A More General Theory of Static Approximations for Conjunctive Queries

Pablo Barceló
Millenium Institute for Foundational Research on Data, DCC, University of Chile, Santiago, Chile
pbarcelo@dcc.uchile.cl

Miguel Romero
University of Oxford, Oxford, UK
miguel.romero@cs.ox.ac.uk

Thomas Zeume
TU Dortmund, Dortmund, Germany
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.

2012 ACM Subject Classification Information systems → Structured Query Language, Theory of computation → Database query languages (principles), Theory of computation → Database theory → Database query processing and optimization (theory)

Keywords and phrases conjunctive queries, hypertreewidth, approximations, pebble games

Digital Object Identifier 10.4230/LIPIcs.ICDT.2018.7

Acknowledgements Barceló and Romero have been funded by Millennium Nucleus Center for Semantic Web Research under Grant NC120004. Barceló has also been funded by Fondecyt grant 1170109. Zeume acknowledges the financial support by the European Research Council (ERC), grant agreement No 683080. Romero and Zeume thank the Simons Institute for the Theory of Computing for hosting them.

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. [15, 20, 24, 12, 13]). 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 notions corresponds to *generalized hypertreewidth* [17]. Notably, the classes of CQs of bounded generalized hypertreewidth can be evaluated in polynomial time (see [16] 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 $\mathcal{D}$ and an approximation that has been computed once can be used for all databases. 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 CQs 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 then correspond 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 underapproximations must be contained in the original query, and overapproximations must contain it, symmetric difference approximations do not impose any constraint on approximations with respect to the partial order $\subseteq$. Thus, 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 [5]. Indeed, it is known that for each $k \geq 1$ the $\mathsf{GHW}(k)$-underapproximations have good properties that justify their application: (a) they always exist, and (b) evaluating all $\mathsf{GHW}(k)$-underapproximations of a CQ q over a database $\mathcal{D}$ is *fixed-parameter tractable* with the size of q as parameter. This is an improvement over general CQ evaluation for which the latter is believed not to hold [26].

The notions of $\mathsf{GHW}(k)$-overapproximations and $\mathsf{GHW}(k)$-Δ-approximations, while already introduced in [4], are much less understood. No general tools have been identified so far 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. It has also been observed that 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 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* [22]. Such games have been used to show that CQs of bounded width can be evaluated efficiently [11, 9]. 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 us to optimize q' before evaluating it. There is also a difference in complexity, as existential pebble game techniques are PTIME-complete in general [21], and thus inherently sequential, while evaluation of CQs in $\mathsf{GHW}(k)$ is highly parallelizable (Gottlob et al. [17]).

Table 1 Summary of results on under- and overapproximations of bounded generalized hypertreewidth. The complexity of identification coincides with that of evaluation in both cases. New results are marked with (*). All remaining results follow from [4, 5].

	Existence?	**Unique?**	**Evaluation**	**Existence check**
GHW(k)-*underapp.*	always	not always	NP-hard	N/A
GHW(k)-*overapp.*	not always	always	PTIME*	For $k = 1$: 2EXPTIME*; PTIME* on binary schemas; For $k > 1$: Open

By exploiting automata techniques, we show that checking if a CQ q has a (finite) GHW(1)-overapproximation q' is in 2EXPTIME. Also, when such q' exists it can be computed in 3EXPTIME. This is important since GHW(1) coincides with the well-known class of *acyclic* CQs [27]. If the arity of the schema is fixed, these bounds drop to 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, 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 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 [25, 7].
Table 1 shows a summary of these results in comparison with underapproximations.

Our contributions for GHW(k)-Δ-approximations are as follows. As a preliminary step, we show that GHW(k)-under and GHW(k)-overapproximations are particular cases of GHW(k)-Δ-approximations, but not vice versa. Afterwards, as for GHW(k)-overapproximations, we provide a link between GHW(k)-Δ-overapproximations and the existential pebble game, and use it to characterize when a CQ q has at least one GHW(k)-Δ-approximation that is neither a GHW(k)-underapproximation nor a GHW(k)-overapproximation (a so-called *incomparable* 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 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 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 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.

