that $q \not\subseteq q'$ and $q' \not\subseteq q$ – which are in coNP – and $(q, \bar{x}) \to_k (q', \bar{x})$ – which is in PTIME from Proposition 4. This is optimal:

▶ **Proposition 11.** *Fix $k \geq 1$. Checking if a given CQ $q' \in \mathsf{GHW}(k)$ is an incomparable* $\mathsf{GHW}(k)$-Δ-*approximation of a given CQ q, is* coNP-*complete.*

As in the case of $\mathsf{GHW}(k)$-overapproximations, we do not know how to check existence of incomparable $\mathsf{GHW}(k)$-Δ-approximations, for $k \geq 1$. Nevertheless, for $k = 1$ we can exploit the automata techniques developed in Section 4 and obtain an analogous decidability result:

▶ **Proposition 12.** *There is a* 2EXPTIME *algorithm that checks if a CQ q has a incomparable* $\mathsf{GHW}(1)$-Δ-*approximation and, if one exists, it computes one in triple exponential time. The bounds become* EXPTIME *and* 2EXPTIME*, respectively, if the arity of the schema is fixed.*

Now we study evaluation. Let us note that, unlike $\mathsf{GHW}(k)$-overapproximations, incomparable $\mathsf{GHW}(k)$-Δ-approximations of a CQ q are not unique. In fact, there can be infinitely many (see Proposition 10). Thus, it is reasonable to start by trying to evaluate *at least one* of them. It would be desirable, in addition, if the one we evaluate depends only on q (i.e., it is independent of the underlying database $\mathcal{D}$). Proposition 12 allows us to do so as follows. Given a CQ q with at least one incomparable $\mathsf{GHW}(1)$-Δ-approximation, we can compute in 3EXPTIME one such an incomparable $\mathsf{GHW}(1)$-Δ-approximation q'. We can then evaluate q' over a database $\mathcal{D}$ in time $O(|\mathcal{D}| \cdot |q'|)$ [29], which is $O(|\mathcal{D}| \cdot f(|q|))$, for f a triple-exponential function. This means that the evaluation of such q' over $\mathcal{D}$ is *fixed-parameter tractable*, i.e., it can be solved by an algorithm that depends polynomially on the size of the large database $\mathcal{D}$, but more loosely on the size of the small CQ q. (This is a desirable property for evaluation, which is not in general held for the class of all CQs [27]). Formally, then:

▶ **Theorem 17.** *There is a fixed-parameter tractable algorithm that, given a CQ q that has incomparable* $\mathsf{GHW}(1)$-Δ-*approximations, a database $\mathcal{D}$, and a tuple $\bar{a}$, checks whether $\bar{a} \in q'(\mathcal{D})$, for some incomparable* $\mathsf{GHW}(1)$-Δ-*approximation q' of q that depends only on q.*

It is worth noticing that the automata techniques are essential for proving this result, and thus for evaluating incomparable $\mathsf{GHW}(1)$-Δ-approximations. This is in stark contrast with $\mathsf{GHW}(k)$-overapproximations, which can be evaluated in polynomial time by simply checking if $(q, \bar{x}) \to_k (\mathcal{D}, \bar{a})$. It is not at all clear whether such techniques can be extended to allow for the efficient evaluation of incomparable $\mathsf{GHW}(k)$-Δ-approximations.

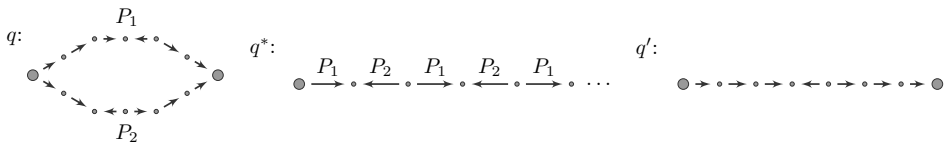


Figure 4 The CQ $q \in \mathsf{GHW}(2)$ from Example 18. The CQ $(q^* \wedge q')$ is an incomparable $\mathsf{GHW}(1)^\infty$-Δ-approximation of q. On the other hand, q has no incomparable $\mathsf{GHW}(1)$-Δ-approximations.

The infinite case. All the previous results continue to apply for the class of infinite CQs in $\mathsf{GHW}(k)^\infty$. The following example shows that, as in the case of $\mathsf{GHW}(k)$-overapproximations, considering $\mathsf{GHW}(k)^\infty$ helps us to obtain better incomparable $\mathsf{GHW}(k)$-Δ-approximations.

▶ **Example 18.** Consider the CQ q that asks for the existence of the two parallel paths P_1 and P_2, as shown in Figure 4. Theorem 14 can be used to show that q has no incomparable $\mathsf{GHW}(1)$-Δ-approximation. However, q has an incomparable $\mathsf{GHW}(1)^\infty$-Δ-approximation. In fact, let q^* be the $\mathsf{GHW}(1)^\infty$-overapproximation of q which is depicted in Figure 4 (a P_1-labeled edge represents a copy of the path P_1, similarly for P_2). Also, let q' be an arbitray CQ in $\mathsf{GHW}(1)$ which is incomparable with q (one such a q' is shown in Figure 4). Applying the extension of Theorem 14 to the class $\mathsf{GHW}(k)^\infty$, we can prove that $(q^* \wedge q')$ is an incomparable $\mathsf{GHW}(1)^\infty$-Δ-approximation of q. ◀

Example 18 also illustrates the following fact: If there is a CQ $q' \in \mathsf{GHW}(k)$ which is incomparable with q, then $(q^* \wedge q')$ is an incomparable $\mathsf{GHW}(k)^\infty$-Δ-approximation of q, where q^* is the $\mathsf{GHW}(k)^\infty$-overapproximation of q. Given a database $\mathcal{D}$ and a tuple $\bar{a}$ in $\mathcal{D}$, we can check whether $\bar{a}$ belongs to the evaluation of such a Δ-approximation $(q^* \wedge q')$ over $\mathcal{D}$ as follows: First we compute q', and then we check both $(q, \bar{x}) \to_k (\mathcal{D}, \bar{a})$ and $\bar{a} \in q'(\mathcal{D})$. In other words, we evaluate $(q^* \wedge q')$ via the existential k-cover game, as for the $\mathsf{GHW}(k)^\infty$-overapproximation, and then use the incomparable CQ q' to filter out some tuples in the answer. Interestingly, we can easily exploit automata techniques and compute such an incomparable q' (in case one exists). Thus we have the following:

▶ **Theorem 19.** *Fix $k \geq 1$. There is a fixed-parameter tractable algorithm that given a CQ q that has an incomparable q' in $\mathsf{GHW}(k)$, a database $\mathcal{D}$, and a tuple $\bar{a}$ in $\mathcal{D}$, decides whether $\bar{a} \in \hat{q}(\mathcal{D})$, for some incomparable $\mathsf{GHW}(k)^\infty$-Δ-approximation $\hat{q}$ of q that depends only on q.*

6 Final Remarks

Several problems have been left open by our work; e.g., is the existence of $\mathsf{GHW}(k)$-overapproximations decidable for $k > 1$? What is the complexity of checking for the existence of $\mathsf{GHW}(1)$-overapproximations? What is their exact size? As established in Section 4.2, answering such questions will require a deeper understanding of the boundedness condition for existential cover games stated in Theorem 13.

In the future we plan to study how our notions of approximation can be combined with other techniques to obtain quantitative guarantees. One possibility is to exploit semantic information about the data – e.g., in the form of integrity constraints – in order to ensure that certain bounds on the size of the result of the approximation hold. Another possibility is to try to obtain probabilistic guarantees for approximations based on reasonable assumptions about the distribution of the data.

References

1 Serge Abiteboul, Richard Hull, and Victor Vianu. *Foundations of Databases*. Addison-Wesley, 1995.
2 Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2003.
3 Pablo Barceló. Querying graph databases. In *PODS*, pages 175–188, 2013.
4 Pablo Barceló, Leonid Libkin, and Miguel Romero. Efficient approximations of conjunctive queries. In *PODS*, pages 249–260, 2012.
5 Pablo Barceló, Leonid Libkin, and Miguel Romero. Efficient approximations of conjunctive queries. *SIAM J. Comput.*, 43(3):1085–1130, 2014.
6 Pablo Barceló, Miguel Romero, and Moshe Y. Vardi. Semantic acyclicity on graph databases. *SIAM J. Comput.*, 45(4):1339–1376, 2016.
7 Achim Blumensath, Martin Otto, and Mark Weyer. Decidability results for the boundedness problem. *Logical Methods in Computer Science*, 10(3), 2014.
8 Ashok K. Chandra and Philip M. Merlin. Optimal implementation of conjunctive queries in relational data bases. In *STOC*, pages 77–90, 1977.
9 Hubie Chen and Víctor Dalmau. Beyond hypertree width: Decomposition methods without decompositions. In *CP*, pages 167–181, 2005.
10 Stavros S. Cosmadakis, Haim Gaifman, Paris C. Kanellakis, and Moshe Y. Vardi. Decidable optimization problems for database logic programs (preliminary report). In *STOC*, pages 477–490, 1988.
11 Víctor Dalmau, Phokion G. Kolaitis, and Moshe Y. Vardi. Constraint satisfaction, bounded treewidth, and finite-variable logics. In *CP*, pages 310–326, 2002.
12 Wenfei Fan, Jianzhong Li, Shuai Ma, Nan Tang, Yinghui Wu, and Yunpeng Wu. Graph pattern matching: From intractable to polynomial time. *PVLDB*, 3(1):264–275, 2010.
13 Robert Fink and Dan Olteanu. On the optimal approximation of queries using tractable propositional languages. In *ICDT*, pages 174–185, 2011.
14 Minos Garofalakis and Phillip Gibbon. Approximate query processing: taming the terabytes. In *VLDB*, page 725, 2001.
15 Georg Gottlob, Gianluigi Greco, Nicola Leone, and Francesco Scarcello. Hypertree decompositions: Questions and answers. In *PODS*, pages 57–74, 2016.
16 Georg Gottlob, Nicola Leone, and Francesco Scarcello. Hypertree decompositions and tractable queries. *J. Comput. Syst. Sci.*, 64(3):579–627, 2002.
17 Georg Gottlob, Zoltán Miklós, and Thomas Schwentick. Generalized hypertree decompositions: Np-hardness and tractable variants. *J. ACM*, 56(6), 2009.
18 P. Hell and J. Nešeťril. *Graphs and homomorphisms*. Oxford University Press, 2004.
19 Pavol Hell and Jaroslav Nesetril. The core of a graph. *Discrete Mathematics*, 109(1-3):117–126, 1992.
20 Pavol Hell, Jaroslav Nesetril, and Xuding Zhu. Complexity of tree homomorphisms. *Discrete Applied Mathematics*, 70(1):23–36, 1996.
21 Yannis Ioannidis. Approximations in database systems. In *ICDT*, pages 16–30, 2003.
22 Phokion G. Kolaitis and Jonathan Panttaja. On the complexity of existential pebble games. In *CSL*, pages 314–329, 2003.
23 Phokion G. Kolaitis and Moshe Y. Vardi. On the expressive power of datalog: Tools and a case study. *J. Comput. Syst. Sci.*, 51(1):110–134, 1995.
24 Phokion G. Kolaitis and Moshe Y. Vardi. Conjunctive-query containment and constraint satisfaction. *J. Comput. Syst. Sci.*, 61(2):302–332, 2000.
25 Qing Liu. Approximate query processing. In *Encyclopedia of Database Systems*, pages 113–119, 2009.

