Bag Equivalence of Bounded Symmetry-Degree Conjunctive Queries with Inequalities

Mingmin Chen and Todd J. Green

Department of Computer Science
University of California, Davis
Davis, CA 95616 USA
{chenmi,green}@cs.ucdavis.edu

Abstract. We consider the problem of checking equivalence of conjunctive queries with inequalities under bag (multiset) semantics. The problem is known to be decidable in PSPACE and as hard as graph isomorphism, but its exact complexity remains open. We introduce a natural restriction based on what we call the symmetry degree of queries, and show that when the symmetry degree is bounded by a fixed polynomial in the size of the query, the equivalence problem is in Π_2^p. We also show that for asymmetric queries (those of symmetry degree 1), checking bag-equivalence is polynomial-time Turing equivalent to the graph *auto*morphism problem. These results can be interpreted as first steps in a more general program of finding "islands of tractability" for testing equivalence of conjunctive queries with inequalities under bag (rather than set) semantics.

1 Introduction

We study the problem of checking equivalence of conjunctive queries (CQs) with built-in inequality predicates over a densely-ordered domain under bag (multiset) semantics. This problem is well-understood assuming the classical set semantics, where it is known that the complexity jumps from NP-complete for ordinary CQs [5] to Π_2^p-complete when inequalities are allowed [21, 27]. For bag semantics (implemented by most DBMSs, and hence of practical as well as theoretical interest), the problem is comparatively less-well understood: it is known to be as hard as graph isomorphism [6] and decidable in PSPACE [25], but the exact complexity is unknown (and seems to be difficult to resolve).

In this paper, we introduce a natural restriction on the structure of CQs called *bounded symmetry degree* (a bound on the number of query automorphisms) and use this notion to study the complexity of the bag equivalence problem. The notion is adapted from the well-known concept of *graph symmetry* studied extensively in the graph theory literature [15, 19, 23, 16]. We show that if the symmetry degree of CQs with inequalities is bounded, then bag equivalence can be decided in Π_2^p (and we also show that checking for bounded symmetry degree can be done in Π_2^p). In fact, we show this holds even when we relax the restriction to allow queries of *polynomially bounded* symmetry degree (see Section 3.1). We

also consider several stronger forms of symmetry degree bounds, and establish that for asymmetric CQs, bag equivalence of CQs with inequalities reduces to checking graph automorphism, while a bound on the number of self-joins in the queries yields a coNP upper bound. We also compare the notion of bounded symmetry degree with the standard notion of *bounded hypertree width*, which turns out to be orthogonal and not obviously applicable to the setting of bag semantics. These results may be interpreted as first steps in a more general program of finding "islands of (relative) tractability" for testing equivalence of CQs with inequalities under bag (rather than set) semantics.

Outline. Section 2 contains preliminaries and a discussion of related work. Section 3 defines the notion of bounded symmetry degree, establishes some basic results, and compares it with bounded hypertree width. Section 4 presents the main results on equivalence, for bag and bag-set semantics. We conclude in Section 5. The Appendix contains formal proofs of our results.

2 Preliminaries and Related Work

Conjunctive queries with inequalities. Following the standard terminology [1], a *conjunctive query* (CQ) is a safe, nonrecursive datalog rule whose body is a conjunction of relational atoms. A *conjunctive query with inequalities* ($\text{CQ}^<$) is a conjunctive query extended with a conjunction of inequality atoms using $<, \leq$, and $\neq$. We use uppercase letters for predicate symbols, $x, y, z, \ldots$ for variables, $a, b, c, \ldots$ for constants, and $t, u, v, \ldots$ for variable or constant terms, with overbars $\bar{x}, \bar{a}, \bar{t}$, etc. indicating lists. We write a $\text{CQ}^<$ Q as $Q(\bar{x}) :- R, C$ where R, the *relational part*, is a conjunction of relational atoms and C, the *conditions*, is a conjunction of inequalities $x\ \theta\ t$, $\theta \in \{<, \leq, \neq\}$. We denote by Q^R the CQ obtained from Q by deleting C. For example, if Q is $Q(x, y) :- T(x, y), S(x, y), x > y, y \neq 2$, then Q^R is $Q^R(x, y) :- T(x, y), S(x, y)$.

Semantics of queries. We fix a densely-ordered domain of *constants* which, for concreteness, we will assume is that of the rationals $\mathbb{Q}$. A *set relation of arity k* is a finite set R of *tuples* from $\mathbb{Q}^k$. A *bag (multiset) relation of arity k* is a mapping $R : \mathbb{Q}^k \to \mathbb{N}$ of tuples to their associated *multiplicities*, where the mapping is non-zero on only finitely many tuples. A *set* (resp., *bag*) *database instance* I is an assignment of set (resp., bag) relations R^I to predicate names R.