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., those 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 $1 \leq i \leq m$. The *result of* q *over* $\mathcal{D}$, denoted $q(\mathcal{D})$, is 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* [17], 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 [16].

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 $q'' \not\sqsubseteq_q q'$.

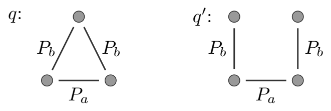

Figure 1 The CQ q and its $\mathsf{GHW}(1)$-overapproximation q' from Example 2.

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

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 2.** 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

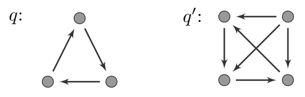

Figure 2 The CQ q is in $\mathsf{GHW}(2)$ but has no $\mathsf{GHW}(1)$-overapproximations, while q' is in $\mathsf{GHW}(3)$ but has no $\mathsf{GHW}(\ell)$-overapproximations for $\ell \in \{1,2\}$.

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

As shown in [4], existence of overapproximations is not a general phenomenon. 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 21, we can strengthen this further:

▶ **Proposition 3.** *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 4.** *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 that there is a CQ $q'' \in \mathsf{GHW}(k)$ such that $q \subseteq q''$ but $q' \not\subseteq q''$. Note that we can assume that q' and q'' have the same free variables $\bar{x}$; otherwise we can rename them accordingly. 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''. The tuple of free variables of $(q' \wedge q'')$ is $\bar{x}$. Observe 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 $q' \subseteq (q' \wedge q'')$. On the other hand, we have that $(q' \wedge q'') \subseteq q''$, and then $q' \subseteq q''$. This is a contradiction. ◀

As a corollary, we immediately obtain the following:

▶ **Corollary 5.** *If a CQ q has $\mathsf{GHW}(k)$-overapproximations q_1 and q_2, then $q_1 \equiv q_2$.*

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

3.2 A link with the existential pebble game

We characterize $\mathsf{GHW}(k)$-overapproximations 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 $c_1, \dots, c_\ell$ ($\ell \leq p$) are the elements marked by Spoiler's pebbles in $\mathcal{D}$, there must be a set of 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 no more than k atoms).

Duplicator wins if she has a *winning strategy*, i.e., she can indefinitely continue playing the game in such a way that after each round, if $c_1, \dots, c_\ell$ are the elements that are marked by Spoiler's pebbles in $\mathcal{D}$ and $d_1, \dots, d_\ell$ are the elements marked by the corresponding pebbles of Duplicator in $\mathcal{D}'$, then $\big((c_1, \dots, c_\ell, \bar{a}), (d_1, \dots, d_\ell, \bar{b})\big)$ 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 $(c_1, \dots, c_\ell, \bar{a})$, 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 $(d_1, \dots, d_\ell, \bar{b})$. We write $(\mathcal{D}, \bar{a}) \to_k (\mathcal{D}', \bar{b})$ if Duplicator has a winning strategy.

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

▶ **Proposition 6.** *[9] Fix $k \geq 1$. Checking whether $(\mathcal{D}, \bar{a}) \to_k (\mathcal{D}', \bar{b})$ is in polynomial time.*

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

▶ **Proposition 7.** *[9] 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 $(q, \bar{x}) \to (\mathcal{D}', \bar{b})$.*

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

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

That is, the "approximation" of $\to$ provided by $\to_k$ is sufficient for evaluating CQs in $\mathsf{GHW}(k)$. Together with Proposition 6, 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 8.** *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 $(q', \bar{x}) \to_k (q, \bar{x})$ since $\to \subset \to_k$. We prove now that $(q, \bar{x}) \to_k (q', \bar{x})$. From Proposition 7, 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 4.