26 Martin Otto. The boundedness problem for monadic universal first-order logic. In *LICS*, pages 37–48, 2006.

27 Christos H. Papadimitriou and Mihalis Yannakakis. On the complexity of database queries. *J. Comput. Syst. Sci.*, 58(3):407–427, 1999.

28 D. Sangiorgi. On the origins of bisimulation and coinduction. *ACM Trans. Program. Lang. Syst.*, 31(4), 2009.

29 Mihalis Yannakakis. Algorithms for acyclic database schemes. In *VLDB*, pages 82–94, 1981.

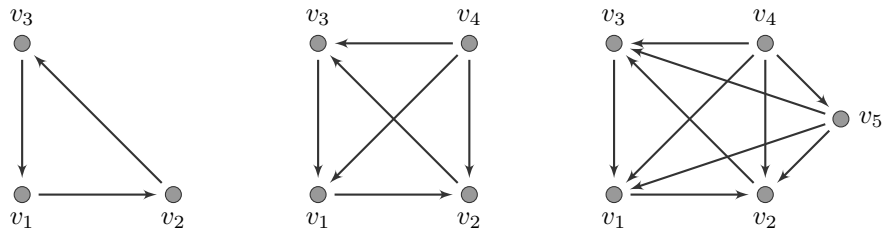


Figure 5 Directed graphs that satisfy condition (†) for $k = 2, 3, 4$, respectively.

7 Appendix

7.1 Proofs for Section "Existence and uniqueness of overapproximations"

Proposition 2. *For each $k > 1$, there is a Boolean CQ $q \in \mathsf{GHW}(k)$ without $\mathsf{GHW}(\ell)$-overapproximations for any $1 \leq \ell < k$.*

Proof. Fix $k > 1$. The CQ q is defined over graphs and consists of $k+1$ nodes $v_1, \ldots, v_{k+1}$. For every $1 \leq i < j \leq k+1$ we add either the atom (i.e., edge) $E(v_i, v_j)$ or $E(v_j, v_i)$ to q in such a way that the subgraph of G induced by $\{v_1, v_2, v_3\}$ is a directed cycle and a certain condition (†), defined below, holds. We start introducing some terminology.

Let G be a directed graph on nodes $v_1, \ldots, v_{k+1}$ that contains, for each $1 \leq i < j \leq k+1$, either the edge $E(v_i, v_j)$ or $E(v_j, v_i)$. For a $B \subseteq \{v_1, \ldots, v_{k+1}\}$ of size $2 \leq \ell \leq k-1$ and a node $v \in \{v_1, \ldots, v_{k+1}\} \setminus B$, we define $\mathsf{conn}(v, B)$ as the tuple $(e_1, \ldots, e_{k+1}) \in \{-1, 1, \#\}^{k+1}$ such that for each $1 \leq p \leq k+1$:

$$e_p \;=\; \begin{cases} \#, & \text{if } v_p \notin B, \\ 1, & \text{if } v_p \in B \text{ and the edge } E(v, v_p) \text{ is in } G, \\ -1, & \text{otherwise, i.e., } v_p \in B \text{ and } E(v_p, v) \text{ is in } G. \end{cases}$$

In simple terms, $\mathsf{conn}(v, B)$ specifies how v connects with the nodes in B.

Our condition (†) then establishes the following:

(†) For each $B \subseteq \{v_1, \ldots, v_{k+1}\}$ of size $2 \leq \ell \leq k-1$ and node $v \in \{v_1, \ldots, v_{k+1}\} \setminus B$, there is a node $v' \in \{v_1, \ldots, v_{k+1}\} \setminus B$ such that:

$$\mathsf{conn}(v, B) \quad \neq \quad \mathsf{conn}(v', B).$$

That is, for each such B and v we will always be able to find another v' outside B that connects to the nodes in B in a different way than v.

▶ **Example 20.** *The graphs in Figure 5 satisfy this condition for $k = 2, 3, 4$, respectively. Notice that the directed cycle on nodes $\{v_1, v_2, v_3\}$, shown in the left-hand side, satisfies condition (†) trivially.*

The next claim establishes that for each $k > 1$ there is always a graph that satisfies this condition.

▶ **Claim 1.** *There is a directed graph G on nodes $v_1, \ldots, v_{k+1}$ such that the following hold:*

1. *For each $1 \leq i < j \leq k+1$, either the edge $E(v_i, v_j)$ or $E(v_j, v_i)$ is in G;*

2. *the subgraph of G induced by $\{v_1, v_2, v_3\}$ is a directed cycle; and*
3. *G satisfies condition (†).*

Proof. For $k = 2$ this is given by the graph in Example 20. For $k \geq 3$ we prove by induction a stronger claim: There is a directed graph G on nodes $v_1, \ldots, v_{k+1}$ such that:

1. G contains either the edge $E(v_i, v_j)$ or $E(v_j, v_i)$ for each $1 \leq i < j \leq k+1$.
2. The subgraph of G induced by $\{v_1, v_2, v_3\}$ is a directed cycle.
3. G contains the edges $E(v_1, v_2)$ and $E(v_4, v_3)$.
4. G satisfies condition (†).

The basis case $k = 3$ is given by the graph in Example 20. For the inductive case, assume by induction hypothesis that there is a directed graph G on nodes $v_1, \ldots, v_{k+1}$ that satisfies the claim above. A new graph G' is then created from G by adding a new node v_{k+2} and connecting it to the nodes in $\{v_1, \ldots, v_{k+1}\}$ as follows: For each $1 \leq i \leq k$, if $E(v_i, v_{i+1})$ is in G then we add the edge $E(v_{k+2}, v_i)$ to G', otherwise we add the edge $E(v_i, v_{k+2})$. Moreover, if $E(v_{k+1}, v_1)$ is in G then we add the edge $E(v_{k+2}, v_{k+1})$ to G', otherwise we add the edge $E(v_{k+1}, v_{k+2})$. Notice that G coincides with the subgraph of G' that is induced by nodes $v_1, \ldots, v_{k+1}$. Moreover, by construction G' satisfies the first three conditions of the claim. We prove next that it also satisfies condition (†).

Take an arbitrary $B \subseteq \{v_1, \ldots, v_{k+2}\}$ of size $2 \leq \ell \leq k$ and a node v outside B. We prove that the condition holds by cases:

- $v_{k+2} \notin B$, $v \in \{v_1, \ldots, v_{k+1}\}$, and $2 \leq \ell \leq k-1$: By inductive hypothesis there is a node $v' \in \{v_1, \ldots, v_{k+1}\} \setminus B$ such that $\mathsf{conn}(v, B) \neq \mathsf{conn}(v', B)$.
- $v_{k+2} \notin B$, $v \in \{v_1, \ldots, v_{k+1}\}$, and $\ell = k$: We set $v' := v_{k+2}$ and claim that the predecessor u of v in $\{v_1, \ldots, v_{k+1}\}$ distinguishes v and v'. Here, the "predecessor" of v_{i+1} is v_i if $2 \leq i \leq k+1$, and the predecessor of v_1 is v_{k+1} (note that $u \in B$ as $\ell = k$). By construction of G', we have that $E(u, v) \in G'$ if and only if $E(v', u) \in G'$. We conclude that $\mathsf{conn}(v, B) \neq \mathsf{conn}(v', B)$.
- $v_{k+2} \notin B$ and $v = v_{k+2}$: There must exist some node v' in $\{v_1, \ldots, v_{k+1}\}$ that does not belong to B but its predecessor u with respect to $\{v_1, \ldots, v_{k+1}\}$ does. Then by construction of G', we have that $E(u, v') \in G'$ if and only if $E(v, u) \in G'$. We conclude that $\mathsf{conn}(v, B) \neq \mathsf{conn}(v', B)$.
- $v_{k+2} \in B$ and $\ell \geq 3$: Then $B' = B \setminus \{v_{k+2}\}$ is of size $2 \leq \ell - 1 \leq k - 1$. By induction hypothesis, for every node v outside B' there is another node $v' \in \{v_1, \ldots, v_{k+1}\} \setminus B'$ such that $\mathsf{conn}(v, B') \neq \mathsf{conn}(v', B')$. This implies that $\mathsf{conn}(v, B) \neq \mathsf{conn}(v', B)$.
- $v_{k+2} \in B$ and $\ell = 2$: Then $B = \{v_{k+2}, u\}$ for some $u \in \{v_1, \ldots, v_{k+1}\}$. Suppose first that $u \in \{v_1, v_2, v_3\}$. Since the subgraph induced by $\{v_1, v_2, v_3\}$ in G defines a directed cycle, it is the case that $E(u, z)$ holds if and only if $E(w, u)$ holds, where $\{u, w, z\} = \{v_1, v_2, v_3\}$. Therefore, for each $v \in \{v_1, \ldots, v_{k+1}\} \setminus B$ there is a node $v' \in \{z, w\}$ such that $\mathsf{conn}(v, \{u\}) \neq \mathsf{conn}(v', \{u\})$. It follows that $\mathsf{conn}(v, B) \neq \mathsf{conn}(v', B)$. Suppose now that $u \notin \{v_1, v_2, v_3\}$. It suffices to exhibit two nodes v' and v'' outside B such that $E(v', v_{k+2})$ and $E(v_{k+2}, v'')$. By induction hypothesis the edges $E(v_1, v_2)$ and $E(v_4, v_3)$ are in G'. Therefore, v_{k+2} is connected via edges $E(v_3, v_{k+2})$ and $E(v_{k+2}, v_1)$ in G'.

This concludes the proof. ◀

Fix $k \geq 1$. We then take as q any Boolean CQ whose canonical database is a graph G on nodes $v_1, \ldots, v_{2k+1}$ such that: (1) For each $1 \leq i < j \leq 2k+1$, either the edge $E(v_i, v_j)$ or $E(v_j, v_i)$ is in G, (2) the subgraph of G induced by $\{v_1, v_2, v_3\}$ is a directed cycle, and (3) G

satisfies condition (†). It is easy to see that q is in $\mathsf{GHW}(k+1) \setminus \mathsf{GHW}(k)$ as its underlying undirected graph is a clique on $2k+1$ elements. In fact, these elements can be covered with $(k+1)$ edges, but not with k.

We claim that q has no $\mathsf{GHW}(\ell)$-overapproximation for any $1 \leq \ell \leq k$. The proofs for the cases when $\ell = 1$ and $\ell > 1$ are slightly different. We start with the latter, i.e., when $1 < \ell \leq k$. The proof for every such an ℓ is analogous, and thus we concentrate on proving the claim for $\ell = k > 1$. According to Theorem 3, we need to prove that there is no constant $c \geq 0$ such that for every database $\mathcal{D}$:

$$q \to_k \mathcal{D} \quad \iff \quad q \to_k^c \mathcal{D}.$$

The proof is conceptually simple. We inductively define a CQ q'_c in $\mathsf{GHW}(k)$, for each $c \geq 0$, such that q'_c defines the existential k-cover game played from q for up to c rounds. Formally, q'_c is the CQ in $\mathsf{GHW}(k)$ such that for every $\mathcal{D}$:

$$q'_c \to \mathcal{D} \quad \iff \quad q \to_k^c \mathcal{D}.$$

Afterwards we show that there is no integer c such that q'_c and q'_{c+1} are equivalent. It follows that q has no $\mathsf{GHW}(k)$-overapproximation.