The semantics of $\text{CQ}^<$s on set and bag relations is based on the notion of *valuations*. A valuation is mapping ν of variables to constants, extended to map constants to themselves. Valuations operate pointwise on lists/tuples and atoms in the expected way. Let Q be a CQ $Q(\bar{t}) :- R_1(\bar{t_1}), \ldots, R_n(\bar{t_n}), C$ and let I be a set instance. The *result of evaluating Q on I under set semantics* is the set relation $[\![Q]\!]^I$ defined

$$[\![Q]\!]^I = \{\nu(\bar{t}) \mid \nu \models C \text{ and } \nu(\bar{t_1}) \in R_1^I \wedge \cdots \wedge \nu(\bar{t_n}) \in R_n^I\}$$

Now suppose I is a bag instance. The *result of evaluating Q on I under bag semantics* is the bag relation $[\![Q]\!]_b^I$ defined

$$[\![Q]\!]_b^I(\bar{a}) = \sum_{\nu} (R_1^I(\nu(\bar{t_1})) \times \cdots \times R_n^I(\nu(\bar{t_n})))$$

where the sum is over all valuations ν such that $\nu(\bar{t}) = \bar{a}$ and $\nu \models C$. Finally, suppose I is a set instance. The *result of evaluating Q on I under bag-set semantics* is the bag relation $[\![Q]\!]_{bs}^I$ obtained by viewing I as a (duplicate-free) bag relation, and evaluating Q on I under bag semantics.

A CQ$^<$ Q is *satisfiable* (or *consistent* [21]) if there exists a set (or, equivalently, a bag) database instance I such that $[\![Q]\!]^I \neq \emptyset$. Every CQ is satisfiable, and a CQ$^<$ is satisfiable iff its comparison part is satisfiable. The latter can be checked in linear time using an algorithm in [4]. Here, we focus only on satisfiable queries, and we do not allow equality atoms $x = t$ (since, for satisfiable queries, such atoms can always be eliminated by replacing each occurrence of x with t).

Containment mappings and isomorphisms. Let $Q(\bar{x}) :- R, C$ and $Q'(\bar{x}') :- R', C'$ be two CQ$^<$s, and denote by $\mathcal{V}(Q)$ and $\mathcal{V}(Q')$ the sets of variables and constants that occur in Q and Q', respectively. A *containment mapping* [5] (or *homomorphism*) *from Q' to Q* is a mapping $h : \mathcal{V}(Q') \to \mathcal{V}(Q)$, lifted to operate on atoms and conjunctions of atoms in the obvious way, such that:

1. $h(\bar{t}') = \bar{t}$.
2. h is the identity mapping on constants.
3. $h(R') \subseteq R$.
4. $C \models h(C')$, i.e. $C \models h(a')\ \theta\ h(b')$ if $a'\ \theta\ b'$ is in C' where $\theta \in \{<, \leq, \neq\}$

A homomorphism h is called an *isomorphism* if h is bijective and its inverse is also a homomorphism. Note that Q, Q' are isomorphic via mapping h iff Q^R and Q'^R are isomorphic and C and C' are logically equivalent modulo h. Given a candidate isomorphism mapping, the latter condition can be checked by comparing the transitive closures of C and C' (see Appendix for details).

An *endomorphism* (resp., *automorphism*) is a homomorphism (resp., isomorphism) from a CQ$^<$ to itself. For a CQ$^<$ Q, the set of automorphisms of Q^R, denoted by $\mathsf{Aut}(Q^R)$, forms a subgroup of the symmetric group on $\mathcal{V}(Q)$. The identity element in this group is the identity mapping. Similarly, the set of endomorphisms of Q^R, denoted by $\mathsf{End}(Q^R)$, forms a monoid.

Linearizations and linear expansions. We use the notions of *linearizations* and *linear expansions* of CQ$^<$s due to Nutt et al. [25], which are based on the standard notion of linear extensions for partially ordered sets. Let C be a set of inequalities $(<, \leq, \neq)$ over a set T of variables and constants. A *linearization of C* is a total order L over T that is compatible with C. (Note that L may equate distinct variables, when this is allowed by C.)

If L is a linearization of C, and C is the condition of a CQ$^<$ Q, then we denote Q_L the CQ$^<$ obtained from Q by replacing C with L, and we say that

Q_L is a *linearization of* Q. By default, we do not eliminate equalities in linearizations unless explicitly pointed out. (Thus, the condition for a linearization is a conjunction of inequalities of the form $x\ \theta\ t$, $\theta \in \{<, =\}$.) The *linear expansion of* Q is the bag $\mathsf{lin}(Q)$ of all linearizations of Q (it can be viewed as a *union* of $\mathrm{CQ}^<$s). We say that linear expansions $\mathsf{lin}(Q)$ and $\mathsf{lin}(Q')$ are *isomorphic* if there is a bijection $f : \mathsf{lin}(Q) \to \mathsf{lin}(Q')$ such that Q_L is isomorphic to $f(Q_L)$ for every $Q_L \in \mathsf{lin}(Q)$.

Known results on equivalence. The equivalence problem for conjunctive queries with or without inequalities under set semantics, bag semantics, bag-set semantics have been studied extensively, beginning with the seminal paper by Chandra and Merlin [5] which established the NP-completeness of checking containment/equivalence of CQs under set semantics using a characterization in terms of containment mappings. Containment/equivalence of $\mathrm{CQ}^<$ queries was shown to be in Π_2^P by Klug [21] and Π_2^P-hard by van der Meyden [27].

