A monotone preservation result for Boolean queries expressed as a containment of conjunctive queries

A Monotone Preservation Result for Boolean Queries Expressed as a Containment of Conjunctive Queries

Dimitri Surinx[a,*], Jan Van den Bussche[a]

[a] *Hasselt University, Martelarenlaan 42, 3500 Hasselt, Belgium*

Abstract

When a relational database is queried, the result is normally a relation. Some queries, however, only require a yes/no answer; such queries are often called *boolean queries*. It is customary in database theory to express boolean queries by testing nonemptiness of query expressions. Another interesting way for expressing boolean queries are containment statements of the form $Q_1 \subseteq Q_2$ where Q_1 and Q_2 are query expressions. Here, for any input instance I, the boolean query result is *true* if $Q_1(I)$ is a subset of $Q_2(I)$ and *false* otherwise.

In the present paper we will focus on nonemptiness and containment statements about conjunctive queries. The main goal is to investigate the monotone fragment of the containments of conjunctive queries. In particular, we show a preservation like result for this monotone fragment. That is, we show that, in expressive power, the monotone containments of conjunctive queries are exactly equal to conjunctive queries under nonemptiness.

Keywords: Databases, Query languages, Expressive power

1. Introduction

In this paper, we compare boolean queries (or integrity constraints) expressed using conjunctive queries (CQs [1]) in two different ways:

Nonemptiness: As an expression of the form $Q \neq \emptyset$, with Q a CQ;

Containment: As an expression of the form $Q_1 \subseteq Q_2$, with Q_1 and Q_2 two CQs.

An example of a nonemptiness query is "there exists a customer who bought a luxury product". An example of a containment query is "every customer who bought a luxury product also bought a sports product". A qualitative difference between nonemptiness and containment queries is that nonemptiness queries are always monotone: when the result is true on some input instance, it

*Corresponding author
Email address: dimitri.surinx@uhasselt.be (Dimitri Surinx)

is also true on any larger instance. In contrasts, containment queries need not be monotone, as shown by the example above. The nonemptiness of a CQ is always expressible as the containment of two CQs. For example, the nonemptiness of $(x) \leftarrow \mathit{Customer}(x), \mathit{Bought}(x, y), \mathit{Luxury}(y)$ is expressed as

$$() \leftarrow \mathit{true} \quad \subseteq \quad () \leftarrow \mathit{Customer}(x), \mathit{Bought}(x, y), \mathit{Luxury}(y).$$

Conversely, one may suspect that, as far as *monotone* queries are concerned, nothing more is expressible by a containment of two CQs. Indeed, we show in this paper that every monotone query expressed as the containment of two CQs is already expressible as the nonemptiness of a CQ. Such a result fits the profile of a preservation theorem since it gives a syntactical language for a semantical sublanguage. Preservation theorems have been studied intensively in model theory, finite model theory and database theory [2, 3, 4, 5, 6, 7].

From our proof it also follows that monotonicity testing of a containment of two CQs is decidable; specifically, the problem is NP-complete.

2. Preliminaries

A database schema Γ is a finite nonempty set of relation names. Every relation name R is assigned an arity, which is a natural number greater than zero. Let V be some fixed infinite universe of data elements and let R be a relation name of arity n. An *R-fact* is an expression of the form $R(a_1, \ldots, a_n)$ where $a_i \in V$ for $i = 1, \ldots, n$. Generally, a *fact* is an R-fact for some R. An *R-instance* I is a finite set of R-facts. More generally, an instance I of a database schema Γ is defined to be a nonempty union $\bigcup_{R \in \Gamma} I(R)$, where $I(R)$ is an R-instance[1]. The active domain of an instance I, denoted by $\mathrm{adom}(I)$, is the set of all data elements from V that occur in I. An instance I is called *connected* when for every two data elements $a, b \in \mathrm{adom}(I)$ there is a sequence of facts $f_1, \ldots, f_n$ in I such that: a is in $\mathrm{adom}(\{f_1\})$, b is in $\mathrm{adom}(\{f_n\})$, and $\mathrm{adom}(\{f_i\}) \cap \mathrm{adom}(\{f_{i+1}\}) \neq \emptyset$ for any $i = 1, \ldots, n-1$. An instance J is a called a *connected component* of I if J is connected, $J \subseteq I$ and J is maximal in I with respect to inclusion.