The construction of the CQ q'_c is technical, but mimics that of $q' \in \mathsf{GHW}(k)^\infty$ in Lemma 8. Here we obtain q'_c to be finite since only a finite number of rounds of the game have to be represented.

For the definition of q'_c, for $c \geq 1$, some sets of variables of size $2k$ in q'_c are *distinguished*. Each such set will be associated, in a 1-1 correspondence, with a subset S of $\{v_1, \ldots, v_{2k+1}\}$ of size $2k$. We exploit this correspondence in the construction. (A similar notion, called *boundaried* databases, is used in the proof of some results in [24]). In order to make the proof more uniform, we assume without loss of generality that we deal with a stronger version of the existential k-cover game in which the spoiler is allowed to place as many pebbles as desired in the first round of the game (notice, however, that he cannot place more than $2k$ such pebbles, since these have to be covered by at most k atoms from q).

Let us start with the base cases q'_0 and q'_1. The CQ q'_0 has no elements, and therefore no distinguished sets of elements. We now define q'_1. Recall that G is the graph that represents the canonical database of the CQ q and that $\{v_1, \ldots, v_{2k+1}\}$ are the vertices of G. Given a subset S of $\{v_1, \ldots, v_{2k+1}\}$ of size $2k$, we denote by $G(S)$ the subgraph of G induced by the nodes in S. (Notice that the underlying undirected graph of such $G(S)$ is a clique on $2k$ elements, and thus it belongs to $\mathsf{GHW}(k) \setminus \mathsf{GHW}(k-1)$). The CQ q'_1 is then defined as the graph that consists of a disjoint copy of each $G(S)$, for S a $2k$ element subset of $\{v_1, \ldots, v_{2k+1}\}$. This graph represents all possible moves of Spoiler in the first round of the existential k-cover game played from q (recall that we allow the Spoiler to place $2k$ pebbles in such first round). The distinguished sets of $2k$ variables from q'_1 are then defined precisely as its connected components. Each one of them can thus be naturally associated with a $2k$ element subset of $\{v_1, \ldots, v_{2k+1}\}$.

We now explain how to construct the CQ q'_{c+1} from q'_c. For each $S \subseteq \{v_1, \ldots, v_{2k+1}\}$ of size $2k$ and node $v \in S$, let S_v be the $2k$ element set obtained from S by removing v and adding the only element $v(S)$ in $\{v_1, \ldots, v_{2k+1}\} \setminus S$. The CQ q'_{c+1} is then constructed as follows: For each distinguished set Q of $2k$ variables in q'_c that is naturally associated (in 1-1 correspondence) with $S \subseteq \{v_1, \ldots, v_{2k+1}\}$, and for each node $v \in S$, add a fresh copy $Q(S_v)$ of $G(S_v)$ to q'_c. Then "glue" together the "common elements" of the distinguished set Q and $Q(S_v)$. Formally, this means to identify in $Q(S_v)$ each node v' in $S_v \cap S$ with the node $v'' \in Q$ that is in 1-1 correspondence with it. (Notice that all nodes in $Q(S_v)$, save for $v(S)$,

are glued with elements in Q). Intuitively, the new nodes correspond to all possible moves of one Spoiler pebble in q, starting from the configuration in which his pebbles cover S. The resulting graph is q'_{c+1}.

We now explain what are the distinguished sets of $2k$ elements in q'_{c+1}. For each distinguished set Q of q'_c that is naturally associated (in 1-1 correspondence) with $S \subseteq \{v_1, \ldots, v_{2k+1}\}$ of size $2k$, we distinguish the fresh copies of the form $Q(S_v)$ that are glued to Q (after gluing them). We associate this distinguished set precisely with S_v (respecting the natural 1-1 correspondence).

The following claim states that the CQs of the form q'_c, for $c \geq 0$, satisfy the desired properties. The proof of this claim is standard but cumbersome:

▶ **Claim 2.** *For each $c \geq 0$ the following statements hold:*

1. *q'_c is in $\mathsf{GHW}(k)$.*
2. *For every database $\mathcal{D}$ it is the case that:*

$$q'_c \to \mathcal{D} \iff q \to^c_k \mathcal{D}.$$

We now prove that for no integer $c \geq 0$ it is the case that:

$$q'_{c+1} \equiv q'_c.$$

Clearly, $q'_{c+1} \subseteq q'_c$ for every $c \geq 0$, as $q'_c \to q'_{c+1}$ by construction. Thus we show that it is not the case that $q'_c \subseteq q'_{c+1}$ for some $c \geq 0$. To this end we prove:

$$q'_{c+1} \not\to q'_c, \qquad \text{for each integer } c \geq 0. \tag{3}$$

We do this by induction. But before doing so, we briefly review the notion of *core* of a CQ [8, 19], which is heavily used in the rest of the proof.

Formally, a CQ q is a core if there is no CQ q' with fewer atoms than q such that $q \equiv q'$. Given CQs q and q', we say that q is a core of q' if q is a core and $q \equiv q'$. In other words, the core of q is the minimal CQ (in terms of number of atoms) that is equivalent to q. The following result summarizes some important properties of cores:

▶ **Proposition 13.** [19] *The following statements hold:*

1. *Each CQ q has a core. Moreover, there is a unique core of q up to renaming of variables. (Therefore, we can talk about* the *core of q).*
2. *Each core is a* retraction *of q. This means that a core q' of q can always be obtained by removing some (perhaps none) of the atoms of q. In addition, if q' is a core of q then there is a homomorphism $h : q \to q'$ that is the identity on the elements of q'.*

We now continue with the proof of the proposition. The claim in Equation (3) clearly holds for $c = 0$ (as q'_0 is empty while q'_1 is not). Let us assume now that the claim holds for $c \geq 0$. That is, $q'_{c+1} \not\to q'_c$. This means, in particular, that the core of q'_{c+1} is not contained in q'_c. That is, this core contains at least one node w in q'_{c+1} that does not belong to q'_c. By definition, this w must satisfy the following: Either $c \geq 0$, or there is a distinguished set Q of q'_c that is associated with an $S \subseteq \{v_1, \ldots, v_{2k+1}\}$ of size $2k$, and there is a node $v \in S$, such that w corresponds to the node in $Q(S_v)$ that represents the unique node in $\{v_1, \ldots, v_{2k+1}\} \setminus S$. Recall that $Q(S_v)$ is the fresh copy of $G(S_v)$ that is glued with Q while building q'_{c+1}. Thus, w is the only node in such copy that is not identified with a node in Q.

From Proposition 13, we can assume that the homomorphism that maps q'_{c+1} to its core is a retraction, i.e., it is the identity on the nodes of such core, in particular, in w. On

the other hand, w is only linked in q'_{c+1} to the other nodes of $Q(S_v)$. Furthermore, the underlying undirected graph of $Q(S_v)$ forms a clique on $2k$ elements. This implies that the elements of $Q(S_v)$ are also in the core of q'_{c+1}.

Recall that $Q(S_v)$ is a distinguished set of variables in q'_{c+1} which is associated with S_v. Take an arbitrary node $v' \in S_v$ that is different from the one that is represented by w in $Q(S_v)$. By definition, q'_{c+2} contains a fresh copy $Q((S_v)_{v'})$ of $G((S_v)_{v'})$ whose nodes are then glued with $Q(S_v)$, save for one node, say w'.

Assume now, for the sake of contradiction, that $q'_{c+2} \to q'_{c+1}$. Then the core of q'_{c+2} corresponds to the core of q'_{c+1}. Henceforth, from Proposition 13 there is a retraction h from q'_{c+2} to this core. Since all elements in $Q(S_v)$ are in such a core, the homomorphism h must be the identity on them. But then h maps w' to the only element in q'_{c+1} that is linked to the same nodes than w' in q'_{c+2}; namely, the node v'. Suppose that w' and v' represent the nodes v_i and v_j in $\{v_1, \ldots, v_{2k+1}\}$, respectively. By assumption $i \neq j$. But this implies then that in the canonical database G of q we have that:

$$\mathsf{conn}(v_i, B) \;=\; \mathsf{conn}(v_j, B),$$

where $B = \{v_1, \ldots, v_{2k+1}\} \setminus \{v_i, v_j\}$. This is a contradiction since B is of size $2k-1 > 1$ and G satisfies condition (†). This concludes our proof that q has no $\mathsf{GHW}(k)$-overapproximation (and, analogously, that it has no $\mathsf{GHW}(\ell)$-overapproximation for every $1 < \ell \leq k$).

We prove next that q neither has a $\mathsf{GHW}(1)$-overapproximation. Let us assume, for the sake of contradiction, that q has a $\mathsf{GHW}(1)$-overapproximation q'. It is an easy observation that the directed graphs in $\mathsf{GHW}(1)$ are precisely those whose underlying undirected graph is acyclic. Note also that q' has no directed cycles of length 2 (i.e., atoms $E(u,v)$ and $E(v,u)$); otherwise q would also have such a cycle (since $q' \to q$). Using the fact that $q' \in \mathsf{GHW}(1)$ and has no directed cycles of length 2, it is not difficult to show (see e.g. [18]) that there is a sufficiently large integer $n \geq 1$ such that, if $\vec{P}_n$ is the directed path on n vertices, then:

$$q' \to \vec{P}_n, \text{ but } \vec{P}_n \not\to q'.$$

This implies that if q'' is the Boolean CQ which is naturally defined by $\vec{P}_n$, then $q'' \subsetneq q'$. Moreover, $\vec{P}_n \to G$. This is due to the fact that G contains a directed cycle on $\{v_1, v_2, v_3\}$. We conclude that:

$$q \subseteq q'' \subsetneq q',$$

and, therefore, that q' is not a $\mathsf{GHW}(1)$-overapproximation of q. This is a contradiction. This concludes our proof of Proposition 2. ◀

7.2 Proofs for Section "A Link to Existential Pebble Games"

A proof of Proposition 5 for Boolean queries is given in [9]. Here we outline a different proof for arbitrary queries. The proof is easily seen to extend to infinite databases and queries. Thus Proposition 6 follows. The proof uses the following lemma.

▶ **Lemma 21.** *Fix $k \geq 1$. If $q(\bar{x}) \in \mathsf{GHW}(k)$ then for every $\mathcal{D}$ and tuple $\bar{a}$ of constants in $\mathcal{D}$:*

$$(q, \bar{x}) \to (\mathcal{D}, \bar{a}) \quad \Longleftrightarrow \quad (q, \bar{x}) \to_k (\mathcal{D}, \bar{a}).$$

Proof sketch. The direction from left to right is straightforward. For the direction from right to left we define an homomorphism h from $(q, \bar{x})$ to $(\mathcal{D}, \bar{a})$ by inspecting the winning strategy of Duplicator along a decomposition of $(q, \bar{x})$ in a top-down fashion.