The papers by Chaudhuri and Vardi [6] and Ioannidis and Ramakrishnan [18] initiated the study of query containment and equivalence under bag/bag-set semantics. The former paper showed that two CQs are bag-equivalent iff they are isomorphic, which yields a logspace equivalence of the problem with the graph isomorphism problem (GI). The latter paper showed that containment of *unions* of CQs is undecidable via a reduction from Hilbert's Tenth Problem. The decidability of bag containment of CQs remains an open problem. (A recent paper by Afrati et al. [2] gives positive decidability results for certain classes of CQs.) Bag containment of $\mathrm{CQ}^<$s, on the other hand, was shown to be undecidable by Jayram et al. [20].

Checking bag-set equivalence of $\mathrm{CQ}^<$s was shown to be decidable by Nutt et al. [25] (Cohen [8] extends it to bag semantics), a result we shall refer to again in the sequel:

Theorem 1 ([25, 8]). *Two $CQ^<$s Q, Q' are bag (bag-set) equivalent if and only if they have isomorphic linear expansions, which can be checked in* PSPACE.

This theorem also holds for bag-set equivalence for $\mathrm{CQ}^<$s. The PSPACE upper bound seems unlikely to be tight: in fact, using their characterization of equivalence in terms of linear expansions, it is not hard to show that the upper bound can be sharpened somewhat to $\mathrm{coNP}^{\#P}$. ($\mathrm{coNP}^{\#P}$ is contained in PSPACE, but it is not known whether the containment is strict.) However, the exact complexity of the problem remains open.

The following simple variation on an example given in [25] shows that non-isomorphic $\mathrm{CQ}^<$s may indeed have isomorphic linear expansions, hence isomorphism is not a necessary condition for bag equivalence of $\mathrm{CQ}^<$s:

Example 1. Let $\varphi(x, y, z)$ be the conjunction of relational atoms $R(x, y, z)$, $R(x, z, y)$, $R(y, x, z)$, $R(y, z, x)$, $R(z, x, y)$, $R(z, y, x)$. Now consider two Boolean $CQs^<$ Q, Q' defined

$$\begin{aligned} Q() &:- \varphi(x, y, z), x < z, y < z, x \neq y \\ Q'() &:- \varphi(x, y, z), x < y, x < z, y \neq z \end{aligned}$$

The linear expansions of Q and Q' are

$$\begin{aligned} Q_1() &:- \varphi(x,y,z), x < z, y < z, x > y \\ Q_2() &:- \varphi(x,y,z), x < z, y < z, x < y \end{aligned}$$

$$\begin{aligned} Q'_1() &:- \varphi(x,y,z), x < y, x < z, y > z \\ Q'_2() &:- \varphi(x,y,z), x < y, x < z, y < z \end{aligned}$$

Observe that Q_1 is isomorphic to Q'_1 and Q_2 is isomorphic to Q'_2. By Theorem 1, Q and Q' are bag-equivalent since they have isomorphic linear expansions. However, Q and Q' are not isomorphic.

3 Symmetry

3.1 Motivation and basic definitions

A $\text{CQ}^<$ is *linear* [9, 25] if its body contains no two occurrences of the same relational predicate symbol (in other words, self-joins are not allowed). For linear $\text{CQ}^<$, an easy corollary of Theorem 1 (pointed out in [25]) is that bag equivalence can be checked in polynomial time.

In this section, we introduce a generalization of this restriction based on what we term the *symmetry degree of a query*. This is a natural idea derived from the well-studied notion of the symmetry degree of a graph (see, e.g., [15, 19, 23, 16]), but it does seem to have been applied in database theory before. An *automorphism* of a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is a permutation σ of the vertex set $\mathcal{V}$, such that for any edge $e = (u, v)$, $\sigma(e) = (\sigma(u), \sigma(v))$ is also an edge. The *automorphism group* of $\mathcal{G}$ contains all these automorphisms. The order of the automorphism group of $\mathcal{G}$ is called its *symmetry degree*, denoted $\mathsf{sym}(\mathcal{G})$. For queries, the notion is exactly analogous: we define the *symmetry degree* of a $\text{CQ}^<$ Q, $\mathsf{sym}(Q)$, to be the order of its automorphism group $\mathsf{Aut}(Q^R)$.

Example 2. To illustrate, consider the Boolean CQs below:

$$\begin{aligned} Q_1() &:- P(x_1, x_2), P(x_2, x_3), \ldots, P(x_{n-1}, x_n) \\ Q_2() &:- S(u, v), S(u, w) \\ Q_3() &:- R(y_1), \ldots, R(y_n) \end{aligned}$$

Q_1 has symmetry degree 1 (the variables must all stay put), Q_2 has symmetry degree 2 (v and w can be swapped), and Q_3 has symmetry degree $n!$ (any permutation of the variables is an automorphism).

We will examine the complexity of computing $\mathsf{sym}(Q)$ in Section 3.2. In the meantime, we derive an easy bound based on counting self-joins.