We have defined database and instances under the so called "logic programming perspective" [1]. We will define the results of conjunctive queries, however, under the so-called "named" perspective [1]. This will allow a lighter notation in our proof of Lemma 6 where we are taking subtuples of heads of conjunctive queries. In the *named perspective*, *tuples* are defined over a finite set of attributes, which we refer to as a *relation schema*. Formally, *tuples*, say $t = (u_i)_{i \in S}$ on a relation schema S, are considered as mappings, so t is a mapping on S and $t(i) = u_i$. Then, subtuples, say $t|_K$ for $K \subseteq S$ are treated as restrictions of the mapping H to K. On the empty relation schema, there is only one tuple, namely the empty mapping, also called the empty tuple. We denote the empty tuple by ().

[1] The reason for considering only nonempty instances will be explained in Remark 4

We formalize the notion of conjunctive queries as follows. From the outset we assume an infinite universe of variables. A *conjunctive query* is an expression of the form $Q : H \leftarrow B$ where the head H is a tuple of variables (tuple in the sense as just defined), and the body B is a set of atoms over Γ. An *atom* is an expression of the form $R(v_1, \ldots, v_n)$ where $R \in \Gamma$ and $v_1, \ldots, v_n$ are variables. We will denote the set of conjunctive queries over Γ as CQ_Γ. For a conjunctive query Q we will write H_Q for the head and B_Q for the body of Q. The *result schema* of a conjunctive query Q is the relation schema of the head H_Q. Note that we allow *unsafe* queries, i.e., queries with head variables that do not appear in the body. Semantically, for any instance I over Γ, $Q(I)$ is defined as:

$$\{f \circ H_Q \mid f \text{ is a homomorphism from } Q \text{ into } I\}.$$

Here, a homomorphism f from Q into I is a function on the variables in H_Q and B_Q to $\mathrm{adom}(I)$ such that $f(B_Q) \subseteq I$. When the variables in H_Q are all present in B_Q, we will also write that f is a homomorphism from B_Q into I. Interchangeably, we will write that B_Q maps into I.

Example 1. Consider the database schema with the relation name *Flights* of arity two. The following conjunctive query returns all the city pairs that are connected by flight with one stopover: $(A : x, B : y) \leftarrow \mathit{Flight}(x, z), \mathit{Flights}(z, y)$.

This query returns $\{(A : \mathit{Vienna}, B : \mathit{Brussels}), (A : \mathit{Paris}, B : \mathit{Rome})\}$ on the instance

$$\{\mathit{Flights}(\mathit{Paris}, \mathit{Brussels}), \mathit{Flights}(\mathit{Brussels}, \mathit{Rome}), \mathit{Flights}(\mathit{Vienna}, \mathit{Paris})\}.$$

Remark 2. It is convenient to assume that variables are data elements in V. Then, we can use a nonempty body of a conjunctive query as a database instance. As a consequence, an R-atom can then be thought of as an R-fact.

For any two CQs Q_1 and Q_2, we write $Q_1 \sqsubseteq Q_2$ if $Q_1(I) \subseteq Q_2(I)$ for any database instance I over Γ. We recall:

Theorem 3 ([8]). *Let Q_1 and Q_2 be conjunctive queries with the same result schema where $B_{Q_1} \neq \emptyset$. Then, $Q_1 \sqsubseteq Q_2$ iff $H_{Q_1} \in Q_2(B_{Q_1})$.*

Two CQs Q_1 and Q_2 are equivalent if $Q_1 \sqsubseteq Q_2$ and $Q_2 \sqsubseteq Q_1$. An atom in B_Q is called *redundant* if the query obtained from Q by removing that atom is equivalent to Q.