Thus let $\mathcal{H}$ be a *compact winning strategy* of Duplicator for the game on $(q, \bar{x})$ and $(D, \bar{a})$ and let (F, λ) be a tree decomposition of $(q, \bar{x})$ of generalized hypertreewidth k. Consider the move of Spoiler that pebbles all nodes $\lambda(r)$ in the root r of F. The answer of Duplicator according to $\mathcal{H}$ defines a partial homomorphism h_r from $(q, \bar{x})$ to $(\mathcal{D}, a)$ with domain $\lambda(r)$. Let $r_1, \ldots, r_p$ be the children of r in F. For each $1 \leq i \leq p$, consider the moves of Spoiler that pebble all nodes in $\lambda(r_i)$. The answers to those moves by Duplicator define partial mappings h_{r_i} such that each h_{r_i} is consistent with h_r. Moreover, $h_r \cup h_{r_i}$ and $h_r \cup h_{r_j}$ are also consistent for $i \neq j$, as each element in the domain of both mappings are also in the domain of h_r (since (F, λ) is a tree decomposition of cover width k). Thus by taking the union of these mappings we obtain an homomorphism from q^1 to $\mathcal{D}$, where q^1 is the subgraph of q induced by $\lambda(r) \cup \lambda(r_1) \cup \cdots \lambda(r_p)$. This argument can be inductively repeated until all leaves of F are reached. ◀

Proposition 5. *Fix $k \geq 1$. Then $(\mathcal{D}, \bar{a}) \to_k (\mathcal{D}', \bar{b})$ iff for each CQ $q(\bar{x})$ in $\mathsf{GHW}(k)$ we have that if $(q, \bar{x}) \to (\mathcal{D}, \bar{a})$ then $(q, \bar{x}) \to (\mathcal{D}', \bar{b})$.*

Proof sketch. Suppose $(\mathcal{D}, \bar{a}) \to_k (\mathcal{D}', \bar{b})$. Let $(q, x) \in \mathsf{GHW}(k)$ such that $(q, x) \to (\mathcal{D}, \bar{a})$. We have that winning strategies for Duplicator compose, thus $(q, \bar{x}) \to_k (\mathcal{D}', \bar{b})$. By Lemma 21, we have that $(q, \bar{x}) \to (\mathcal{D}', \bar{b})$.

Now, suppose $(\mathcal{D}, \bar{a}) \not\to_k (\mathcal{D}', \bar{b})$. From Spoiler's winning strategy we derive a CQ $(q, \bar{x}) \in \mathsf{GHW}(k)$ such that $(q, \bar{x}) \to (\mathcal{D}, \bar{a})$ and $(q, \bar{x}) \not\to (\mathcal{D}', \bar{b})$. We construct $(q, \bar{x})$ and its tree decomposition (F, λ) simultaneously.

Suppose Spoiler strategy starts by choosing a set of variables $U \subseteq \mathcal{D}$. Then q initially contains a copy q_0 of the substructure of $(\mathcal{D}, \bar{a})$ induced by U and $\bar{a}$. Further, if r is the root of F, then $\lambda(r)$ corresponds to the set $\mathsf{Var}(q_0)$ of variables of q_0 (note that $\mathsf{Var}(q_0)$ can be covered by k atoms). Then for each partial homomorphism h from $(\mathcal{D}, \bar{a})$ to $(\mathcal{D}', \bar{b})$ with domain U (that is, for each possible response of Duplicator to Spoiler's initial move), Spoiler can play on some $U_h \subseteq \mathcal{D}$ and win the game from this position. For each such partial homomorphism h we add to q a copy q_h of the subgraph of $\mathcal{D}$ induced by U_h. The variables in q_h are fresh variables except for those corresponding to elements in $(U \cup \bar{a}) \cap U_h$. Thus $\mathsf{Var}(q_h) \cap \mathsf{Var}(q_0)$ contains precisely the copies of the elements in $(U \cup \bar{a}) \cap U_h$. For each such a h we add a child r_h to r and we let $\lambda(r_h) = \mathsf{Var}(q_h)$. We continue this construction until we reach the leaves of the strategy tree of Spoiler.

By construction $(q, \bar{x}) \in \mathsf{GHW}(k)$ and $(q, \bar{x}) \to (\mathcal{D}, \bar{a})$ via Φ (just invert the "copying" process). Suppose by contradiction that $(q, \bar{x}) \to (\mathcal{D}', \bar{b})$ via some homomorphism g. Then g restricted to $\lambda(r)$, denoted $g|_{\lambda(r)}$, is a partial homomorphism from $(q, \bar{x})$ to $(\mathcal{D}', \bar{b})$. Take the partial homomorphism $g|_{\lambda(r)} \circ \Phi^{-1}$ with domain $\Phi(\lambda(r))$ from $(\mathcal{D}, \bar{a})$ to $(\mathcal{D}', \bar{b})$. By construction there is a child r' of r associated with $g|_{\lambda(r)} \circ \Phi^{-1}$. Inductively, repeat the argument from r' until we reach a leaf s of F. At this point, $g|_{\lambda(s)} \circ \Phi^{-1}$ is partial homomorphism with domain $\Phi(\lambda(s))$ from $(\mathcal{D}, \bar{a})$ to $(\mathcal{D}', \bar{b})$, but this contradicts the construction of T, as s is a leaf. ◀

Lemma 8. *Fix $k \geq 1$. For every CQ q there is q' in $\mathsf{GHW}(k)^\infty$ such that for every $\mathcal{D}$ and $\bar{a}$:*

$$\bar{a} \in q'(\mathcal{D}) \Leftrightarrow (q', \bar{x}) \to (\mathcal{D}, \bar{a}) \Leftrightarrow (q, \bar{x}) \to_k (\mathcal{D}, \bar{a}).$$

This holds even for countably infinite databases $\mathcal{D}$.

We only sketch the proof as it follows almost directly from techniques developed in [23].

Proof sketch. The main idea is to see a generalized hypertreewidth k tree decomposition of q' as representing all possible strategies of the Spoiler in the existential k-cover game played from q. The root contains all atoms induced in q by the elements where the Spoiler starts placing his pebbles. The children of this root represent all possible moves of one pebble from this initial configuration (i.e., the atoms induced in q by set of elements obtained by changing one element from the initial configuration), and so on. Elements correspond to variables in q and elements in common between a node and its children correspond to join variables in q (i.e., a node and any one of its children share all their variables, save perhaps for one). ◀

7.3 Proofs for Section "Existence of Overapproximations"

7.3.1 The acyclic case

Addendum to the Proof of Proposition 7: We analyze the number of states needed by the 2ATA more precisely. As described in the main part, the automaton reads a encoding of $T' = \mathsf{Enc}(T_{q'}, \chi)$ in which each node t' encodes (a) the variables $\{u_1, \ldots, u_{2r}\}$, (b) the atoms of q' covered by those variables, and is annotated by (c) an encoding of an intended homomorphism $h : q' \to q$ and (d) an encoding of the part of the winning strategy of Duplicator for variables from $\{u_1, \ldots, u_{2r}\}$ that annotate t'. The size of an alphabet for encoding (a)-(d) is of double-exponential size in q if the schema is unbounded (basically due to (d), which requires to store a subset of exponentially many partial homomorphisms) and of polynomial size in q if the schema is bounded.

Even though the alphabet is double-exponential in q, the number of states used by $\mathcal{A}_1$ and $\mathcal{A}_2$ is exponential for schemas of unbounded arity and polynomial for schemas of bounded arity.

The automaton $\mathcal{A}_1$ can be easily seen to be implementable by an exponential number of states. Using universal nodes it checks that (1) the homomorphism from (c) is consistent for variables that annotate adjacent nodes of the tree, and that (2) for all nodes t' of the tree the encoded variables from (a), the atoms from (b), and the part of the homomorphism from q' to q from (c) are consistent.

For $\mathcal{A}_2$, an exponential upper bound for the number of states for schemas of unbounded arity has already been shown in the main part. The essential part for reducing the number of states for schemas of fixed arity k is that after universally guessing (U, U') and g, the automaton only has to store the encodings of those objects in its state. However, as the arity is fixed, the sets U and U' are of constant size and each of their elements, variables of q, can be encoded by logarithmically many bits in q. Similarly the partial homomorphism g can be encoded by $O(\log |q|)$ many bits. Hence only polynomially many states are necessary.

We now turn from boolean queries to non-boolean queries. The construction provided in the main part can be easily extended to non-boolean queries $(q, \bar{x})$ and $(q', \bar{x}')$. In this case, the encoding $T' = \mathsf{Enc}(T_{q'}, \chi)$ of q' includes atoms that may contain free variables. The 2ATA $\mathcal{A}_1$ additionally checks that whenever a node in T' is annotated by an atom $R(\bar{y}')$ then there is an atom $R(\bar{y})$ in q such that $h'(\bar{y}') = \bar{y}$ where h' is the extension of the intended homomorphism h that also maps $\bar{x}'$ to $\bar{x}$. The 2ATA $\mathcal{A}_2$ does an analogous check for the partial homomorphisms.

7.3.2 The case of binary schemas

Theorem 12. *There is a* PTIME *algorithm that checks if a CQ q over a schema of maximum arity two has a* $\mathsf{GHW}(1)$*-overapproximation q', and computes such q' if it exists.*

Proof. We need to introduce some notation. The *Gaifman graph* of a CQ q, denoted by $\mathcal{G}(q)$, is the undirected graph whose set of nodes is the set of variables of q and where there is an edge $\{z, z'\}$ whenever z and z' are distinct variables that appear together in an atom of q. The *existential* Gaifman graph of q, denoted by $\mathcal{G}_\exists(q)$, is the subgraph of $\mathcal{G}(q)$ induced by the existentially quantified variables of q. The following claim is straightforward:

▶ **Claim 3.** *Let $q(\bar{x})$ be a CQ. Then $q \in \mathsf{GHW}(1)$ iff $\mathcal{G}_\exists(q)$ is an acyclic graph.*

We start by proving Theorem 12 for Boolean CQs; thus, until stated otherwise, we assume all CQs to be Boolean. A CQ q is *connected* if $\mathcal{G}(q)$ is connected. The following claim is straightforward:

▶ **Lemma 22.** *Let q be a connected CQ. If q has a $\mathsf{GHW}(1)$-overapproximation, then it has one that is connected.*

First we show Theorem 12 for connected CQs and then we extend it to the non-connected case. The idea is to show that if a CQ q has a $\mathsf{GHW}(1)$-overapproximation then it can be extracted from q in a simple way: it is a subquery of q or it is a subquery of a CQ $q_u \# q_v$ constructed from q and two distinguished variables u, v in q. The algorithm then greedily searches through the subqueries of q and $q_u \# q_v$ to find an overapproximation of q.

We start with the following technical lemma:

▶ **Lemma 23.** *Let q be a connected CQ in $\mathsf{GHW}(1)$. If q is a core then for all variables u and v in q there is at most one endomorphism of q that maps u to v.*

Proof. Assume that there are two distinct endomorphisms h_1 and h_2 with $h_1(u) = h_2(u) = v$. Recall that, since q is a core, h_1 and h_2 are isomorphisms. We root $\mathcal{G}(q)$ at u. Let x be a variable in q with $h_1(x) \neq h_2(x)$ whose distance to u is minimal in $\mathcal{G}(q)$. Then there is a unique y such that $h_1(x) = h_2(y)$. We claim that x and y have the same parent in $\mathcal{G}(q)$. Let z be the parent of x. Since h_1 is an isomorphism, $h_1(x)$ and $h_1(z)$, and therefore also $h_2(y)$ and $h_2(z)$, are adjacent in $\mathcal{G}(q)$. Thus, also y and z are adjacent. However, y cannot be the parent of z since h_1 and h_2 agree on all variables above z in $\mathcal{G}(q)$. Therefore z is the parent of y.