Assume now that $(q', \bar{x}) \to_k (q, \bar{x})$ and $(q, \bar{x}) \to_k (q', \bar{x})$. Since $q' \in \mathsf{GHW}(k)$, we have that $q \subseteq q'$ by Equation (2). From Proposition 7, 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 $q \subseteq q'' \subset q'$. Hence q' is a $\mathsf{GHW}(k)$-overapproximation. ◀

▶ **Example 9.** (Example 2 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 GHW(k)-overapproximations

A direct corollary of Proposition 6 and Theorem 8 is that the *identification* problem for $\mathsf{GHW}(k)$-overapproximations is in polynomial time:

▶ **Corollary 10.** *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 for every fixed $k \geq 2$ [18, 14].

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 11.** *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 $\bar{a}$:*

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

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 $(q', \bar{x}) \to (q, \bar{x})$. Since winning strategies for Duplicator compose and $\to \subset \to_k$ 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 8, we have that $(q, \bar{x}) \to_k (q', \bar{x})$. By composition and $\to \subset \to_k$, it follows that $(q, \bar{x}) \to_k (\mathcal{D}, \bar{a})$ holds. ◀

As a corollary to Theorem 11 and Proposition 6 we obtain:

▶ **Corollary 12.** *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 $\mathsf{GHW}(k)$-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 is countably infinite. 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. Similarly, the generalized hypertreewidth of an infinite CQ is defined as in the finite case, but now tree decompositions can be infinite. We write $\mathsf{GHW}(k)^\infty$ for the class of all CQs, finite and 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 13.** *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 $\mathcal{D}$:*

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

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

The proof of this result follows from techniques in [22]. The basic idea is that q' has an (infinite) generalized hypertree decomposition of width k that represents all possible moves of Spoiler in the existential k-cover game played from q.

Since we now deal with infinite CQs and databases, we cannot apply Proposition 7 directly in our analysis of $\mathsf{GHW}(k)^\infty$-overapproximations. Instead, we use the following suitable reformulation of it, which we obtain by inspection of its proof:

▶ **Proposition 14.** *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 $(q, \bar{x}) \to (\mathcal{D}', \bar{b})$.*

$\mathsf{GHW}(k)^\infty$-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 15.** *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 13, 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 13). 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 13). Proposition 14 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 6 and the proof of Theorem 15 that they can be evaluated efficiently:

▶ **Corollary 16.** *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.*

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 a finite overapproximation. We start with the case $k = 1$, which we show to be decidable in 2EXPTIME (we do not know if this is optimal). For $k > 1$ we leave the decidability open, but provide some explanation about 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 17.** *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 18.** *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 a tree labeled by (a) the variables $\{u_1, \dots, u_{2r}\}$ 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 8, 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 2ATA $\mathcal{A}_1$ can guess and verify a homomorphism from q' to q. 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. The set of variables appearing in an atom of q constitute a 1-*union* of q. Then $q \to_1 q'$ iff there is a non-empty 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 $\{u_1, \dots, u_{2r}\}$ 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 $\{u_1, \dots, u_{2r}\}$. 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. ◀

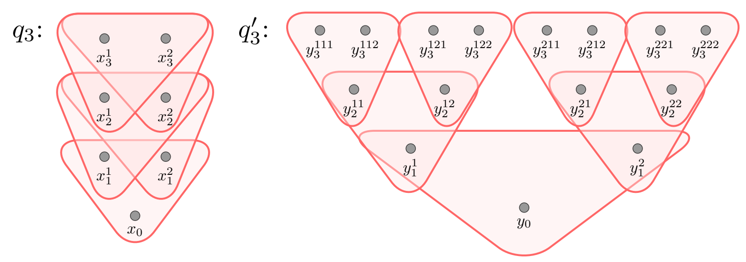
q_3:
x_3^1 x_3^2
x_2^1 x_2^2
x_1^1 x_1^2
x_0
q_3':
y_3^{111} y_3^{112} y_3^{121} y_3^{122} y_3^{211} y_3^{212} y_3^{221} y_3^{222}
y_2^{11} y_2^{12} y_2^{21} y_2^{22}
y_1^1 y_1^2
y_0

Figure 3 Illustration of the CQs q_3 and q_3' from Proposition 20. 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'.

