The Data Complexity of Consistent Query Answering for Self-Join-Free Conjunctive Queries Under Primary Key Constraints

Paraschos Koutris
University of Washington, Seattle, USA
pkoutris@cs.washington.edu

Jef Wijsen
Université de Mons, Mons, Belgium
jef.wijsen@umons.ac.be

ABSTRACT

A relational database is said to be uncertain if primary key constraints can possibly be violated. A repair (or possible world) of an uncertain database is obtained by selecting a maximal number of tuples without ever selecting two distinct tuples with the same primary key value. For any Boolean query q, $\mathsf{CERTAINTY}(q)$ is the problem that takes an uncertain database $\mathbf{db}$ as input, and asks whether q is true in every repair of $\mathbf{db}$. The complexity of this problem has been particularly studied for q ranging over the class of self-join-free Boolean conjunctive queries. A research challenge is to determine, given q, whether $\mathsf{CERTAINTY}(q)$ belongs to complexity classes $\mathbf{FO}$, $\mathbf{P}$, or $\mathbf{coNP}$-complete. In this paper, we combine existing techniques for studying the above complexity classification task. We show that for any self-join-free Boolean conjunctive query q, it can be decided whether or not $\mathsf{CERTAINTY}(q)$ is in $\mathbf{FO}$. Further, for any self-join-free Boolean conjunctive query q, $\mathsf{CERTAINTY}(q)$ is either in $\mathbf{P}$ or $\mathbf{coNP}$-complete, and the complexity dichotomy is effective. This settles a research question that has been open for ten years.

Categories and Subject Descriptors

H.2.3 [**Database Management**]: Languages—*query languages*; H.2.4 [**Database Management**]: Systems—*relational databases*

Keywords

Conjunctive queries; consistent query answering; primary keys

1. INTRODUCTION

Primary key violations provide an elementary means for capturing uncertainty in the relational data model. A *block* is a maximal set of tuples of the same relation that agree on the primary key of the relation. Tuples in the same block are mutually exclusive: exactly one tuple is true, but we are uncertain about which one. We will refer to databases as "uncertain databases" to stress that they can violate primary key constraints.

A *repair* (or possible world) of an uncertain database is obtained by selecting exactly one tuple from each block. In general, the number of repairs of an uncertain database can be exponential in its size. For instance, if an uncertain database contains n blocks with two tuples each, then it contains $2n$ tuples and has 2^n repairs.

There are two natural semantics for answering Boolean queries q on an uncertain database. Under the *possibility semantics*, the question is whether the query evaluates to true on some repair. Under the *certainty semantics*, which is adopted in this paper, the question is whether the query evaluates to true on every repair. The certainty semantics adheres to the paradigm of *consistent query answering* [1, 4], which introduces the notion of database repairs with respect to general integrity constraints. In this work, repairing is exclusively with respect to primary key constraints, one per relation.

For any Boolean query q, the decision problem $\mathsf{CERTAINTY}(q)$ is the following.

PROBLEM:	$\mathsf{CERTAINTY}(q)$
INPUT:	uncertain database $\mathbf{db}$
QUESTION:	Does every repair of $\mathbf{db}$ satisfy q?

Three comments are in place. First, the Boolean query q is not part of the input. Every Boolean query q gives thus rise to a new problem. Since the input to $\mathsf{CERTAINTY}(q)$ is an uncertain database, we consider the *data complexity* of the problem. Second, we will assume that every relation name in q or $\mathbf{db}$ has a fixed known arity and primary key. The primary key constraints are thus implicitly present in all problems. Third, all the complexity results obtained in this paper can be carried over to non-Boolean queries; the restriction to Boolean queries eases the technical treatment, but is not fundamental.

The complexity of $\mathsf{CERTAINTY}(q)$ has gained considerable research attention in recent years, especially for q ranging over the set of self-join-free conjunctive queries. A challenging question is to distinguish queries q for which the problem $\mathsf{CERTAINTY}(q)$ is tractable from queries for which the problem is intractable. Further, if $\mathsf{CERTAINTY}(q)$ is tractable, one may ask whether it is first-order expressible. We will refer to these questions as the *complexity classification task of* $\mathsf{CERTAINTY}(q)$.

In the past decade, a variety of tools and techniques have been used in the complexity classification task of $\mathsf{CERTAINTY}(q)$ for self-join-free conjunctive queries q. In their pioneering work, Fuxman and Miller [7] introduced the notion of *join graph* (not to be confused with the classical notion of join tree). Later on, Wijsen [16] introduced the notion of *attack graph*. Kolaitis and Pema [8] applied Minty's algorithm [14] to the task. Koutris and Suciu [9] introduced the notion of *query graph* and the distinction between consistent and possibly inconsistent relations. All these techniques have limited applicability: join graphs seem too rudimentary to obtain general complexity dichotomies; attack graphs enable to characterize first-order expressibility of $\mathsf{CERTAINTY}(q)$, but only for

PODS'15, May 31–June 4, 2015, Melbourne, Victoria, Australia.
Copyright is held by the owner/author(s). Publication rights licensed to ACM.
ACM 978-1-4503-2757-2/15/05 ...$15.00.
http://dx.doi.org/10.1145/2745754.2745769 .

acyclic (in the sense of [3]) queries q; Minty's algorithm has been used to establish a $\mathbf{P}$-$\mathbf{coNP}$-complete dichotomy in the complexity of $\mathsf{CERTAINTY}(q)$, but only for queries q with exactly two atoms; the framework of Koutris and Suciu has also resulted in a $\mathbf{P}$-$\mathbf{coNP}$-complete dichotomy, but only when all primary keys consist of a single attribute. On top of the limited applicability of each individual technique, there is the difficulty that complexity classifications expressed in terms of different techniques cannot be easily compared.