Construct a new endomorphism h that maps variables w in the subtree $\mathcal{G}(q)$ rooted at y to $h_2(w)$, and all other variables w' to $h_1(w')$. Then h is an endomorphism, but not injective as both x and y are mapped to $h_1(x) = h_2(y)$. This is a contradiction to q being a core, and therefore there are no two distinct endomorphisms h_1 and h_2 mapping u to v. ◀

Let q be a CQ. We say that u and v are *adjacent* if $\{u, v\}$ is an edge in $\mathcal{G}(q)$, that is, if u and v appear together in an atom of q. Suppose $q \in \mathsf{GHW}(1)$ is connected and u, v are adjacent. If we remove all the atoms that mention u, v, we obtain two connected CQs, one containing u and the other containing v. We denote these CQs by t_u^q and t_v^q, respectively.

Suppose $q \in \mathsf{GHW}(1)$ is a connected core. If an endomorphism h of q maps u to v where u and v are adjacent, then it is the case that $h(u) = v$ and $h(v) = u$, and h swaps the subqueries t_u^q and t_v^q. We call such an h a *swapping endomorphism* for u and v. Note that Lemma 23 tells us that if such a swapping homomorphism for u and v exists, then it is unique.

▶ **Lemma 24.** *Let q be a connected core in $\mathsf{GHW}(1)$. Then q has at most one endomorphism besides the identity mapping. If this endomorphism exists, it is a swapping endomorphism.*

Proof. Let $P = x_0, x_1, ...x_m$ be a simple path of maximal length in $\mathcal{G}(q)$. For each endomorphism h of q, the path $P' = y_1 \dots y_m$ where $y_i = h(x_i)$ is a simple path of the same length (as q is a core and therefore h is an isomorphism). Furthermore, P and P' share a vertex. Indeed, if this not the case, since q is connected, one can pick w in P and w' in P' such that w and w' are connected by a path P'' vertex-disjoint (except for w and w') from P and P', and construct a longer path than P.

Now, if $|P|$ is even, then its middle vertex $u = x_{m/2}$ is in the intersection of P and P' (as otherwise q would contain a path longer than P). But then $h(u) = u$ and then h must be the identity mapping by Lemma 23.

If $|P|$ is odd, then $u = x_{\lfloor m/2 \rfloor}$ and $v = x_{\lceil m/2 \rceil}$ are in the intersection of P and P' (again, as otherwise q would contain a path longer than P). If $h(u) = u$, again we have that h is the identity. Otherwise, if $h(u) = v$, since u and v are adjacent, we have that h must be the swapping endomorphism for u and v. ◀

For a CQ q and variables u, v in q the CQ $q_u \# q_v$ is defined as follows. Denote by $q \setminus v$ the CQ obtained from q by removing all atoms that contain v. Let q_u be the query constructed from $q \setminus v$ by replacing each variable z by a fresh variable z_u. Similarly let q_v be the CQ where each variable z in $q \setminus u$ is replaced by a fresh variable z_v. The CQ $q_u \# q_v$ is the union of q_u and q_v plus all atoms $R(u_u, v_v)$ when $R(u, v)$ is an atom in q. Likewise for atoms $R(v, u)$. The following claim is straightforward:

▶ **Claim 4.** *For each CQ q and variables u, v in q, it is the case that $q_u \# q_v \to q$.*

Before immersing into the proof of Theorem 12 for connected CQs, we state some important properties of $\mathsf{GHW}(1)$-overapproximations.

▶ **Lemma 25.** *Suppose q is a CQ and suppose q' is a connected core that is a $\mathsf{GHW}(1)$-overapproximation of q. Then we have the following:*

- *If the only endomorphism of q' is the identity, then any homomorphism from q' to q is injective. In particular, q' is a subquery of q.*
- *If q' has a swapping endomorphism for u' and v', then for any homomorphism h from q' to q, we have that*
  - *$(q, \{h(u'), h(v')\}) \to_1 (q', \{u', v'\})$, and*
  - *h is "almost" injective, more precisely, $h(z') \neq h(z'')$ for all pairs of variables z', z'', except maybe for $z' \neq u'$ in $t^{q'}_{u'}$ and $z' \neq v'$ in $t^{q'}_{v'}$. In particular, q' is a subquery of $q_{h(u')} \# q_{h(v')}$.*

Proof. Suppose the only endomorphism of q' is the identity and towards a contradiction, suppose there is a non-injective homomorphism h from q' to q. Then we have $h(z') = h(z'')$, for distinct variables z', z'' in q'. Using the fact that $q \to_1 q'$, it is easy to see that there is a Duplicator winning strategy on q' and q' such that z'' is a possible response of Duplicator when Spoiler starts playing on z'. Since $q' \in \mathsf{GHW}(1)$, we can define an endomorphism g of q' that maps z' to z''. Then g is an endomorphism different from the identity, which is a contradiction.

Suppose now that q' has a swapping endomorphism for u' and v', and let h be a homomoprhism from q' to q. First, assume by contradiction that Duplicator's strategy witnessing $q \to_1 q'$ is such that for $h(u')$ (the case for $h(v')$ is analogous), Duplicator responds with $z' \notin \{u', v'\}$. By composing h with this strategy, and using the fact that $q' \in \mathsf{GHW}(1)$, it follows that there is an endomorphism g of q' that maps u' to z'. This endomorphism is different from the identity and from the swapping endomorphism for u' and

v', which contradicts Lemma 24. Finally, suppose towards a contradiction, that $h(z') = h(z'')$, where $z' \neq z''$ and $z' = u'$ and z'' is in $t^{q'}_{v'}$ (the other case is analogous). Again by composing h with the strategy witnessing $q \to_1 q'$ and the fact that $q' \in \mathsf{GHW}(1)$, it is easy to derive an endomorphism of q' that is neither the identity nor the swapping endomorphism for u' and v'.

◀

As a corollary of Lemma 25 and Lemma 24, we have that whenever q' is a connected core, and it is a $\mathsf{GHW}(1)$-overapproximation of q, then q' is a subquery of q or a subquery of $q_u \# q_v$, for some variables u, v in q.

Proof of Theorem 12 for connected, boolean CQs. We assume that the given CQ q is connected. The algorithm first checks whether a subquery of q is a $\mathsf{GHW}(1)$-overapproximation. This is Step 1. In Step 2, the algorithm checks whether a subquery of $q_u \# q_v$ is a $\mathsf{GHW}(1)$-overapproximation, for some u and v in q. If neither step succeed then the algorithm rejects. Step 1 is as follows:

1. Set q_0 to be q.
2. While $q_i \notin \mathsf{GHW}(1)$, search for an atom e such that $q_i \to_1 q_i \setminus e$. If there is no such atom then continue with Step 2. Otherwise, set q_{i+1} to be $q_i \setminus e$.
3. If $q_i \in \mathsf{GHW}(1)$, for some i, then accept and output q_i.

For Step 2, let $\mathcal{P}$ be an enumeration of the pairs (u, v) such that u, v are adjacent in q and $q \to_1 q_u \# q_v$. Suppose $\hat{q}, \hat{q}'$ are CQs and X, Y are set of variables from $\hat{q}$ and $\hat{q}'$, respectively. We denote by $(\hat{q}, X) \to_1 (\hat{q}', Y)$ the fact that Duplicator has a winning strategy in the existential 1-cover game on $\hat{q}$ and $\hat{q}'$ with the property that whenever Spoiler places a pebble on an element of X in $\hat{q}$, then Duplicator responds with some element of Y in $\hat{q}'$. It is easy to see that whether $(\hat{q}, X) \to_1 (\hat{q}', Y)$ holds can be checked in polynomial time. Step 2 is as follows:

1. Let (u, v) be the first pair in $\mathcal{P}$.
2. Set q_0 to be $q_u \# q_v$.
3. While $q_i \notin \mathsf{GHW}(1)$, search for an atom e that does not mention u_u and v_v simultaneously such that $(q_i, \{u_u, v_v\}) \to_1 (q_i \setminus e, \{u_u, v_v\})$. If there is no such atom, let (u, v) be the next pair in $\mathcal{P}$ and repeat from item 2. Otherwise, set q_{i+1} to be $q_i \setminus e$.
4. If $q_i \in \mathsf{GHW}(1)$, for some i, then accept and output q_i.

It is easy to see that the described algorithm can be implemented in polynomial time. Below we argue that it is correct.

Suppose first that the algorithm, on input q, accepts with output q^*. By construction $q^* \in \mathsf{GHW}(1)$. Assume first that the algorithm accepts in the m-th iteration of Step 1, and thus $q^* = q_m$. By construction, for each $0 \leq i < m$, we have that $q_i \to_1 q_{i+1}$ and $q_{i+1} \to_1 q_i$. In particular, $q \to_1 q^*$ and $q^* \to_1 q$, and thus q^* is a $\mathsf{GHW}(1)$-overapproximation of q. Suppose now that the algorithm accepts in Step 2 for a pair $(u, v) \in \mathcal{P}$, in the m-th iteration. Again we have that $q_i \to_1 q_{i+1}$ and $q_{i+1} \to_1 q_i$, for each $0 \leq i < m$, and thus $q_u \# q_v \to_1 q^*$ and $q^* \to_1 q_u \# q_v$. Since $(u, v) \in \mathcal{P}$, it follows that $q \to_1 q_u \# q_v$, and then $q \to_1 q^*$. Using the fact that $q_u \# q_v \to q$, we have that $q^* \to_1 q$. Hence, q^* is a $\mathsf{GHW}(1)$-overapproximation of q.

It remains to show that if q has a $\mathsf{GHW}(1)$-overapproximation q' then the algorithm accepts. Since q is connected, we can assume that q' also is. Moreover, we can assume

w.l.o.g. that q' is a core. By Lemma 24, we have two cases: (1) the only endomorphism of q' is the identity, or (2) q' has two endomorphisms, namely, the identity and the swapping endomorphism for some variables u' and v'.

First suppose case (1) applies. We show that the algorithm accepts in Step 1. By definition, $q_i \to_1 q_{i+1}$ and $q_{i+1} \to_1 q_i$ (actually $q_{i+1} \to q_i$), for each $0 \leq i \leq m$, where m is the number of iteration in Step 1. It follows that $q_0 = q \to_1 q_m$ and $q_m \to_1 q$. Since the relation $\to_1$ composes, q' is a $\mathsf{GHW}(1)$-overapproximation of q_m and by using Lemma 25, q' is a subquery of q_m. Now for the sake of contradiction assume that the algorithm does not accept in Step 1. Then $q_m \notin \mathsf{GHW}(1)$ and there is no edge e in q_m such that $q_m \to_1 q_m \setminus e$. Since q' is $\mathsf{GHW}(1)$-overapproximation of q_m, we have that $q_m \to_1 q'$ and, since $q' \in \mathsf{GHW}(1)$, q' is a proper subquery of q_m. It follows that there is an edge e in q_m such that $q_m \to_1 q_m \setminus e$, which is a contradiction.