It is easy to see how Theorem 17 follows from Proposition 18. Checking if a CQ q has a 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 GHW(1)-overapproximation of q.

The case of binary schemas. For schemas of arity two the existence and computation of 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 GHW(1) coincides with the class of CQs of treewidth one [11]. Then:

▶ **Theorem 19.** *There is a* PTIME *algorithm that checks if a CQ q over a schema of maximum arity two has a* GHW(1)*-overapproximation q', and computes such a q' if it exists.*

The proof of this result can be found in the appendix.

Size of overapproximations. Over binary schemas 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 GHW(1)-overapproximations:

▶ **Proposition 20.** *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* 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 atoms $R(y_0, y_1^1, y_1^2)$, $R(y_0, y_1^2, y_1^1)$, as well as $R(y_{|w|}^w, y_{|w|+1}^{w1}, y_{|w|+1}^{w2})$ and $R(y_{|w|}^w, y_{|w|+1}^{w2}, y_{|w|+1}^{w1})$, for each word w over $\{1, 2\}$ of length $1 \leq |w| \leq n-1$. Figure 3 depicts q_3 and q_3'.

The query q_n' indeed is an overapproximation of q_n. The mapping $h : q_n' \to q_n$ defined as $h(y_0) = x_0$ and $h(y_{|w|+1}^{wj}) = x_{|w|+1}^j$, for each word w over $\{1, 2\}$ of length $0 \leq |w| \leq n-1$ and $j \in \{1, 2\}$, is a homomorphism. On the other hand, a compact winning strategy for Duplicator can be obtained by basically "inverting" the homomorphism h.

The size of q_n' is $\Omega(2^n)$. We claim that q_n' is the smallest GHW(1)-overapproximation of q_n, which proves the proposition. A straightforward case-by-case analysis shows that q_n' is a *core* [8, 19]. Now assume, towards a contradiction, that q' is a GHW(1)-overapproximation of q_n with fewer atoms than q_n'. Then $q_n' \equiv q'$ by Corollary 5. 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 to q_n' being a core. ◀

4.2 Beyond acyclicity

Theorem 8 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 $(\mathcal{D}, \bar{a}) \rightarrow_k^c (\mathcal{D}, \bar{b})$, 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., if 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 21.** *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 $(q, \bar{x}) \rightarrow_k (\mathcal{D}, \bar{a})$ iff $(q, \bar{x}) \rightarrow_k^c (\mathcal{D}, \bar{a})$, 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 [25, 7]. The existence of winning Duplicator strategies in existential pebble games is expressible in LFP [23], yet results from this context are not directly applicable to determine the decidability status of the condition from Theorem 21.

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 22.** *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 GHW(k)-Δ-approximations

Let q be a CQ. In view of Proposition 22, 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 23.** *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 $(q, \bar{x}) \rightarrow_k (q', \bar{x})$, and both $q \not\subseteq q'$ and $q' \not\subseteq q$.*