A *boolean query* over a database schema Γ is a mapping from instances of Γ to $\{\mathit{true}, \mathit{false}\}$. We can associate to any conjunctive query Q, a boolean query $Q \neq \emptyset$, that is *true* on I if $Q(I) \neq \emptyset$ and *false* if $Q(I) = \emptyset$. We will write $\mathrm{CQ}_\Gamma^{\neq\emptyset}$ for the family of boolean queries of the form $Q \neq \emptyset$ where Q is in CQ_Γ.

As argued in the introduction, this is not the only natural way to express boolean queries. Containment statements of the form $Q_1 \subseteq Q_2$, where Q_1 and Q_2 have the same result schema, provide a clean way to express interesting nonmonotone boolean queries. Formally, the boolean query $Q_1 \subseteq Q_2$ is *true* on I if $Q_1(I)$ is a subset of $Q_2(I)$, and *false* on I otherwise. It is understood that we can only take containment boolean queries of two conjunctive queries

Q_1 and Q_2 if they have the same result schema. We write $\mathrm{CQ}_\Gamma^\subseteq$ for the family of boolean queries expressible by containment statements $Q_1 \subseteq Q_2$ where Q_1 and Q_2 are in CQ_Γ with the same result schema.

Recall that every conjunctive query Q is *monotone*, in the sense that for any two instances I, J over Γ, such that $I \subseteq J$, we have $Q(I) \subseteq Q(J)$. Furthermore, we say that a boolean query Q is *monotone* if for any two instances I, J over Γ, such that $I \subseteq J$, we have $Q(I) = \mathit{true}$ implies $Q(J) = \mathit{true}$. We denote the set of monotone boolean queries with MON.

Remark 4. In this paper we have defined instances to be nonempty. Indeed, the notion of monotonicity of a boolean query, over all instances including the empty set $\emptyset$, is not very interesting. Specifically, consider a boolean query Q. Either $Q(\emptyset)$ is *true*, in which case Q can only be monotone, with the empty instance, if Q is *true* everywhere. On the other hand, if $Q(\emptyset)$ is *false*, then there is no difference in the definition we use.

We will frequently use the following property of conjunctive queries with connected bodies. If Q is a conjunctive query with a connected body, then $Q(I \cup J) = Q(I) \cup Q(J)$ for any domain-disjoint instances I and J. We will refer to this property as the *additivity* property. Furthermore, we say that a query Q is additive if it has the additivity property.

3. Main result

In this section we will prove the main theorem of the present paper. This preservation theorem can be summarized as follows:

Theorem 5. *For any database schema Γ, $\mathrm{CQ}_\Gamma^\subseteq \cap \mathrm{MON} = \mathrm{CQ}_\Gamma^{\neq\emptyset}$. Specifically, every monotone query $Q_1 \subseteq Q_2$, where Q_1 and Q_2 are CQs, is equivalent to a query of the form $(() \leftarrow B) \neq \emptyset$, where B is empty or B is the union of some of the connected components of B_{Q_2}.*

Note that $\mathrm{CQ}_\Gamma^{\neq\emptyset} \subseteq \mathrm{CQ}_\Gamma^\subseteq \cap \mathrm{MON}$ already follows from the fact that $Q \neq \emptyset$ is equivalent to $() \leftarrow \emptyset \subseteq () \leftarrow B_Q$. To prove the remaining inclusion we first establish a few technical results. First, we show that any monotone containment of conjunctive queries is equivalent to a containment of conjunctive queries with empty heads. For the remainder of this section, we write Z_a to be the instance where there is exactly one fact $R(a, a, \ldots, a)$ for every $R \in \Gamma$. Note that for every CQ Q, we have $Q(Z_a) = \{(a, a, \ldots, a)\}$.