Suppose case (2) holds. In this case the algorithm accepts in Step 2. Let h be a homomorphism from q' to q, and let $u = h(u')$ and $v = h(v')$. By Lemma 25, $u \neq v$ and then u and v are adjacent. Also, by Lemma 25, q' is a subquery of $q_u \# q_v$. Since $q \to_1 q'$, it follows that $q \to_1 q_u \# q_v$, and then $(u, v) \in \mathcal{P}$. We claim that the algorithm accepts when (u, v) is chosen from $\mathcal{P}$. First, note that q' is a $\mathsf{GHW}(1)$-overapproximation of q_m. Indeed, by definition, $q_m \to q_u \# q_v$, $q_u \# q_v \to q$ (Claim 4), and $q \to_1 q'$. It follows that $q_m \to_1 q'$. On the other hand, we have that $(q', (u', v')) \to (q_u \# q_v, (u_u, v_v))$ (q' is a subquery of $q_u \# q_v$) and $(q_u \# q_v, \{u_u, v_v\}) \to_1 (q_m, \{u_u, v_v\})$. It follows that $(q', \{u', v'\}) \to_1 (q_m, \{u_u, v_v\})$, which implies that $(q', (u', v')) \to (q_m, (u_u, v_v))$ via a homomorphism g. Then q' is a $\mathsf{GHW}(1)$-overapproximation of q_m. By applying Lemma 25 to q_m, q' and g, we obtain that $(q_m, \{u_u, v_v\}) \to_1 (q', \{u', v'\})$, and g is "almost" injective. Observe that $g(z') \neq g(z'')$ for all $z' \neq u'$ in $t^{q'}_{u'}$ and $z'' \neq v'$ in $t^{q'}_{v'}$, since $\{u_u, v_v\}$ is a bridge of $\mathcal{G}(q_m)$, that is, its removal disconnect $\mathcal{G}(q_m)$. We conclude that g is injective and then q' is a subquery of q_m.

Towards a contradiction, assume that the algorithm do not accept when (u, v) is chosen from $\mathcal{P}$. Then $q_m \notin \mathsf{GHW}(1)$ and there is no edge e that does not mention both u_u, v_v such that $(q_m, \{u_u, v_v\}) \to_1 (q_m \setminus e, \{u_u, v_v\})$. Since $(q_m, \{u_u, v_v\}) \to_1 (q', \{u', v'\})$, $(q', (u', v')) \to (q_m, (u_u, v_v))$ via the injective homomorphism g and $q' \in \mathsf{GHW}(1)$, it follows that there is an edge e that does not mention both u_u, v_v such that $(q_m, \{u_u, v_v\}) \to_1 (q_m \setminus e, \{u_u, v_v\})$. This is a contradiction. ◀

Now we consider the non-connected case. A *connected component* of a CQ is a maximal connected subquery. Given a CQ q with connected components $q_1, \ldots, q_m$, the algorithm proceeds as follows:

1. Start by simplifying q: Compute a minimal subset of CQs $\mathcal{Q}$ in $\{q_1, \ldots, q_m\}$ such that for each $1 \leq i \leq m$, there is a $p \in \mathcal{Q}$ with $q_i \to_1 p$.
2. Check whether each $p \in \mathcal{Q}$ has a $\mathsf{GHW}(1)$-overapproximation p' using the algorithm described for connected CQs. If this is the case then accept and output $\bigwedge_{p \in \mathcal{Q}} p'$.

Clearly, the algorithm can be implemented in polynomial time. For the correctness, suppose first that the algorithm accepts and outputs $q' = \bigwedge_{p \in \mathcal{Q}} p'$. Then $q' \to \bigwedge_{p \in \mathcal{Q}} p \to q$. We also have that $q \to_1 \bigwedge_{p \in \mathcal{Q}} p$ (by definition of $\mathcal{Q}$), and $\bigwedge_{p \in \mathcal{Q}} p \to_1 q'$. This implies that q' is a $\mathsf{GHW}(1)$-overapproximation of q.

Suppose now that q has a $\mathsf{GHW}(1)$-overapproximation q'. Since $q \to_1 \bigwedge_{p \in \mathcal{Q}} p$ and $\bigwedge_{p \in \mathcal{Q}} p \to_1 q$, it follows that q' is also a $\mathsf{GHW}(1)$-overapproximation of $\bigwedge_{p \in \mathcal{Q}} p$. By the minimality of $\mathcal{Q}$, we have that $p \not\to_1 \hat{p}$, for each pair of distinct CQs $p, \hat{p} \in \mathcal{Q}$. Let p be a CQ in $\mathcal{Q}$. Since $p \to_1 q'$ and p is connected, it follows that $p \to_1 p^*$, where p^* is a connected

component of q'. Also, since $q' \to \bigwedge_{p\in\mathcal{Q}} p$, there is $p_0 \in \mathcal{Q}$ such that $p^* \to p_0$. In particular, $p \to_1 p_0$. It follows that $p_0 = p$, and then p^* is a $\mathsf{GHW}(1)$-overapproximation of p. We conclude that each $p \in \mathcal{Q}$ has a $\mathsf{GHW}(1)$-overpproximation, and thus the algorithm accepts.

Finally, we consider the general case that includes non-boolean queries. Let $q(\bar{x})$ be a CQ. We denote by q^B the Boolean CQ obtained from $q(\bar{x})$ by existentially quantifying the free variables $\bar{x}$. Recall that $\mathcal{G}(q)$ is the Gaifman graph of q, while $\mathcal{G}_\exists(q)$ denotes the restriction of $\mathcal{G}(q)$ to the existentially quantified variables of q. A CQ $q(\bar{x})$ is connected if $\mathcal{G}(q)$ is connected. A connected component of q is a maximal connected subquery. For a connected CQ $q(\bar{x})$, we say that $q'(\bar{x})$ is a *part* of q if it is a maximal subquery of q with $\mathcal{G}_\exists(q')$ connected.

Let $q(\bar{x})$ be a CQ. Let $q_1 \ldots, q_m$ be the connected component of q. Let $\mathcal{C}^{\mathtt{free}}$ be the CQs in $\{q_1 \ldots, q_m\}$ that contain a free variable from $\bar{x}$, and let $\mathcal{C}^\exists$ be the rest of the CQs. The algorithm proceeds as follows:

1. Simplify $q(\bar{x})$: Compute a minimal subset of CQs $\mathcal{Q}$ in $\{q_1, \ldots, q_m\}$ such that $\mathcal{C}^{\mathtt{free}} \subseteq \mathcal{Q}$ and for each $1 \leq i \leq m$, there is a $p \in \mathcal{Q}$ with $q_i^B \to_1 p^B$.
2. Check whether each $p \in \mathcal{Q}$ has a $\mathsf{GHW}(1)$-overapproximation p'. If this is the case then accept and output $\bigwedge_{p\in\mathcal{Q}} p'$. To check if $p \in \mathcal{Q}$ has a $\mathsf{GHW}(1)$-overapproximation, for a $p \in \mathcal{C}^\exists$, we simply apply the algorithm for the Boolean and connected case described previously. In case $p(\bar{z}) \in \mathcal{C}^{\mathtt{free}} \cap \mathcal{Q}$, where $\bar{z}$ are the free variables from $\bar{x}$ present in p, the algorithm does the following:
   a. Simplify $p(\bar{z})$: Compute a minimal subset $\mathcal{S}$ of the parts of $p(\bar{z})$ such that for each part $p'(\bar{z})$ of $p(\bar{z})$ there is a part $p''(\bar{z}) \in \mathcal{S}$ such that $(p', \bar{z}) \to_1 (p'', \bar{z})$.
   b. Check whether each part $p'(\bar{z}) \in \mathcal{S}$ has a $\mathsf{GHW}(1)$-oveapproximation $p'_*(\bar{z})$. If this is the case then accept and output $\bigwedge_{p'\in\mathcal{S}} p'_*(\bar{z})$.

It remains to explain how the algorithm checks the existence of $\mathsf{GHW}(1)$-overapproximations for a part $p'(\bar{z})$ of a connected CQ $p(\bar{z})$. This is done by applying an adaptation of the algorithm described for the connected and Boolean case. For $p'(\bar{z})$ and two existentially quantified variables u, v adjacent in $\mathcal{G}_\exists(p')$, we define $p'_u \# p'_v(\bar{z})$ as the CQ obtained from $p'(\bar{z})$ as follows: the free variables are $\bar{z}$ and the atoms in $p'_u \# p'_v(\bar{z})$ mentioning variables in $\bar{z}$ are exactly those in $p'(\bar{z})$. The CQ induced by the existentially quantified variables of $p'_u \# p'_v(\bar{z})$ is the Boolean CQ $p''_u \# p''_v$, where p'' is the subquery of p' induced by the existential variables. Finally, if there is an atom in p' mentioning a free variable and an existential variable w, then the same atom appears in $p'_u \# p'_v(\bar{z})$ but replacing w by its "copies", that is, by w_u or w_v, if $w = u$ or $w = v$ respectively, or by w_u and w_v, if $w \notin \{u, v\}$.

Now it is easy to check that Lemma 3–5, Claim 4 and Lemma 25 hold for the non-Boolean case, when we consider the adapted definition for $q_u \# q_v$ (exactly the same arguments apply). Using this, it is straightforward to check that the algorithm developed for Boolean and connected CQs still works for non-Boolean CQs $p'(\bar{z})$ that are parts of connected CQs.

This finishes the proof of Theorem 12.

◀

7.3.3 Size of overapproximations

Addendum to Proof of Proposition 8: For showing that q'_n is indeed a core it suffices to show that each homomorphism h from q'_n to q'_n is a bijection. Denote the substructure of q'_n induced by y_i^w, y_i^{wa}, and y_i^{wb} by C^w (where $w \in \{1,2\}^{\leq n-1}$, $i \stackrel{\text{def}}{=} |w|$, and $a, b \in \{1,2\}$ with $a + 1 = b$ modulo 2). Observe that the homomorphism h bijectively maps each C^w to

some $C^{w'}$. Now, h has to map C^{ϵ} to itself, since no other C^{w} can be mapped to C^{ϵ} (as y_0 is the only variable occurring only in one R-atom). But then h has to either map C^1 and C^2 to themselves or to swap them. Inductively continuing this argument, one can show that h induces a bijection on the structures C^w, and is therefore a bijection from q'_n to q'_n.

7.3.4 Beyond acyclicity

Theorem 13. *Fix $k \geq 1$ and let $q(\bar{x})$ be a CQ. Then q has a $\mathsf{GHW}(k)$-overapproximation iff there is an integer $c \geq 0$ such that for every database $\mathcal{D}$ and tuple $\bar{a}$ in $\mathcal{D}$:*

$$(q, \bar{x}) \to_k (\mathcal{D}, \bar{a}) \iff (q, \bar{x}) \to_k^c (\mathcal{D}, \bar{a}).$$

Proof. For the direction from right to left, recall from the proof of Proposition 2 that for every integer $c \geq 0$ there is a CQ $q'_c(\bar{x})$ in $\mathsf{GHW}(k)$ such that for every database $\mathcal{D}$ and tuple $\bar{a}$ in $\mathcal{D}$:

$$(q'_c, \bar{x}) \to (\mathcal{D}, \bar{a}) \iff (q, \bar{x}) \to_k^c (\mathcal{D}, \bar{a}).$$

By hypothesis, it is then the case that for every database $\mathcal{D}$ and tuple $\bar{a}$ in $\mathcal{D}$:

$$(q'_c, \bar{x}) \to (\mathcal{D}, \bar{a}) \iff (q, \bar{x}) \to_k (\mathcal{D}, \bar{a}).$$

Then, $(q'_c, \bar{x})$ can be seen to be a $\mathsf{GHW}(k)$-overapproximation of $(q, \bar{x})$ by choosing $(D, \bar{a})$ as $(q, \vec{x})$ and as $(q'_c, \bar{x})$, respectively. the claim follows.