Proof. Suppose that q' is an incomparable $\mathsf{GHW}(k)$-Δ-approximation of q and assume, by contradiction, that $(q,\bar{x}) \not\to_k (q',\bar{x})$. By Proposition 7, there is a $q'' \in \mathsf{GHW}(k)$ such that $q \subseteq q''$ and $q' \not\subseteq q''$. We show that $q' \sqsubset_q (q'' \wedge q')$, which is a contradiction as $(q'' \wedge q') \in \mathsf{GHW}(k)$. Assume that $\bar{a} \in \Delta(q(\mathcal{D}), (q'' \wedge q')(\mathcal{D}))$, for some $\mathcal{D}$ and $\bar{a} \in \mathcal{D}$. If $\bar{a} \notin q(\mathcal{D})$, then $\bar{a} \in (q'' \wedge q')(\mathcal{D}) \subseteq q'(\mathcal{D})$, and thus, $\bar{a} \in \Delta(q(\mathcal{D}), q'(\mathcal{D}))$. Otherwise, $\bar{a} \in q(\mathcal{D})$ and $\bar{a} \notin (q'' \wedge q')(\mathcal{D})$. Since $q \subseteq q''$, we have $\bar{a} \notin q'(\mathcal{D})$, and then $\bar{a} \in \Delta(q(\mathcal{D}), q'(\mathcal{D}))$. Hence $q' \sqsubseteq_q (q'' \wedge q')$. Now, 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 converse, we need the following:

▶ **Lemma.** 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 $(q,\bar{x}) \to_k (q',\bar{x})$. 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 by hypothesis, $\bar{x} \notin \Delta(q(\mathcal{D}_{(q' \wedge q)}), q''(\mathcal{D}_{(q' \wedge q)}))$. Hence, $\bar{x} \in q''(\mathcal{D}_{(q' \wedge q)})$. By the lemma above, we have $(q'',\bar{x}) \to (q',\bar{x})$, that is, $q' \subseteq q''$. For $q'' \subseteq q'$, note that $\bar{x} \notin q(\mathcal{D}_{q''})$; otherwise, $q'' \subseteq q$ would hold, implying that $q' \subseteq q$, which is a contradiction. Since $\bar{x} \in q''(\mathcal{D}_{q''})$, we have $\bar{x} \in \Delta(q(\mathcal{D}_{q''}), q''(\mathcal{D}_{q''}))$. This implies that $\bar{x} \in \Delta(q(\mathcal{D}_{q''}), q'(\mathcal{D}_{q''}))$, and then $\bar{x} \in q'(\mathcal{D}_{q''})$, i.e., $q'' \subseteq q'$. Hence, $q' \equiv q''$. ◀

▶ **Example 24.** Consider again the CQ $q = \exists x \exists y \exists z (E(x,y) \wedge E(y,z) \wedge E(z,x))$ from Figure 2. Then 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 23, 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 24 shows, incomparable $\mathsf{GHW}(k)$-Δ-approximations may exist for some CQs. However, in contrast with overapproximations, they are not unique in general:

▶ **Proposition 25.** *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 23 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. It suffices 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 6. This is optimal:

▶ **Proposition 26.** *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 $k > 1$. Nevertheless, for $k = 1$ we can exploit the automata techniques developed in Section 4 and obtain an analogous decidability result:

▶ **Proposition 27.** *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.*

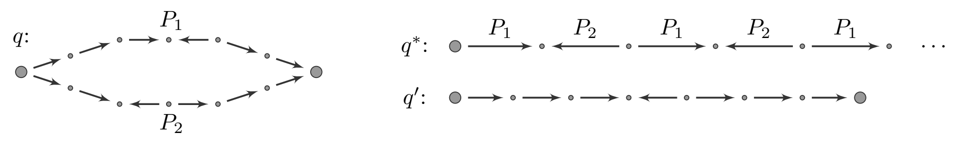

Figure 4 The CQ $q \in \mathsf{GHW}(2)$ from Example 29. The CQ $(q^* \wedge q')$ is an incomparable $\mathsf{GHW}(1)^\infty$-Δ-approximation of q. On the other hand, q has no incomparable $\mathsf{GHW}(1)$-Δ-approximations.

Now we study evaluation. Recall 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 25). 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 27 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 $O(|\mathcal{D}| \cdot |q'|)$ [27], which is $O(|\mathcal{D}| \cdot f(|q|))$, for f a triple-exponential function. This means that the evaluation of such a 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 does not hold in general for the class of all CQs [26]). Formally, then:

▶ **Theorem 28.** *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 $(q, \bar{x}) \rightarrow_k (\mathcal{D}, \bar{a})$. 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.