Lemma 6. *Let Q_1 and Q_2 be conjunctive queries. If $Q_1 \subseteq Q_2$ is monotone, then it is equivalent to the conjunctive query $() \leftarrow B_{Q_1} \subseteq () \leftarrow B_{Q_2}$.*

Proof. Let S be the result schema of Q_1 and Q_2. Write B_{Q_2} as $B_1, \ldots, B_k, B$ where the B_j are the connected components of B_{Q_2} each of which contain at least one variable in H_{Q_2}, and B is the collection of the remaining connected components. Define $A_j = \{i \in S \mid H_{Q_2}(i) \in \mathrm{adom}(B_j)\}$ for $j = 1, \ldots, k$ and let A_0 contain the remaining attributes in S. Furthermore, define $A = \bigcup_{1 \leq j \leq k} A_j$.

We first show that there is a function f such that $f \circ H_{Q_2}|_{A_0} = H_{Q_1}|_{A_0}$. Let a be a fresh data element. Define $I = Z_a \cup B_{Q_1} \cup \bigcup_{i \in C} Z_{H_{Q_1}(i)}$ where $C = \{i \in S \mid H_{Q_1}(i) \notin \text{adom}\,(B_{Q_1})\}$. Since, $Q_1(Z_a) = Q_2(Z_a)$ and $Q_1 \subseteq Q_2$ is monotone, we have $Q_1(I) \subseteq Q_2(I)$. Therefore, $H_{Q_1} \in Q_2(I)$ since $H_{Q_1} \in Q_1(I)$. Hence, there is a homomorphism f from Q_2 into I such that $f \circ H_{Q_2} = H_{Q_1}$. In particular, $f \circ H_{Q_2}|_{A_0} = H_{Q_1}|_{A_0}$ as desired.

Next, we show for each $j = 1, \ldots, k$ that

$$(H_{Q_1}|_{A_j} \leftarrow B_{Q_1}) \sqsubseteq (H_{Q_2}|_{A_j} \leftarrow B_j). \qquad (\star)$$

Let I be an instance over Γ and let a be a fresh data element. Suppose $t \in (H_{Q_1}|_{A_j} \leftarrow B_{Q_1})(I)$. Since $(H_{Q_1}|_{A_j} \leftarrow B_{Q_1})$ and Q_1 have the same body, and $H_{Q_1}|_{A_j}$ is a subtuple of H_{Q_1}, we can extend t to t' such that $t' \in Q_1(I)$. Furthermore, since $Q_1 \subseteq Q_2$ is monotone and $Q_1(Z_a) = Q_2(Z_a)$, we have $Q_1(I \cup Z_a) \subseteq Q_2(I \cup Z_a)$. Thus, $t' \in Q_2(I \cup Z_a)$, whence we also have $t \in (H_{Q_2}|_{A_j} \leftarrow B_j)(I \cup Z_a)$. Since $H_{Q_2}|_{A_j} \leftarrow B_j$ is additive, $t \in (H_{Q_2}|_{A_j} \leftarrow B_j)(I) \cup (H_{Q_2}|_{A_j} \leftarrow B_j)(Z_a)$. This implies that $t \in (H_{Q_2}|_{A_j} \leftarrow B_j)(I)$ since t is a tuple of data elements in I.

We now show that $Q_1 \subseteq Q_2$ is equivalent to $Q_1' \subseteq Q_2'$ where $Q_1' = () \leftarrow B_{Q_1}$ and $Q_2' = () \leftarrow B_{Q_2}$, which proves our lemma. Clearly, $Q_1(I) \subseteq Q_2(I)$ implies that $Q_1'(I) \subseteq Q_2'(I)$. For the other direction, suppose that $Q_1'(I) \subseteq Q_2'(I)$ and let $t \in Q_1(I)$. Then, we have the following:

- There is a homomorphism f_1 from B_{Q_1} to I such that $f_1 \circ H_{Q_1} = t$.
- There is a homomorphism f_2 from B_{Q_2} to I since $\emptyset \neq Q_1'(I) \subseteq Q_2'(I)$.
- There is a function h such that $h \circ H_{Q_2}|_{A_0} = H_{Q_1}|_{A_0}$.
- For every $j = 1, \ldots, k$, $t|_{A_j} \in (H_{Q_2}|_{A_j} \leftarrow B_j)(I)$ by $(\star)$. Hence, there is a homomorphism h_j from B_j into I such that $h_j \circ H_{Q_2}|_{A_j} = t|_{A_j}$.