Now we prove the direction from left to right. Let q'' be a $\mathsf{GHW}(k)$-overapproximation of q. From Theorem 6 we obtain that for every (finite) database $\mathcal{D}$ and tuple $\bar{a}$ in $\mathcal{D}$:

$$(q'', \bar{x}) \to (\mathcal{D}, \bar{a}) \iff (q, \bar{x}) \to_k (\mathcal{D}, \bar{a}).$$

As a first step we prove that this continues being true for countably infinite databases. We do so by refining the proof of Theorem 6. Let $\mathcal{D}$ be a countably infinite database. Assume first that $(q, \bar{x}) \to_k (\mathcal{D}, \bar{a})$. Due to the fact that q'' is a $\mathsf{GHW}(k)$-overapproximation of q, we have that $(q'', \bar{x}) \to (q, \bar{x})$. Proposition 6 then implies that $(q'', \bar{x}) \to (\mathcal{D}, \bar{a})$ since q'' is in $\mathsf{GHW}(k)$.

Assume, on the other hand, that $(q'', \bar{x}) \to (\mathcal{D}, \bar{a})$. Suppose, for the sake of contradiction, that $(q, \bar{x}) \not\to_k (\mathcal{D}, \bar{a})$. Proposition 6 then establishes that there is a CQ $q^*(\bar{x})$ in $\mathsf{GHW}(k)^\infty$ such that:

$$(q^*, \bar{x}) \to (q, \bar{x}), \text{ but } (q^*, \bar{x}) \not\to (\mathcal{D}, \bar{a}).$$

Therefore, $q \subseteq q^*$ since this direction of Equation 1 continues being true when q is finite. This implies that $(q^*, \bar{x}) \to (q'', \bar{x})$ since Corollary 2 continues being true for CQs in $\mathsf{GHW}(k)^\infty$. We conclude that $(q^*, \bar{x}) \to (\mathcal{D}, \bar{a})$ since $(q'', \bar{x}) \to (\mathcal{D}, \bar{a})$. This is a contradiction.

Therefore, if $q'(\bar{x})$ is the CQ given by Lemma 8 for CQ $q(\bar{x})$, then for every countable database $\mathcal{D}$ and tuple $\bar{a}$ in $\mathcal{D}$:

$$(q', \bar{x}) \to (\mathcal{D}, \bar{a}) \iff (q, \bar{x}) \to_k (\mathcal{D}, \bar{a}) \iff (q'', \bar{x}) \to (\mathcal{D}, \bar{a}).$$

In particular, $(q', \bar{x}) \to (q'', \bar{x})$ and $(q'', \bar{x}) \to (q', \bar{x})$. Moreover, by composing the first homomorphism with the second one, we obtain that there is a homomorphism from $(q', \bar{x})$ to $(q'_{\text{fin}}, \bar{x})$, where q'_{fin} is a finite subset of the atoms of q'.

It is easy to prove that any finite subset of the atoms of q' must be contained (in usual set theoretical terms) in the atoms of a CQ in $\mathsf{GHW}(k)$ of the form q'_c, for a constant $c \geq 0$,

i.e., the one that describes the first c rounds of the existential k-cover game played from $(q, \bar{x})$. That is, there exists a $c \geq 0$ such that $(q', \bar{x}) \to (q'_c, \bar{x})$. On the other hand, it is also easy to prove that $(q'_c, \bar{x}) \to (q', \bar{x})$. (As is to be expected, since q' describes the full spoiler strategy in the existential pebble game, while q'_c does it for the first c rounds only). We conclude that for every database $\mathcal{D}$ and tuple $\bar{a}$ in $\mathcal{D}$:

$$(q', \bar{x}) \to (\mathcal{D}, \bar{a}) \iff (q'_c, \bar{x}) \to (\mathcal{D}, \bar{a}).$$

In other words, over every database $\mathcal{D}$ and tuple $\bar{a}$ of elements in $\mathcal{D}$,

$$(q, \bar{x}) \to_{k+1} (\mathcal{D}, \bar{a}) \iff (q, \bar{x}) \to^{c}_{k+1} (\mathcal{D}, \bar{a}).$$

This finishes the proof of the theorem. ◀

7.4 Proofs for Section "Beyond under and overapproximations: Δ-approximations"

Proposition 9. *Fix $k \geq 1$. Let q, q' be CQs such that $q' \in \mathcal{C}$. If $q \subseteq q'$ (resp., $q' \subseteq q$), then q' is a $\mathsf{GHW}(k)$-Δ-approximation of q if and only if q' is a $\mathsf{GHW}(k)$-overapproximation (resp., underapproximation) of q.*

Proof. Suppose first that q' is a $\mathsf{GHW}(k)$-Δ-approximation of q. Assume, towards a contradiction, that there is a query q'' such that $q \subseteq q'' \subset q'$. Then $q(\mathcal{D}) \subseteq q''(\mathcal{D}) \subseteq q'(\mathcal{D})$ for all databases $\mathcal{D}$ and there is a database $\mathcal{D}^*$ such that $q'(\mathcal{D}^*) \not\subseteq q''(\mathcal{D}^*)$. In particular $\Delta(q(\mathcal{D}), q''(\mathcal{D})) \subseteq \Delta(q(\mathcal{D}), q'(\mathcal{D}))$ for all databases $\mathcal{D}$ and $\Delta(q(\mathcal{D}^*), q'(\mathcal{D}^*)) \not\subseteq \Delta(q(\mathcal{D}^*), q''(\mathcal{D}^*))$. This is a contradiction to q' being a $\mathsf{GHW}(k)$-Δ-approximation of q.

Now suppose that q' is a $\mathsf{GHW}(k)$-overapproximation of q. Assume, towards a contradiction, that there is a query q'' such that $\Delta(q(\mathcal{D}), q''(\mathcal{D})) \subseteq \Delta(q(\mathcal{D}), q'(\mathcal{D}))$ for all databases $\mathcal{D}$ and $\Delta(q(\mathcal{D}^*), q'(\mathcal{D}^*)) \not\subseteq \Delta(q(\mathcal{D}^*), q''(\mathcal{D}^*))$ for some database $\mathcal{D}^*$. Then $q \subseteq q'' \subseteq q'$ as $\Delta(q(\mathcal{D}), q'(\mathcal{D}))$ may only contain tuples in $q'(\mathcal{D})$ but not in $q(\mathcal{D})$. Moreover $q'(\mathcal{D}^*) \not\subseteq q''(\mathcal{D}^*)$. This is a contradiction to q' being a $\mathsf{GHW}(k)$-overapproximation of q.

The argument for underapproximations is analogous. ◀

7.4.1 Incomparable $\mathsf{GHW}(k)$-Δ-approximations

Lemma 15. *Fix $k \geq 1$. Let $q(\bar{x}), q'(\bar{x}), q''(\bar{x})$ be CQs such that $q'' \in \mathsf{GHW}(k)$. Suppose that $(q, \bar{x}) \to_k (q', \bar{x})$. Then $(q'', \bar{x}) \to (q' \wedge q, \bar{x})$ implies $(q'', \bar{x}) \to (q', \bar{x})$.*

Proof. Before proving the lemma, we need some terminology and claims. Let $\mathcal{D}$ be a database and $(A_1, \ldots, A_n)$ be a tuple of pairwise-disjoint subsets of elements of $\mathcal{D}$, where $n \geq 0$. Let also be $\mathcal{D}'$ be a database and $(a_1, \ldots, a_n)$ a tuple of elements in $\mathcal{D}'$. Then we write $(\mathcal{D}, (A_1, \ldots, A_n)) \to (\mathcal{D}', (a_1, \ldots, a_n))$ iff there is a homomorphism h from $\mathcal{D}$ to $\mathcal{D}'$ such that, for each $i \in \{1, \ldots, n\}$ and $a \in A_i$, it is the case that $h(a) = a_i$.

For such a pair $(\mathcal{D}, (A_1, \ldots, A_n))$, with $n \geq 0$, we define its generalized hypertreewidth in the natural way. The intuition is that we see $(\mathcal{D}, (A_1, \ldots, A_n))$ as a "query", where $A_1 \cup \cdots \cup A_n$ are the "free variables" and the rest of the elements the "existential variables". Formally, a *tree decomposition* of $(\mathcal{D}, (A_1, \ldots, A_n))$ is a pair (T, χ), where T is a tree and χ is a mapping that assigns a subset of the elements in $\mathcal{D} \setminus A_1 \cup \cdots \cup A_n$ to each node $t \in T$, such that:

1. For each atom $R(\bar{a})$ in $\mathcal{D}$, it is the case that $\bar{a} \cap (\mathcal{D} \setminus A_1 \cup \cdots \cup A_n)$ is contained in $\chi(t)$, for some $t \in T$.
2. For each element a in $\mathcal{D} \setminus A_1 \cup \cdots \cup A_n$, the set of nodes $t \in T$ for which a occurs in $\chi(t)$ is connected.

The *width* of node t in (T, χ) is the minimal number ℓ for which there are ℓ atoms in $\mathcal{D}$ covering $\chi(t)$, i.e., atoms $R(\bar{a}_1), \ldots, R(\bar{a}_\ell)$ in such that $\chi(t) \subseteq \bigcup_{1 \leq i \leq \ell} \bar{a}_i$ The width of (T, χ) is the maximal width of the nodes of T. The *generalized hypertreewidth* of $(\mathcal{D}, (A_1, \ldots, A_n))$ is the minimum width of its tree decompositions.

Using the same argument as in the proof of the forward implication of Proposition 5, we can show the following:

▶ **Claim 5.** *Fix $k \geq 1$. Let $q(\bar{x}), q'(\bar{x})$ be CQs, where $\bar{x} = (x_1, \ldots, x_n)$, for $n \geq 0$. Suppose that $(q, \bar{x}) \to_k (q', \bar{x})$. Then, for each database $\mathcal{D}$ and tuple $(A_1, \ldots, A_n)$ of subsets of $\mathcal{D}$ such that $(\mathcal{D}, (A_1, \ldots, A_n))$ has generalized hypertreewidth at most k, it is the case that $(\mathcal{D}, (A_1, \ldots, A_n)) \to (q, (x_1, \ldots, x_n))$ implies $(\mathcal{D}, (A_1, \ldots, A_n)) \to (q', (x_1, \ldots, x_n))$.*

Proof Sketch. Let $\mathcal{H}$ be a winning strategy of Duplicator witnessing the fact that $(q, \bar{x}) \to_k (q', \bar{x})$. Let $(\mathcal{D}, (A_1, \ldots, A_n))$ have generalized hypertreewidth at most k and to be such that $(\mathcal{D}, (A_1, \ldots, A_n)) \to (q, (x_1, \ldots, x_n))$ via a homomorphism h. Then we can compose h with the strategy $\mathcal{H}$ to define a homomorphism g witnessing $(\mathcal{D}, (A_1, \ldots, A_n)) \to (q', (x_1, \ldots, x_n))$. The mapping g is defined in a top-down fashion over the tree decomposition (T, χ) of width at most k of $(\mathcal{D}, (A_1, \ldots, A_n))$. One starts at the root r of T, and forces Spoiler to play his pebbles over the set $h(\chi(r))$. If Duplicator responds according to $\mathcal{H}$ with a partial homomorphism f_r, we then let $g(a) = f_r(h(a))$, for each $a \in \chi(r)$. We then move to each child of r and so on, until all leaves are reached and g is defined over all elements in $\mathcal{D} \setminus A_1 \cup \cdots \cup A_n$. Since Duplicator responds to Spoiler's moves with consistent partial homomorphisms, g is actually a well-defined homomorphism from $(\mathcal{D}, (A_1, \ldots, A_n))$ to $(q', (x_1, \ldots, x_n))$. ◀

Now we are ready to show our lemma. Suppose that $(q, \bar{x}) \to_k (q', \bar{x})$, where $\bar{x} = (x_1, \ldots, x_n)$, for some $n \geq 0$. Assume that $(q'', \bar{x}) \to (q' \wedge q, \bar{x})$ via a homomorphism h, for $q''(\bar{x}) \in \mathsf{GHW}(k)$. For each $i \in \{1, \ldots, n\}$, we define V_i to be the set of variables x in q'' such that $h(x) = x_i$. In particular, $x_i \in V_i$, for each $i \in \{1, \ldots, n\}$. We define V to be the set of variables x in q'' such that $h(x) = y$, where y is an existentially quantified variable of q. Similarly, we define V' with respect to the existentially quantified variables of q'. Note that the sets $V, V', V_1, \ldots, V_n$ form a partition of the variables of q''.