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 29.** Consider the CQ q that asks for the existence of the two oriented paths P_1 and P_2, as shown in Figure 4. Theorem 23 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 oriented path P_1, similarly for P_2). Also, let q' be an arbitrary CQ in $\mathsf{GHW}(1)$ which is incomparable with q (one such a q' is shown in Figure 4). Applying the extension of Theorem 23 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 29 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 $(q, \bar{x}) \rightarrow_k (\mathcal{D}, \bar{a})$ 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 30.** *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 remain open: is the existence of $\mathsf{GHW}(k)$-overapproximations decidable for $k > 1$? What is the precise complexity of checking for the existence of $\mathsf{GHW}(1)$-overapproximations? In particular, can we improve the 2EXPTIME upper bound from Theorem 17? What is an optimal upper bound on the size of $\mathsf{GHW}(1)$-overapproximations?

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 Wolfgang Fischl, Georg Gottlob, and Reinhard Pichler. General and fractional hypertree decompositions: Hard and easy cases. *CoRR*, abs/1611.01090, 2016. `arXiv:1611.01090`.

15 Minos Garofalakis and Phillip Gibbon. Approximate query processing: taming the terabytes. In *VLDB*, page 725, 2001.

16 Georg Gottlob, Gianluigi Greco, Nicola Leone, and Francesco Scarcello. Hypertree decompositions: Questions and answers. In *PODS*, pages 57–74, 2016.

17 Georg Gottlob, Nicola Leone, and Francesco Scarcello. Hypertree decompositions and tractable queries. *J. Comput. Syst. Sci.*, 64(3):579–627, 2002.

18 Georg Gottlob, Zoltán Miklós, and Thomas Schwentick. Generalized hypertree decompositions: NP-hardness and tractable variants. *J. ACM*, 56(6), 2009.

19 Pavol Hell and Jaroslav Nesetril. The core of a graph. *Discrete Mathematics*, 109(1-3):117–126, 1992.

20 Yannis Ioannidis. Approximations in database systems. In *ICDT*, pages 16–30, 2003.

21 Phokion G. Kolaitis and Jonathan Panttaja. On the complexity of existential pebble games. In *CSL*, pages 314–329, 2003.

22 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.

23 Phokion G. Kolaitis and Moshe Y. Vardi. Conjunctive-query containment and constraint satisfaction. *J. Comput. Syst. Sci.*, 61(2):302–332, 2000.

24 Qing Liu. Approximate query processing. In *Encyclopedia of Database Systems*, pages 113–119, 2009.

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

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

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

7 Appendix

▶ **Theorem 19** (restated). *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.*

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 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.

▶ **Claim 31.** Let $q(\bar{x})$ be a CQ. Then $q \in \mathsf{GHW}(1)$ iff $\mathcal{G}_\exists(q)$ is an acyclic graph.

A CQ q is *connected* if $\mathcal{G}(q)$ is connected. Using Theorem 8, we have the following:

▶ **Lemma 32.** *Let q be a connected CQ. If q has a* $\mathsf{GHW}(1)$*-overapproximation, then it has one that is connected.*

We start by proving Theorem 19 for Boolean CQs; thus, until stated otherwise, we assume all CQs to be Boolean. First we show Theorem 19 for connected CQs, then we extend it to the non-connected case and finally prove the general non-Boolean statement.

7.1 The connected case

We start with the following technical lemma. 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.

▶ **Lemma 33.** *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. ◀

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^q_u and t^q_v. We call such an h a *swapping endomorphism* for u and v. Note that Lemma 33 tells us that if such a swapping homomorphism for u and v exists, then it is unique.

▶ **Lemma 34.** *Let q be a connected core in* 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 33.