We now define a homomorphism f from Q_2 into I such that $f \circ H_{Q_2} = t$:

$$f : x \mapsto \begin{cases} f_2(x), & \text{if } x \in B; \\ h_j(x), & \text{if } x \in \text{adom}\,(B_j) \text{ with } j \in \{1, \ldots, k\}; \\ f_1 \circ h(x), & \text{otherwise.} \end{cases}$$

We first show that $f \circ H_{Q_2} = t$.

$$\begin{aligned} f \circ H_{Q_2} &= f \circ (H_{Q_2}|_{A_0} \cup \bigcup_{1 \leq j \leq k} H_{Q_2}|_{A_j}) \\ &= f \circ H_{Q_2}|_{A_0} \cup \bigcup_{1 \leq j \leq k} f \circ H_{Q_2}|_{A_j} \\ &= f_1 \circ h \circ H_{Q_2}|_{A_0} \cup \bigcup_{1 \leq j \leq k} h_j \circ H_{Q_2}|_{A_j} \\ &= f_1 \circ H_{Q_1}|_{A_0} \cup \bigcup_{1 \leq j \leq k} t|_{A_j} = t|_{A_0} \cup \bigcup_{1 \leq j \leq k} t|_{A_j} = t \end{aligned}$$

Finally, we show that $f(B_{Q_2}) \subseteq I$.

$$f(B_{Q_2}) = f(B \cup \bigcup_{1 \leq j \leq k} B_j) = f(B) \cup \bigcup_{1 \leq j \leq k} f(B_j) = f_2(B) \cup \bigcup_{1 \leq j \leq k} h_j(B_j) \subseteq I$$

□

To prove Theorem 5 we may thus limit ourselves to conjunctive queries with empty heads.

First, we have a look at containments of the form $Q_1 \subseteq Q_2$ where B_{Q_1} contains at least two non-redundant atoms. In what follows, when we write that a conjunctive query Q is *minimal*, we mean that B_Q does not contain redundant atoms.

Lemma 7. *Let Q_1 and Q_2 be* CQ*s where Q_1 is minimal and $H_{Q_1} = H_{Q_2} = ()$. If B_{Q_1} contains at least two atoms, then $Q_1 \subseteq Q_2$ is equivalent to* true *or is not monotone.*

Proof. If $Q_1 \subseteq Q_2$ is not equivalent to true, then $Q_1 \not\sqsubseteq Q_2$. Thus, $Q_2(B_{Q_1}) = \emptyset$ by Theorem 3, whence we have $Q_1(B_{Q_1}) \not\subseteq Q_2(B_{Q_1})$. Since $|B_{Q_1}| \geq 2$, there exists a nonempty $B \subsetneq B_{Q_1}$. We have $Q_1(B) = \emptyset$ for otherwise Q_1 would not be minimal.

Clearly, $Q_1(B) = \emptyset$ implies that $Q_1(B) \subseteq Q_2(B)$. Hence, $Q_1 \subseteq Q_2$ is not monotone. □

We are now ready to prove Theorem 5.

Proof of Theorem 5. Let $Q_1 \subseteq Q_2$ be in $\mathrm{CQ}_\Gamma^{\subseteq} \cap \mathrm{MON}$. We want to show that $Q_1 \subseteq Q_2$ is equivalent to $(() \leftarrow B) \neq \emptyset$ where B is empty or B consists of some of the connected components of B_{Q_2}.

By Lemma 6 we may assume that $H_{Q_1} = H_{Q_2} = ()$. We may furthermore assume that Q_1 is minimal. The constant true query is expressed by $() \leftarrow \emptyset \neq \emptyset$. If $B_{Q_1} = \emptyset$, then $Q_1 \subseteq Q_2$ is equivalent to $Q_2 \neq \emptyset$ which is in $\mathrm{CQ}_\Gamma^{\neq\emptyset}$. Hence we may assume that $Q_1 \not\sqsubseteq Q_2$. Thus, $Q_2(B_{Q_1}) = \emptyset$ by Theorem 3.