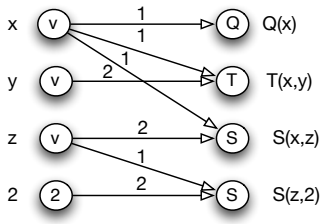
Proposition 1. *Suppose predicates $p_1, \ldots, p_k$ each occur $n_1, \ldots, n_k$ times, respectively, in the body of a CQ Q. Define the* number of self-joins of Q *to be* $n = n_1 + \cdots + n_k - k$. *Then*

$$\mathit{sym}(Q) \leq \prod_{i=1}^{k} (n_i!) \leq (n+1)!.$$

In particular, the proposition implies that linear queries have symmetry degree one. On the other hand, queries may have low symmetry degree but many self-joins, e.g., Q_1 in Example 2.

We introduce some additional terminology to use in the remainder of the paper. Q is called *symmetric* if $\mathsf{Aut}(Q^R)$ is a non-trivial subgroup of the symmetric group on $\mathcal{V}(Q)$, i.e. $\mathsf{sym}(Q) > 1$. Q is called *asymmetric* if $\mathsf{Aut}(Q^R)$ is trivial, which only contains the identity, i.e. $\mathsf{sym}(Q) = 1$. Q is called *rigid* if $\mathsf{End}(Q^R)$ is trivial.

Incidence graphs. A CQ has a natural interpretation as an edge-labeled directed hypergraph. However, to apply results of interest from graph theory, we may need to transform this hypergraph into an ordinary directed graph. Given a CQ Q, we construct its *incidence graph* [7] $\mathcal{G}(Q) = (\mathcal{V}, \mathcal{E}, \lambda)$, a labeled directed graph, as follows: $\mathcal{V}$ has a vertex labeled x for each variable x in Q, a vertex labeled c for each constant c in Q, and a vertex labeled P for each occurrence of predicate P in Q; and $\mathcal{E}$ contains an edge labeled n from vertex t to predicate vertex P whenever t occurs as the nth argument in P. For example, the incidence graph of CQ $Q(x) :- T(x, y), S(x, z), S(z, 2)$ is shown below:



We introduce labels in the definition of incidence graph to ensure that the construction preserves the symmetry degree[1] of Q:

Proposition 2. *For any CQ Q, we have $\mathsf{sym}(Q) = \mathsf{sym}(\mathcal{G}(Q))$.*

3.2 Complexity of Computing Symmetry Degrees

We next turn to the question of computing the symmetry degree of a CQ, starting with the simpler problem of deciding whether a query is symmetric. Not surprisingly, the latter problem reduces easily to the *graph automorphism* problem (GA), which is the problem of checking whether a given graph has a non-trivial automorphism.[2]

[1] The automorphism group of a labeled directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \lambda)$ is the subgroup of the automorphism group of $\mathcal{G}' = (\mathcal{V}, \mathcal{E})$ which respects the labeling λ.

[2] As with GI, GA is in NP but not known or believed to be either NP-complete or in PTIME. It is no harder than GI in the sense that the problem is polynomial-time many-one reducible to GI; however the converse relationship is at present unknown.

Proposition 3. *Checking whether a $CQ^<$ (CQ) is symmetric is polynomial-time many-one equivalent to the graph automorphism problem.*

This leads, using results in [22], to a characterization of the complexity of computing the symmetry degree:

Corollary 1. *Computing the symmetry degree of a $CQ^<$ (or CQ) is polynomial-time Turing equivalent to the graph isomorphism problem.*

The following is a direct corollary of a result in [3].

Corollary 2 (of Theorem 15 in [3]). *If the number of self-joins in a $CQ^<$ Q is bounded by some constant k, then there is an $n^{O(k)}$ time algorithm that computes $\mathsf{Aut}(Q^R)$ as a generating set in the symmetric group on $\mathcal{V}(Q)$.*

Finally, we consider testing for rigidity. Again our strategy is to reduce the problem on $\mathrm{CQ}^<$ queries to the analogous problem on graphs, where a result of Goralcik et al. [10] shows as a special case that graph rigidity is coNP-complete.

Proposition 4. *The problem of checking whether a $CQ^<$ (CQ) is rigid is coNP-complete.*

3.3 Symmetry Degree versus Hypertree Width

Gottlob et al. [11, 12] introduced the notion of hypertree width as a generalization of the classical notion of acyclic queries, and proved that Boolean conjunctive queries of bounded hypertree width can be evaluated efficiently (and that, as a consequence, query containment under set semantics is also tractable).

Symmetry degree and hypertree width seem to be essentially orthogonal concepts: intuitively, hypertree width is in some sense a measure of cyclicity, rather than symmetry. Moreover, a query with high symmetry degree may have small hypertree width, such as the following CQ that computes a "star self-join" on the first column of a relation R:

$$Q_1() \text{ :- } R(x, y_1), R(x, y_2), \ldots, R(x, y_k)$$

Q_1 has symmetry degree $k!$ but is acyclic hence has hypertree width 1. At the same time, a query with a high hypertree width may have a small symmetry degree, such as the CQ below:

$$Q_2() \text{ :- } \bigwedge_{i=1}^{k} \bigwedge_{j=1, j\neq i}^{k} R_{ik-k+j}(x_i, x_j)$$

(Q_2 intuitively corresponds to a complete directed graph without self-loops on k vertices where each edge is labeled with a distinct predicate name.) Q_2 has symmetry degree 1, but we can show that it has hypertree width $\lceil k/2 \rceil$.