In this paper, we make significant progress in the complexity classification task of $\mathsf{CERTAINTY}(q)$ for q ranging over the set of self-join-free conjunctive queries, by establishing the following results:

- Given a self-join-free Boolean conjunctive query q, it is decidable whether $\mathsf{CERTAINTY}(q)$ is in $\mathbf{FO}$. In [16], this was only shown under the assumption that queries are acyclic (in the sense of [3]).
- Given a self-join-free Boolean conjunctive query q, if $\mathsf{CERTAINTY}(q)$ is not in $\mathbf{FO}$, then it is $\mathbf{L}$-hard. In previous works [16, 18], Hanf locality was used to show first-order inexpressibility, resulting in involved proofs. The current paper takes a complexity-theoretic approach to first-order inexpressibility, which results in an easier proof of a stronger result.
- For every self-join-free Boolean conjunctive query q, $\mathsf{CERTAINTY}(q)$ is either in $\mathbf{P}$ or $\mathbf{coNP}$-complete, and the dichotomy is effective. In [9], this was only shown under the assumption that all primary keys are simple (i.e., consist of a single attribute).

Furthermore, given a self-join-free Boolean conjunctive query q, it can be decided in polynomial time in the size of q whether $\mathsf{CERTAINTY}(q)$ is in $\mathbf{FO}$, in $\mathbf{P} \setminus \mathbf{FO}$, or $\mathbf{coNP}$-complete. Our results solve a problem that has been open since 2005 [7].

Organization. This paper is organized as follows. Section 2 discusses related work. Section 3 introduces our data and query model. Section 4 defines attack graphs for Boolean conjunctive queries, extending an older notion of attack graph [18] that was defined exclusively for acyclic Boolean conjunctive queries. The section also states the main result of the paper, Theorem 1. Section 5 establishes an effective procedure that takes in a self-join-free Boolean conjunctive query q, and decides whether $\mathsf{CERTAINTY}(q)$ is in $\mathbf{FO}$. Section 6 shows that for every self-join-free Boolean conjunctive query q, $\mathsf{CERTAINTY}(q)$ is either in $\mathbf{P}$ or $\mathbf{coNP}$-complete. Section 7 concludes the paper.

2. RELATED WORK

Consistent query answering (CQA) goes back to the seminal work by Arenas, Bertossi, and Chomicki [1]. Fuxman and Miller [7] were the first ones to focus on CQA under the restrictions that consistency is only with respect to primary keys and that queries are self-join-free conjunctive. The term $\mathsf{CERTAINTY}(q)$ was coined in [16]. A recent and comprehensive survey on $\mathsf{CERTAINTY}(q)$ is [20].

Little is known about $\mathsf{CERTAINTY}(q)$ beyond self-join-free conjunctive queries. An interesting recent result by Fontaine [6] goes as follows. Let UCQ be the class of Boolean first-order queries that can be expressed as disjunctions of Boolean conjunctive queries (possibly with constants and self-joins). A daring conjecture is that for every query q in UCQ, $\mathsf{CERTAINTY}(q)$ is either in $\mathbf{P}$ or $\mathbf{coNP}$-complete. Fontaine showed that this conjecture implies Bulatov's dichotomy theorem for conservative CSP [5], the proof of which is highly involved (the full paper contains 66 pages).

The counting variant of $\mathsf{CERTAINTY}(q)$, denoted $\sharp\mathsf{CERTAINTY}(q)$, asks to determine the exact number of repairs that satisfy some Boolean query q. In [12], it was shown that for every self-join-free Boolean conjunctive query q, the counting problem $\sharp\mathsf{CERTAINTY}(q)$ is either in $\mathbf{FP}$ or $\sharp\mathbf{P}$-complete. For conjunctive queries q with self-joins, the complexity of $\sharp\mathsf{CERTAINTY}(q)$ has been established under the restriction that all primary keys consist of a single attribute [13].

3. PRELIMINARIES

We assume disjoint sets of *variables* and *constants*. If $\vec{x}$ is a sequence containing variables and constants, then $\mathsf{vars}(\vec{x})$ denotes the set of variables that occur in $\vec{x}$. A *valuation* over a set U of variables is a total mapping θ from U to the set of constants. At several places, it is implicitly understood that such a valuation θ is extended to be the identity on constants and on variables not in U. If $V \subseteq U$, then $\theta[V]$ denotes the restriction of θ to V.