If B_{Q_1} contains at least two atoms, then $Q_1 \subseteq Q_2$ is equivalent to true by Lemma 7.

Finally, suppose that B_{Q_1} contains exactly one atom. First, let us consider $B_{Q_1} = \{R(x_1, \ldots, x_n)\}$ where there is a repetition among $x_1, \ldots, x_n$. Define $I_1 = \{R(y_1, \ldots, y_n)\}$ where $y_1, \ldots, y_n$ are all different and not equal to any of $x_1, \ldots, x_n$. Clearly, $Q_1(I_1) = \emptyset$. Since $Q_2(B_{Q_1}) = \emptyset$, there is a connected component C of B_{Q_2} that does not map in B_{Q_1}. Furthermore, C does not map into I_1 either, whence we also have $Q_2(I_1) = \emptyset$. Indeed, if C would map into I_1, then C would also map into B_{Q_1} since I_1 maps into B_{Q_1}. It follows that C does not map into $I_1 \cup B_{Q_1}$ either, since C is connected and $\mathrm{adom}(I_1)$ is disjoint from $\mathrm{adom}(B_{Q_1})$. Therefore, $Q_2(I_1 \cup B_{Q_1}) = \emptyset$. Hence, $Q_1(I_1 \cup B_{Q_1}) \not\subseteq Q_2(I_1 \cup B_{Q_1})$ since the head of Q_1 is in $Q_1(I_1 \cup B_{Q_1})$. This contradicts that $Q_1 \subseteq Q_2$ is monotone, since $Q_1(I_1) = \emptyset \subseteq Q_2(I_1)$.

So, the only body left to consider is $B_{Q_1} = \{R(x_1, \ldots, x_n)\}$ where $x_1, \ldots, x_n$ are all different and $R \in \Gamma$. Our proof now depends on the size of Γ.

1. Suppose that Γ only contains the relation name R. Then $Q_1(I) \neq \emptyset$ for any instance I over Γ since $B_{Q_1} = \{R(x_1, \ldots, x_n)\}$ where $x_1, \ldots, x_n$ are all different. Since Q_1 and Q_2 have empty heads, we may thus conclude that $Q_1 \subseteq Q_2$ is equivalent to $Q_2 \neq \emptyset$ in $\mathrm{CQ}_\Gamma^{\neq\emptyset}$.
2. Suppose that Γ only contains R and exactly one other relation name T. Define $I_1 = \{T(y_1, \ldots, y_m)\}$ where $y_1, \ldots, y_m$ are different from each other and from $x_1, \ldots, x_n$. Since the body of Q_1 is an R-atom and I_1 only contains a T-atom, we have $Q_1(I_1) = \emptyset$. Hence, $Q_1(I_1) \subseteq Q_2(I_1)$. By the monotonicity of $Q_1 \subseteq Q_2$, we also have $Q_1(I_1 \cup B_{Q_1}) \subseteq Q_2(I_1 \cup B_{Q_1})$. Therefore, every connected component of B_{Q_2} maps in I_1 or B_{Q_1}. Indeed, $Q_2(I_1 \cup B_{Q_1}) \neq \emptyset$ since the head of Q_1 is in $Q_1(I_1 \cup B_{Q_1})$. This observation partitions the connected components of B_{Q_2} into two sets B' and B'', where B' contains the components that map into I_1, and B'' contains the components that map into B_{Q_1}.
   We now show that $Q_1 \subseteq Q_2$ is equivalent to $Q' = () \leftarrow B' \neq \emptyset$. To this end, suppose that $Q'(I) \neq \emptyset$ and $Q_1(I) \neq \emptyset$ for some instance I over Γ. Thus B' and B_{Q_1} map into I. Since B'' maps into B_{Q_1} by construction, we also have that B'' maps into I. Hence, $Q_2(I) \neq \emptyset$ as desired. For the other direction, suppose that $Q_1(I) \subseteq Q_2(I)$ for some instance I over Γ. If $Q_1(I) \neq \emptyset$, then $Q_2(I) \neq \emptyset$ by assumption. Clearly, $Q'(I) \neq \emptyset$ since $B_{Q'}$ is a subset of B_{Q_2}. On the other hand, if $Q_1(I) = \emptyset$, then I has no R-facts. Since instances cannot be empty, it must contain at least one T-fact, so I_1 maps into I. Thus B' also maps into I, whence $Q'(I) \neq \emptyset$ as desired.
3. Finally, suppose that Γ contains at least three relation names. Since $Q_2(B_{Q_1}) = \emptyset$, there is a connected component C of B_{Q_2} that does not map into B_{Q_1}. In particular, we know that C is not empty, whence it contains at least one atom, say a T-atom. (Note that T might be equal R.) Since there are three relation names in Γ there is at least one other relation name S in Γ that is not equal to T or R. Define $I_2 = \{S(z_1, \ldots, z_l)\}$ where $z_1, \ldots, z_l$ are all different from each other and from $x_1, \ldots, x_n$. By construction, C do not map into I_2 either, since C contains an atom different from S. Thus, $Q_2(I_2 \cup B_{Q_1}) = \emptyset$, whence we have $Q_1(I_2 \cup B_{Q_1}) \not\subseteq Q_2(I_2 \cup B_{Q_1})$ since $Q_1(I_2 \cup B_{Q_1}) \neq \emptyset$. However, $Q_1(I_2) = \emptyset$ since R and S are different, which implies that $Q_1(I_2) \subseteq Q_2(I_2)$. This contradicts the assumption that $Q_1 \subseteq Q_2$ is monotone. □