We are not aware of any work studying the interaction of hypertree width and bag equivalence of queries. We also do not know of any work studying whether

bounded hypertree width leads to tractable query containment or equivalence checks for $\text{CQ}^<$s under the classical set semantics.

A recent paper by Pichler et al. [26] showed that the the query complexity of evaluating CQs under bag-set semantics becomes tractable when the CQs have bounded treewidth. A natural question (which we leave open here) is whether bounded symmetry degree also makes query evaluation under bag or bag-set semantics tractable.

4 Main Results

We are now ready to present our results on the complexity of bag equivalence of $\text{CQ}^<$s having bounded symmetry degree (Section 4.1). We also discuss how these bounds transfer to bag-set semantics (Section 4.2), and present an initial result that applies these ideas to the query equivalence problem under classical set semantics (Section 4.3).

4.1 Bag Equivalence

As already mentioned, Nutt et al. [25] showed that checking bag-equivalence of $\text{CQ}^<$s can be done in PSPACE, but the exact complexity of the problem remains unclear. Here, we show that if the symmetry degree of the queries is bounded, then the upper bound can be lowered to Π_2^p. In fact, we use a rather liberal notion of "bounded" here where we fix a univariate polynomial P ahead of time and say that a $\text{CQ}^<$ Q of length n has *polynomially bounded symmetry degree* if $\mathsf{sym}(Q) \leq P(n)$.

Theorem 2. *Checking bag equivalence of $CQ^<$s of polynomially bounded symmetry degree is in $coNP^{GI}$, hence in Π_2^p.*

The proof of the theorem relies on the following technical lemma:

Lemma 1. *Let Q be a $CQ^<$ of polynomially bounded symmetry degree, and let Q_L be one of its linearizations. Then the number of linearizations in $\mathsf{lin}(Q)$ that are isomorphic to Q_L can be computed in polynomial time with access to a GI oracle.*

Proof. The basic idea of the proof is as follows: first, we use the GI oracle to compute $\mathsf{Aut}(Q^R)$; second, we use $\mathsf{Aut}(Q^R)$ to compute the number of linearizations in $\mathsf{lin}(Q)$ that are isomorphic to Q_L.

To compute $\mathsf{Aut}(Q^R)$, we first compute the generating set of $\mathsf{Aut}(Q^R)$. This can be done in polynomial time using the GI oracle, since it is well-known that the problem of computing the generating set of the automorphism group of a labeled graph is polynomial-time Turing equivalent to GI [22, 17]. Now, since we have assumed $\mathsf{sym}(Q) \leq P(n)$, and we have the generating set for $\mathsf{Aut}(Q^R)$, it is clear that we can enumerate all the elements of $\mathsf{Aut}(Q^R)$ in polynomial time.

Next, let S denote the set of all linearizations in $\mathsf{lin}(Q)$ that are isomorphic to Q_L. Our goal is to compute $|S|$. We observe that a linearization $Q_{L'} \in S$ iff

$h(L) = L'$ for some $h \in \mathsf{Aut}(Q^R)$ and $L' \models C$, where C denotes the condition part of Q. In other words, $S = \{h(Q_L) \mid h \in \mathsf{Aut}(Q^R) \text{ and } h(L) \models C\}$. Since we have already computed $\mathsf{Aut}(Q^R)$, we can therefore use it to compute S in polynomial time, and return $|S|$ as the final answer. □

Now we are ready to prove Theorem 2.

Proof. By Lemma 1, given a GI oracle and a linearization Q_L of Q, we can compute in polynomial time the numbers of linearizations of Q and Q', respectively, which are isomorphic to Q_L. If these two numbers are different, then it follows by Theorem 1 that Q and Q' are not bag-equivalent. Thus we can guess a non-membership witness and verify it in polynomial time given a GI oracle, which means this problem is in coNP^{GI}. Since $\text{GI} \leq_T^p \text{NP}$, it follows that the problem is also in Π_2^p. □

We note that the pre-condition of Theorem 2 can certainly be checked in Π_2^p, by Corollary 1. Also, we note that it is not hard to show that Lemma 1 and Theorem 2 can be generalized to show a Π_2^p upper bound for checking bag equivalence of *unions* of $\text{CQ}^<$s ($\text{UCQ}^<$s). In Section 4.2, we will also generalize it to apply to bag-set equivalence.

Next we show some other (relatively) tractable subcases of checking bag equivalence of $\text{CQ}^<$s. Here, we show that if the number of self-joins is bounded by some constant, we can push the upper bound down to coNP.