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 $R(v, u)$. By construction, we have the following:

▶ **Claim 35.** 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 19 for connected CQs, we need some notation and properties of GHW(1)-overapproximations. 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 $\hat{q}'$. Checking whether $(\hat{q}, X) \to_1 (\hat{q}', Y)$ can still be done in polynomial time.

▶ **Lemma 36.** *Suppose q is a CQ and suppose q' is a connected core that is a* 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 $q \rightarrow_1 q'$, 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 homomorphism from q' to q. First, assume by contradiction that Duplicator's strategy witnessing $q \rightarrow_1 q'$ 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 34. Finally, suppose towards a contradiction that $h(z') = h(z'')$, where $z' \neq z''$ and $z' = u'$ and z'' is in $t_{v'}^{q'}$ (the other case is analogous). Again by composing h with the strategy witnessing $q \rightarrow_1 q'$ 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 36 and Lemma 34, 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 19 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 $q_i \rightarrow_1 q_i \setminus e$. 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 $q \rightarrow_1 q_u \# q_v$. 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 $(q_i, \{u_u, v_v\}) \rightarrow_1 (q_i \setminus e, \{u_u, v_v\})$. 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.

Notice 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 $q_i \rightarrow_1 q_{i+1}$ and $q_{i+1} \rightarrow_1 q_i$. In particular, $q \rightarrow_1 q^*$ and $q^* \rightarrow_1 q$, 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 $q_i \rightarrow_1 q_{i+1}$ and $q_{i+1} \rightarrow_1 q_i$, for each $0 \leq i < m$, and thus $q_u \# q_v \rightarrow_1 q^*$ and $q^* \rightarrow_1 q_u \# q_v$. Since $(u, v) \in \mathcal{P}$, it follows that $q \rightarrow_1 q_u \# q_v$, and then $q \rightarrow_1 q^*$. Using the fact that $q_u \# q_v \rightarrow q$, we have that $q^* \rightarrow_1 q$. 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 34, 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 36, 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 36, $u \neq v$ and then u and v are adjacent. Also, by Lemma 36, 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 35), 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 36 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. ◀

7.2 The unconnected case

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 $q \rightarrow_1 \bigwedge_{p\in\mathcal{Q}} p$ and $\bigwedge_{p\in\mathcal{Q}} p \rightarrow_1 q$, 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 $p \not\rightarrow_1 \hat{p}$, for each pair of distinct CQs $p, \hat{p} \in \mathcal{Q}$. Let p be a CQ in $\mathcal{Q}$. Since $p \rightarrow_1 q'$ and p is connected, it follows that $p \rightarrow_1 p^*$, where p^* is a connected component of q'. Also, since $q' \rightarrow \bigwedge_{p\in\mathcal{Q}} p$, there is $p_0 \in \mathcal{Q}$ such that $p^* \rightarrow p_0$. In particular, $p \rightarrow_1 p_0$. 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.

7.3 The non-Boolean case

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. Recall also that $q(\bar{x})$ is connected if $\mathcal{G}(q)$ is connected, and a connected component of q is a maximal connected subquery. If $q(\bar{x})$ is connected, $q'(\bar{x})$ 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 $q_1 \dots, q_m$ be the connected components of q. Let $\mathcal{C}^{\mathtt{free}}$ be the CQs in $\{q_1 \dots, q_m\}$ 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 $\{q_1, \dots, q_m\}$ 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 $q_i^B \rightarrow_1 p^B$.
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 $(p', \bar{z}) \rightarrow_1 (p'', \bar{z})$.
   b. Check whether each part $p'(\bar{z}) \in \mathcal{S}$ has a $\mathsf{GHW}(1)$-overapproximation $p'_*(\bar{z})$. 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 only 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 $w \notin \{u, v\}$.

Observe that Lemma 3–5, Claim 35 and Lemma 36 hold for the non-Boolean case, when we consider the adapted definition for $q_u\#q_v$ (exactly the same arguments apply). Using this, we have 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 19.