The proof of Theorem 5 gives us a procedure for deciding monotonicity for containments of CQs.

Corollary 8. *Deciding whether a containment in* $\mathrm{CQ}_\Gamma^\subseteq$ *is monotone is NP-complete.*

Proof. Let $Q_1 \subseteq Q_2$ be in $\mathrm{CQ}_\Gamma^\subseteq$. By Lemma 6 we may remove the head variables of Q_1 and Q_2. The NP-hardness of our problem is taken care of by Lemma 7. Indeed, when B_{Q_1} contains at least two non-redundant body atoms, the problem is equivalent to deciding $Q_1 \sqsubseteq Q_2$, which is known to be NP-hard [8].

Let us now show that the problem is in NP. By the proof of Theorem 5 we have the following cases when Q_1 is minimal:

- If $B_{Q_1} = \emptyset$, then $Q_1 \subseteq Q_2$ is always monotone.
- If $|B_{Q_1}| \geq 2$, then $Q_1 \subseteq Q_2$ is monotone if and only if $Q_1 \sqsubseteq Q_2$ (Lemma 7).
- If $Q_1 = \{R(x_1, \ldots, x_n)\}$ where there is a repetition among $x_1, \ldots, x_n$, then $Q_1 \subseteq Q_2$ is monotone if and only if $Q_1 \sqsubseteq Q_2$.
- If $B_{Q_1} = \{R(x_1, \ldots, x_n)\}$ where $x_1, \ldots, x_n$ are all different, then:
  - (a) If $|\Gamma| = 1$, then $Q_1 \subseteq Q_2$ is always monotone;
  - (b) If $|\Gamma| = 2$, then $Q_1 \subseteq Q_2$ is always monotone;
  - (c) If $|\Gamma| \geq 3$, then $Q_1 \subseteq Q_2$ is monotone if and only if $Q_1 \sqsubseteq Q_2$.

These properties suggest the following algorithm:

1. Check if $B_{Q_1} = \emptyset$; if so, accept;
2. Check if $Q_1 \sqsubseteq Q_2$; if so, accept;
3. Non-deterministically pick an atom $R(x_1, \ldots, x_n)$ in B_{Q_1};
4. Check the following:
   - $() \leftarrow R(x_1, \ldots, x_n) \sqsubseteq Q_1$;
   - $x_1, \ldots, x_n$ are all different.
5. Accept if $|\Gamma| \leq 2$ and the two checks above succeed; otherwise reject.