Theorem 3. *If the numbers of self-joins in $CQ^<$ Q, Q' are bounded by some constant k, then the problem of checking whether Q and Q' are bag equivalent is in coNP.*

Proof. Here wlog we assume that Q and Q' have the same relational part, since their relational parts are isomorphic by Theorem 1. As stated in Proposition 1, the symmetry degree is bounded by $(k+1)!$ and we can enumerate the automorphism group in constant time. For checking bag equivalence of Q and Q', we can guess a linearization Q_L. Initialize two sets S_Q and $S_{Q'}$ to be empty. By applying the automorphism to Q_L one by one and see if the comparison part is compatible with the comparison parts of Q and Q' respectively and if yes add it to the corresponding S_Q or $S_{Q'}$. This can be done in polynomial time. If the orders of these two sets are different, then Q and Q' are not bag-equivalent, since they have different numbers of linearizations that are isomorphic to Q_L. In other words, we can guess a non-member witness and verify it in polynomial time. Hence, this problem is in coNP. □

By imposing various restrictions on the symmetry degree of queries, we can establish other upper bounds on the complexity of the problem. The following result is one example.

Theorem 4. *Asymmetric $CQ^<$s Q and Q' are bag equivalent iff there is an isomorphism h from Q'^R to Q^R such that $h(C') \models\!\!=\!\!| C$. Moreover, bag equivalence of asymmetric $CQ^<$ queries is polynomial-time Turing equivalent to the graph automorphism problem.*

Proof. By Theorem 1, Q, Q' have isomorphic linear expansions. In other words, there is a bijective mapping $\psi : \mathsf{lin}(Q) \mapsto \mathsf{lin}(Q')$ such that Q_L is isomorphic to $\psi(Q_L)$ for all $Q_L \in \mathsf{lin}(Q)$. Denote by φ_i the isomorphism mapping from Q_{L_i} to $\psi(Q_{L_i})$. Since Q, Q' are asymmetric, there is only one isomorphism mapping from Q^R to Q'^R (Suppose there are two φ_1, φ_2, then $\varphi_1 \circ \varphi_2^{-1}$ is a nontrivial automorphism of Q^R, which is a contradiction). So, φ_i are all the same for any $Q_{L_i} \in \mathsf{lin}(Q)$. The isomorphism of Q and Q' follows. The only problem is to find out this only isomorphism mapping of their relational parts and check if their comparisons are equivalent under this isomorphism. By [22], asymmetric graph isomorphism is polynomial-time Turing equivalent to the graph automorphism problem. Hence, finding this isomorphism mapping from Q'^R to Q^R is polynomial-time Turing equivalent to the graph automorphism problem, so is the bag equivalence problem. □

4.2 Bag-Set Equivalence

Next we examine bag-set equivalence of CQ$^<$s. Theorem 4 holds not just for bag semantics, but also for bag-set semantics. Let us now look at extending Theorem 2 as well.

We have the following lemma which is analogous to Lemma 1, but uses an NP (rather than GI) oracle. Intuitively, we use this extra power to deal with the fact that for bag-set semantics, when manipulating linearizations, we are forced to reduce equalities (since this can have the effect revealing that an atom in the body is actually a duplicate copy and should be removed).

Lemma 2. *Let Q be a CQ$^<$ of polynomially bounded symmetry degree, and let Q_L be one of its linearizations (with equalities reduced). The number of the linearizations in $\mathsf{lin}(Q)$ that are isomorphic to Q_L can be computed in polynomial time with access to a NP oracle.*

Using this lemma, we can establish the analog of Theorem 2 for bag-set semantics.

Theorem 5. *Checking bag-set equivalence of CQ$^<$s of polynomially bounded symmetry degree is in coNP$^{NP} = \Pi_2^p$.*

4.3 Set Equivalence

We have already seen in the preceding sections that bounded symmetry degree makes bag (bag-set) equivalence problem of CQ$^<$ easier to tackle. We show in this section a preliminary application of bounded symmetry to checking equivalence under the classical set semantics. As mentioned earlier, Klug [21] and Van der Meyden [27] showed that the *containment/equivalent* problem is Π_2^p-complete for CQ$^<$s, hence equivalence is in Π_2^p. Here we show that the complexity is lower than Π_2^p if the CQ$^<$s are rigid.

Theorem 6. *Checking set equivalence of rigid CQ$^<$s is polynomial-time Turing reducible to the graph automorphism problem.*

5 Conclusions and Future Work

We note that the equivalence results on bag semantics of $\text{CQ}^<$s presented here have wider applicability to any semantics in which $\text{CQ}^<$ equivalence is characterized by isomorphism of linearizations. This includes, for example, equivalence of $\text{CQ}^<$ on databases annotated with certain kinds of provenance information [13], as well as equivalence under the recently proposed $\mathbb{Z}$-semantics [14].

We conjecture that if only $<, \leq$ are allowed as inequalities, two $\text{CQ}^<$s Q, Q' are bag equivalent if and only if there is an isomorphism from Q'^R to Q^R under which the condition of Q' is equivalent to that of Q.