Let $\mathcal{D}_{q''}$ be the canonical database of q''. Since $q'' \in \mathsf{GHW}(k)$, we know that $(\mathcal{D}_{q''}, (\{x_1\}, \ldots, \{x_n\}))$ has generalized hypertreewidth at most k, as defined above. Let $\mathcal{D}_V$ be the database induced in $\mathcal{D}_{q''}$ by the set of variables $V \cup V_1 \cup \cdots \cup V_n$, i.e., the set of atoms $R(\bar{t}) \in \mathcal{D}_{q''}$ such that each element in $\bar{t}$ is in $V \cup V_1 \cup \cdots \cup V_n$. We now show that $(\mathcal{D}_V, (V_1, \ldots, V_n))$ has also generalized hypertreewidth at most k. Indeed, let (T, χ) be the tree decomposition of $(\mathcal{D}_{q''}, (\{x_1\}, \ldots, \{x_n\}))$ of width at most k. Define χ' such that for each $t \in T$, we have that $\chi'(t) = \chi(t) \cap V$. We claim that (T, χ') is a tree decomposition of $(\mathcal{D}_V, (V_1, \ldots, V_n))$ of width at most k. Since (T, χ) is tree decomposition, we have that, for each $a \in V$, it is the case that the set $\{t \in T \mid a \in \chi'(t)\}$ is connected; and for each atom $R(\bar{a}) \in \mathcal{D}_V$, there is a node $t \in T$ such that $\bar{a} \cap V \subseteq \chi'(t)$. To see that the width of (T, χ') is no more than k, let t be a node in T. Since the width of (T, χ) is at most k, there are ℓ atoms $R(\bar{a}_1), \ldots, R(\bar{a}_\ell)$ in $\mathcal{D}_{q''}$, with $\ell \leq k$, such that $\chi(t) \subseteq \bigcup_{1 \leq i \leq \ell} \bar{a}_i$. Let $R(\bar{a}_{i_1}), \ldots, R(\bar{a}_{i_p})$, where $1 \leq i_1 < \cdots < i_p \leq \ell$ and $p \leq \ell$, be the atoms in $\{R(\bar{a}_1), \ldots, R(\bar{a}_\ell)\}$ that contain an element in $\chi'(t)$. Since $\chi'(t) \subseteq \chi(t)$, it is the case that $\chi'(t) \subseteq \bigcup_{1 \leq j \leq p} \bar{a}_{i_j}$. It suffices to show

that each $R(\bar{a}_{i_j})$ is actually an atom in $\mathcal{D}_V$, for $1 \leq j \leq p$. Towards a contradiction, suppose that this is not the case. Then, there is an atom in $\mathcal{D}_{q''}$ that contains simultaneously one variable in $\chi'(t) \subseteq V$ and one variable in V'. By the definitions of V' and V, and the fact that h is a homomorphism, it follows that there is an atom in $(q' \wedge q)$ that mentions simultaneously one existentially quantified variable from q' and one from q; this contradicts the definition of $(q' \wedge q)$. We conclude that the generalized hypertreewidth of $(\mathcal{D}_V, (V_1, \ldots, V_n))$ is no more than k.

Recall that h is our initial homomorphism from $(q'', \bar{x})$ to $((q' \wedge q), \bar{x})$. Let h_V be the restriction of h to the set $V \cup V_1 \cup \cdots \cup V_n$. By construction, $(\mathcal{D}_V, (V_1, \ldots, V_n)) \to (q, (x_1, \ldots, x_n))$ via h_V. We can then apply Claim 5 and obtain that $(\mathcal{D}_V, (V_1, \ldots, V_n)) \to (q', (x_1, \ldots, x_n))$ via a homomorphism h'. We define our required homomorphism g from $(q'', \bar{x})$ to $(q', \bar{x})$ as follows: if $a \in V \cup V_1 \cup \cdots \cup V_n$, then $g(a) = h'(a)$; otherwise, if $a \in V'$, then $g(a) = h(a)$. To see that g is a homomorphism, it suffices to consider an atom $R(\bar{a}) \in \mathcal{D}_{q''}$ such that $\bar{a}$ contains an element in V' and one element not in V', and show that $R(g(\bar{a})) \in \mathcal{D}_{q'}$. Let A be the set of elements in $\bar{a}$ that are not in V'. As mentioned above, there are no atoms in $\mathcal{D}_{q''}$ mentioning elements in V' and V simultaneously, thus $A \subseteq V_1 \cup \cdots \cup V_n$. In particular, $h(a) = h'(a)$, for each $a \in A$. It follows that $R(g(\bar{a})) = R(h(\bar{a}))$, and thus $R(g(\bar{a})) \in \mathcal{D}_{q'}$. ◀

Proposition 10. *There is a CQ with infinitely many (non-equivalent) incomparable* $\mathsf{GHW}(1)$*-*Δ*-approximations. In fact, this holds for the CQ* q *in Figure 1.*

Proof. Consider the Boolean CQ q from Figure 1, defined as

$$q = \exists x \exists y \exists z P_a(x, y) \wedge P_a(y, x) \wedge P_a(y, z) \wedge P_a(z, y) \wedge P_b(z, x) \wedge P_b(x, z)$$

and the CQ q' from the same figure defined by

$$q' = \exists x \exists y_1 \exists y_2 \exists z P_a(x, y_1) \wedge P_a(y_1, x) \wedge P_a(y_2, z) \wedge P_a(z, y_2) \wedge P_b(z, x) \wedge P_b(x, z)$$

For each $n \geq 1$, we define the CQ

$$q_n = \exists x_1 \cdots \exists x_{n+1} P_a(x_1, x_2) \wedge \cdots \wedge P_a(x_n, x_{n+1}) \wedge P_b(x_1, x_1) \wedge P_b(x_{n+1}, x_{n+1})$$

Observe that $q' \wedge q_n \in \mathsf{GHW}(1)$, for each $n \geq 1$. We now show that, for each $n \geq 1$, $q' \wedge q_n$ is an incomparable $\mathsf{GHW}(1)$-Δ-approximation of q. As mentioned in Example 4, we have that $q \to_1 q'$. In particular $q \to_1 q' \wedge q_n$. Clearly, $q \not\to q' \wedge q_n$. Also, $q_n \not\to q$ since variables x_1 and x_{n+1} of q_n cannot be mapped to any variable in q via a homomorphism. Therefore, $q' \wedge q_n \not\to q$. By Theorem 14, it follows that $q' \wedge q_n$ is incomparable $\mathsf{GHW}(1)$-Δ-approximation of q.

Now we show that the CQs $\{q' \wedge q_n\}_{n \geq 1}$ form a family of non-equivalent CQs. First note that $q_n \not\to q'$, for each $n \geq 1$. Also, observe that $q_i \to q_j$ iff $i = j$, for $i, j \geq 1$. It follows that for each $i, j \geq 1$ such that $i \neq j$, it is the case that $q' \wedge q_i \not\to q' \wedge q_j$ and $q' \wedge q_j \not\to q' \wedge q_i$. In particular, $\{q' \wedge q_n\}_{n \geq 1}$ is a family of non-equivalent CQs. ◀

Proposition 11. *Fix* $k \geq 1$*. Checking if a given CQ* $q' \in \mathsf{GHW}(k)$ *is an incomparable* $\mathsf{GHW}(k)$*-*Δ*-approximation of a given CQ* q*, is* coNP*-complete.*

Proof. As already mentioned, the coNP upper bound follows directly from Theorem 14. For the lower bound, we consider the Non-Hom(H) problem, for a fixed directed graph H, which asks, given a directed graph G, whether $G \not\to H$. Let us assume that, for each $k \geq 1$, there is a directed graph H_k such that:

1. $H_k \in \mathsf{GHW}(k)$, or more formally, the Boolean CQ q_{H_k} whose canonical database is H_k belongs to $\mathsf{GHW}(k)$.
2. Non-Hom(H_k) is coNP-complete even when the input directed graph G satisfies that $H_k \not\to G$.

We later explain how to obtain these graphs H_k's. Now reduce from the restricted version of Non-Hom(H_k). Let G be a directed graph such that $H_k \not\to G$. We first check in polynomial time whether $G \to_k H_k$. If $G \not\to_k H_k$, we output a fixed pair q_0, q_0' such that $q_0 \in \mathsf{GHW}(k)$ and q_0' is an incomparable $\mathsf{GHW}(k)$-Δ-approximation of q_0. In the case that $G \to_k H_k$, we output the pair q_G, q_{H_k}, where q_G and q_{H_k} are Boolean CQs whose canonical databases are precisely G and H_k, respectively. Since $q_{H_k} \in \mathsf{GHW}(k)$ by item (1) above, the reduction is well-defined.

Suppose first that $G \not\to H_k$. If $G \not\to_k H_k$, then we are done, since q_0' is an incomparable $\mathsf{GHW}(k)$-Δ-approximation of q_0. Otherwise, if $G \to_k H_k$, since $G \not\to H_k$ and $H_k \not\to G$ (item (2) above), Theorem 14 implies that q_{H_k} is an incomparable $\mathsf{GHW}(k)$-Δ-approximation of q_G. On the other hand, assume that $G \to H_k$. In particular, we have that $G \to_k H_k$, and then, in this case, the reduction outputs the pair q_G, q_{H_k}. Since $G \to H_k$ and Theorem 14, we conclude that q_{H_k} is not an incomparable $\mathsf{GHW}(k)$-Δ-approximation of q_G.

It remains to define the directed graph H_k. If $k \geq 2$, it suffices to consider the clique on $2k$ vertices, that is, the directed graph $\vec{K}_{2k}$ whose vertex set is $\{1, \ldots, 2k\}$ and whose edges are $\{(i,j) \mid i \neq j, \text{ for } i,j \in \{1,\ldots,2k\}\}$. We have that $\vec{K}_{2k} \in \mathsf{GHW}(k)$, and thus item (1) above is satisfied. Also, we can reduce from the non-$2k$-colorability problem by replacing each undirected edge $\{u,v\}$ of a given undirected graph G, by a one directed edge in an arbitrary direction, e.g., from u to v. Clearly, this is a reduction from non-$2k$-colorability to Non-Hom($\vec{K}_{2k}$). Also note that the output $f(G)$ of the reduction satisfies that $\vec{K}_{2k} \not\to f(G)$, as $f(G)$ has no directed loops nor parallel edges. Therefore, item (2) above is satisfied. For $k = 1$, it is known from [20] that there is a *oriented tree* T (i.e., a directed graph whose underlying undirected graph is acyclic and has no loops nor parallel edges) such that Non-Hom(T) is coNP-complete. Since T is an oriented tree then it belongs to $\mathsf{GHW}(1)$, and then item (1) is satisfied. Also, by inspecting the reduction in [20], we have that item (2) also holds. ◀