The containment checks ($\sqsubseteq$) are well known to be in NP [8], so this algorithm is an NP algorithm.

If the algorithm accepts in step 1, then $Q_1 \subseteq Q_2$ is equivalent to $Q_2 \neq \emptyset$, which is monotone. If the algorithm accepts in step 2, then the query $Q_1 \subseteq Q_2$ is the constant true query, whence is trivially monotone. If the algorithm accepts in step 5, then the query $() \leftarrow R(x_1, \ldots, x_n)$ is equivalent to Q_1, which is clearly minimal. Hence, by cases (a) and (b) in the above properties, $Q_1 \subseteq Q_2$ is monotone.

Conversely, suppose that $Q_1 \subseteq Q_2$ is monotone. If $B_{Q_1} = \emptyset$ or $Q_1 \sqsubseteq Q_2$, then the algorithm accepts in step 1 or 2 respectively. Otherwise, consider a CQ Q_1' obtained from Q_1 by omitting all redundant atoms. Certainly, Q_1' is minimal. Since $Q_1' \subseteq Q_2$ is monotone and $Q_1' \not\sqsubseteq Q_2$, the above properties imply that $B_{Q_1'}$ consists of a single atom $R(x_1, \ldots, x_n)$ where $x_1, \ldots, x_n$ are all different, and moreover that $|\Gamma| \leq 2$. Hence, by picking this atom in step 3, the algorithm will accept. □

4. Future Work

There are several directions for future work. In this paper, conjunctive queries are not allowed to have constants in the head and/or body. Our proof method does not work in the presence of constants. Whether our characterization still holds in this case is still open.

Now that we have a syntactical characterization for monotone $\mathrm{CQ}^{\subseteq}$ we can look at other query languages. The first languages that come to mind are conjunctive queries with nonequalities, or negation, or unions. Another interesting language to consider is the more expressive first-order logic. When we allow infinite instances, the monotone first-order boolean queries are characterized by the positive first-order sentences with nonequalities [7]. Whether this characterization still holds in restriction to finite instances remains open.

Another interesting line of work is to consider preservation theorems for other semantical properties, e.g., additivity. It can readily be verified that the additive queries in $\mathrm{CQ}^{\neq\emptyset}$ are exactly those with connected bodies. Another example of a preservation theorem for additivity is: connected $\mathrm{Datalog}^{\neg}$ captures the additive $\mathrm{Datalog}^{\neg}$ queries under stratified semantics [9].

References

[1] S. Abiteboul, R. Hull, V. Vianu, Foundations of Databases, Addison-Wesley, 1995.

[2] Y. Gurevich, Toward logic tailored for computational complexity, in: M. Richter, et al. (Eds.), Computation and Proof Theory, Vol. 1104 of Lecture Notes in Mathematics, Springer-Verlag, 1984, pp. 175–216.

[3] M. Ajtai, Y. Gurevich, Monotone versus positive, Journal of the ACM 34 (4) (1987) 1004–1015.

[4] C. Chang, H. Keisler, Model Theory, 3rd Edition, North-Holland, 1990.

[5] A. P. Stolboushkin, Finitely monotone properties, in: Proceedings of the 10th Annual IEEE Symposium on Logic in Computer Science, LICS '95, IEEE Computer Society, Washington, DC, USA, 1995, pp. 324–330.

[6] B. Rossman, Homomorphism preservation theorems, Journal of the ACM 55 (3) (2008) 15:1–15:53.

[7] M. Benedikt, J. Leblay, B. ten Cate, E. Tsamoura, Generating Plans from Proofs: The Interpolation-based Approach to Query Reformulation, Morgan&Claypool, 2016.

[8] A. Chandra, P. Merlin, Optimal implementation of conjunctive queries in relational data bases, in: Proceedings 9th ACM Symposium on the Theory of Computing, ACM, 1977, pp. 77–90.

[9] T. J. Ameloot, B. Ketsman, F. Neven, D. Zinn, Datalog queries distributing over components, ACM Transactions on Computational Logic 18 (1) (2017) 5:1–5:35.