One possible future direction involves applying bounded symmetry degree to study query containment. Although the decidability of bag containment of CQs remains a longstanding open problem (Chaudhuri and Vardi [6] showed Π_2^P-hardness of this problem), bag containment of $\text{CQ}^<$s or of conjunctive queries is known to be undecidable [18, 20]. A natural question is whether these undecidability results hold even for queries of bounded symmetry degree. In fact, the reduction from Hilbert's Tenth Problem in [18] constructs a pair of UCQs each of which is composed of asymmetric CQs, hence bag containment remains undecidable even for unions of asymmetric CQs. The reduction of [20], on the other hand, uses $\text{CQ}^<$s of arbitrary symmetry degree, hence it remains open whether containment of $\text{CQ}^<$s of bounded symmetry degree is decidable.

Finally, we would like to pin down the exact complexity of the general case of bag equivalence of $\text{CQ}^<$ queries, which seems stubborn to resolve. The problem is reminiscent to another open issue in the area, that of identifying the exact complexity of checking bag equivalence of positive relational algebra queries with inequalities. This class of queries is expressively equivalence to $\text{UCQ}^<$, but exponentially more concise, and here again the known lower and upper bounds on the complexity are GI and PSPACE, respectively.

Acknowledgements. We thank the anonymous reviewers for their helpful comments. This work was supported by NSF CAREER award IIS-1055107.

References

1. S. Abiteboul, R. Hull, and V. Vianu. *Foundations of Databases.* Addison-Wesley, 1995.
2. F. N. Afrati, M. Damigos, and M. Gergatsoulis. Query containment under bag and bag-set semantics. *Inf. Process. Lett.*, 110:360–369, April 2010.
3. V. Arvind and J. Köbler. On Hypergraph and Graph Isomorphism with Bounded Color Classes. *STACS 2006*, pages 384–395, 2006.
4. N. Brisaboa, H. Hernández, J. Paramá, and M. Penabad. Containment of conjunctive queries with built-in predicates with variables and constants over any ordered domain. In *Advances in Databases and Information Systems.* Springer, 1998.
5. A. K. Chandra and P. M. Merlin. Optimal implementation of conjunctive queries in relational data bases. In *STOC*, 1977.

6. S. Chaudhuri and M. Y. Vardi. Optimization of real conjunctive queries. In *PODS '93*, pages 59–70, New York, NY, USA, 1993. ACM.
7. C. Chekuri and A. Rajaraman. Conjunctive query containment revisited. *Theor. Comput. Sci.*, 239:211–229, May 2000.
8. S. Cohen. Equivalence of queries that are sensitive to multiplicities. *The VLDB Journal*, 18(3):765–785, 2009.
9. M. P. Consens and A. O. Mendelzon. Graphlog: a visual formalism for real life recursion. In *PODS '90*, pages 404–416, New York, NY, USA, 1990. ACM.
10. P. Goralcik and V. Koubek. Verifying nonrigidity. *Information Processing Letters*, 22(2):91–95, 1986.
11. G. Gottlob, N. Leone, and F. Scarcello. Hypertree decompositions and tractable queries. In *PODS '99*, pages 21–32, New York, NY, USA, 1999. ACM.
12. G. Gottlob, N. Leone, and F. Scarcello. Hypertree Decompositions: A Survey. In *FOCS*, 2001.
13. T. J. Green. Containment of conjunctive queries on annotated relations. In *ICDT*, 2009.
14. T. J. Green, Z. G. Ives, and V. Tannen. Reconcilable differences. In *ICDT*, 2009.
15. Z. Hedrlín and A. Pultr. Symmetric relations (undirected graphs) with given semigroups. *Monatsh. Math.*, 69(4):318–322, 1965.
16. P. Hell and J. Nešetřil. *Graphs and homomorphisms*. Oxford University Press, USA, 2004.
17. C. Hoffmann. *Group-theoretic algorithms and graph isomorphism*. Lecture notes in computer science. Springer, 1982.
18. Y. E. Ioannidis and R. Ramakrishnan. Containment of conjunctive queries: beyond relations as sets. *ACM Trans. Database Syst.*, 20(3):288–324, 1995.
19. S. Janson, T. Łuczak, and A. Ruciński. *Random graphs*. Wiley-Interscience Series in Discrete Mathematics and Optimization, 2000.
20. T. S. Jayram, P. G. Kolaitis, and E. Vee. The containment problem for real conjunctive queries with inequalities. In *PODS*, 2006.
21. A. Klug. On conjunctive queries containing inequalities. *J. ACM*, 35(1):146–160, 1988.
22. J. Köbler, U. Schöning, and J. Torán. *The graph isomorphism problem: its structural complexity*. Springer, 1993.
23. J. Lauri and R. Scapellato. *Topics in graph automorphisms and reconstruction*. Cambridge Univ Pr, 2003.
24. G. McNulty and C. Shallon. Inherently nonfinitely based finite algebras. In R. Freese and O. Garcia, editors, *Universal Algebra and Lattice Theory*. 1983.
25. W. Nutt, Y. Sagiv, and S. Shurin. Deciding equivalences among aggregate queries. In *PODS '98*, pages 214–223, New York, NY, USA, 1998. ACM.
26. R. Pichler and S. Skritek. The complexity of evaluating tuple generating dependencies. In *ICDT*, 2011.
27. R. van der Meyden. The complexity of querying indefinite data about linearly ordered domains. In *PODS '92*, pages 331–345, New York, NY, USA, 1992. ACM.

A Transitive closure of inequality set

Consider a satisfiable set C of inequalities, and denote by $\mathcal{V}(C)$ the set of variables and constants occurring in C. The *transitive closure of* C, denoted by C^*, is the conjunction of all inequalities over $\mathcal{V}(C)$ entailed by C. For example, if

$C = \{x < y, y \leq 2\}$, its transitive closure $C^* = \{x < y, y \leq 2, x < 2, x \leq y, x \leq 2, x \neq y, y \neq 2, x \neq 2\}$. (Notice that $C \models x < 3$, but $x < 3$ is not contained in C^* since 3 does not occur in C.) The transitive closure of a set of inequalities can be computed in polynomial time.

Proposition 5. *For sets C_1 and C_2 of inequalities, the following are equivalent:*

1. $C_1 \models\!\models C_2$, *i.e.* $C_1 \models C_2$ *and* $C_2 \models C_1$.
2. C_1 *and* C_2 *have the same sets of linearizations.*
3. C_1 *and* C_2 *have the same transitive closures.*

B Proofs

Proof. (of Proposition 3) One direction is clear as the graph automorphism problem can be trivially reduced to the problem of checking symmetry of a Boolean CQ Q with a single binary relation E such that $E(x, y)$ and $E(y, x)$ are both subgoals in Q if vertices x and y are adjacent in graph G. In the other direction, given a CQ Q, we construct its incidence graph $\mathcal{G}(Q)$. By Proposition 2, Q is symmetric iff $\mathcal{G}(Q)$ is symmetric. $\mathcal{G}(Q)$ is a labeled graph, but it is easy to verify that labeled graph automorphism is many-one reducible to (ordinary) graph automorphism, which completes the proof. □

Proof. (of Corollary 1) The reductions used in Proposition 3 are parsimonious, so computing the symmetry degree of a $\text{CQ}^<$ is polynomial-time many-one equivalent to #GA[3]. By results in [22], #GA is polynomial-time Turing equivalent to the graph isomorphism problem. □

Proof. (of Corollary 2) Since the number of self-joins is bounded, the times of labels that are used in its associated hypergraph are also bounded, which corresponds exactly to bounded color classes. Theorem 15 in [3] states that "Given a hypergraph $X = (V, E)$ with color classes of size bounded by k, there is an $n^{O(k)}$ time algorithm that computes $\mathsf{Aut}(X)$ as a generating set in $Sym(V)$. In particular, HGI and HGA are in PTIME.", which completes the proof. □

Proof. (of Proposition 4) For $\text{CQ}^<$ (and hence CQ as well) it is easy to see this problem is in coNP because we can guess a nontrivial endomorphism candidate and check in polynomial time if it is an endomorphism. If so, the query is not rigid. Goralcik et al. [10] showed that the problem of whether a finite algebra A is nonrigid is NP-complete as soon as the type of A has either one binary or two unary symbols. This result can be applied in particular to the graph algebra associated with a directed graph [24] to show that checking rigidity of directed graphs is also coNP-complete. Since a directed graph can be transformed to a

[3] #GA is the problem of computing the order of graph automorphism group of a given graph. It is polynomial-time Turing equivalent to #GI and GI [22], where #GI is short for the counting the number of graph isomorphisms from one graph to the other.

Boolean CQ using a single binary predicate, such that the graph is rigid iff the associated query is rigid, it follows that checking rigidity of CQs (and hence also $\text{CQ}^<$s) is coNP-complete.

Proof. (of Lemma 2) We can find a homomorphism h from Q^R to Q_L^R by using this NP oracle. We can also enumerate all the automorphisms of Q by this NP oracle in polynomial time since it has polynomial symmetry degree. Using similar argument as in Lemma 1, by composing h by the elements in $\mathsf{Aut}(Q^R)$ one by one and applying to Q^R and then see if the comparisons are compatible to L under this new homomorphism, we can get the number of the linearizations in $\mathsf{lin}(Q)$ that are isomorphic to Q_L. □

Proof. (of Theorem 5) Use the same argument as in Theorem 2, but instead of a GI oracle, we use an NP oracle as Lemma 2 suggests. This yields a $\text{coNP}^{\text{NP}} = \Pi_2^p$ upper bound. □

Proof. (of Theorem 6) Assuming that two rigid queries Q and Q' are set equivalent, then there is only one homomorphism mapping ψ from Q'^R to Q^R and only one homomorphism mapping φ from Q^R to Q'^R, i.e. Q^R and Q'^R are homomorphically equivalent. But Q and Q' are rigid, hence cores, and if two cores are homomorphically equivalent then they are isomorphic. Since Q'^R and Q^R are cores, ψ is a unique isomorphism from Q'^R to Q^R. Now the only problem is to find out this unique isomorphism mapping, and check the conditions of these queries are compatible under this mapping which is in polynomial time. By [22], asymmetric graph isomorphism is polynomial-time Turing equivalent to the graph automorphism problem. Since rigid graphs are subclass of asymmetric graphs, the set equivalence problem is polynomial-time Turing reducible to the graph automorphism problem. □