If θ is a valuation over a set U of variables, x is a variable, and a is a constant, then $\theta_{[x \mapsto a]}$ is the valuation over $U \cup \{x\}$ such that $\theta_{[x \mapsto a]}(x) = a$ and for every variable y such that $y \neq x$, $\theta_{[x \mapsto a]}(y) = \theta(y)$. Notice that $x \in U$ is allowed.

Atoms and key-equal facts. Each *relation name* R of arity n, $n \geq 1$, has a unique *primary key* which is a set $\{1, 2, \ldots, k\}$ where $1 \leq k \leq n$. We say that R has *signature* $[n, k]$ if R has arity n and primary key $\{1, 2, \ldots, k\}$. We say that R is *simple-key* if $k = 1$. Elements of the primary key are called *primary-key positions*, while $k + 1, k + 2, \ldots, n$ are *non-primary-key positions*. For all positive integers n, k such that $1 \leq k \leq n$, we assume denumerably many relation names with signature $[n, k]$.

If R is a relation name with signature $[n, k]$, then $R(s_1, \ldots, s_n)$ is called an *R-atom* (or simply atom), where each s_i is either a constant or a variable ($1 \leq i \leq n$). Such an atom is commonly written as $R(\underline{\vec{x}}, \vec{y})$ where the primary key value $\vec{x} = s_1, \ldots, s_k$ is underlined and $\vec{y} = s_{k+1}, \ldots, s_n$. An *$R$-fact* (or simply fact) is an R-atom in which no variable occurs. Two facts $R_1(\underline{\vec{a}_1}, \vec{b}_1)$, $R_2(\underline{\vec{a}_2}, \vec{b}_2)$ are *key-equal* if $R_1 = R_2$ and $\vec{a}_1 = \vec{a}_2$. An R-atom or an R-fact is *simple-key* if R is simple-key.

We will use letters F, G, H for atoms. For an atom $F = R(\underline{\vec{x}}, \vec{y})$, we denote by $\mathsf{key}(F)$ the set of variables that occur in $\vec{x}$, and by $\mathsf{vars}(F)$ the set of variables that occur in F, that is, $\mathsf{key}(F) = \mathsf{vars}(\vec{x})$ and $\mathsf{vars}(F) = \mathsf{vars}(\vec{x}) \cup \mathsf{vars}(\vec{y})$.

Uncertain databases, blocks, and repairs. A *database schema* is a finite set of relation names. All constructs that follow are defined relative to a fixed database schema.

An *uncertain database* is a finite set $\mathbf{db}$ of facts using only the relation names of the schema. We refer to databases as "uncertain databases" to stress that such databases can violate primary key constraints.

We write $\mathsf{adom}(\mathbf{db})$ for the active domain of $\mathbf{db}$ (i.e., the set of constants that occur in $\mathbf{db}$). A *block* of $\mathbf{db}$ is a maximal set of key-equal facts of $\mathbf{db}$. The term R-block refers to a block of R-facts, i.e., facts with relation name R. If A is a fact of $\mathbf{db}$, then $\mathsf{block}(A, \mathbf{db})$ denotes the block of $\mathbf{db}$ that contains A. An uncertain database $\mathbf{db}$ is *consistent* if no two distinct facts are key-equal (i.e., if every block of $\mathbf{db}$ is a singleton). A *repair* of $\mathbf{db}$ is a maximal (with respect to set containment) consistent subset of $\mathbf{db}$. We write $\mathsf{rset}(\mathbf{db})$ for the set of repairs of $\mathbf{db}$.

Boolean conjunctive queries. A *Boolean query* is a mapping q that associates a Boolean (true or false) to each uncertain database, such that q is closed under isomorphism [11]. We write $\mathbf{db} \models q$ to denote that q associates true to $\mathbf{db}$, in which case $\mathbf{db}$ is said to *satisfy* q. A *Boolean first-order query* is a Boolean query that can be defined in first-order logic (with equality and constants, but without other built-in predicates). A *Boolean conjunctive query* is a finite set $q = \{R_1(\underline{\vec{x}_1}, \vec{y}_1), \dots, R_n(\underline{\vec{x}_n}, \vec{y}_n)\}$ of atoms. We denote by $\mathsf{vars}(q)$ the set of variables that occur in q. The set q represents the first-order sentence

$$\exists u_1 \cdots \exists u_k \left(R_1(\underline{\vec{x}_1}, \vec{y}_1) \wedge \cdots \wedge R_n(\underline{\vec{x}_n}, \vec{y}_n) \right),$$

where $\{u_1, \dots, u_k\} = \mathsf{vars}(q)$. This query q is satisfied by uncertain database $\mathbf{db}$ if there exists a valuation θ over $\mathsf{vars}(q)$ such that for each $i \in \{1, \dots, n\}$, $R_i(\underline{\vec{a}}, \vec{b}) \in \mathbf{db}$ with $\vec{a} = \theta(\vec{x}_i)$ and $\vec{b} = \theta(\vec{y}_i)$.

We say that a Boolean conjunctive query q has a *self-join* if some relation name occurs more than once in q. If q has no self-join, then it is called *self-join-free*. By a little abuse of notation, we may confuse atoms with their relation names in a self-join-free Boolean conjunctive query q. That is, if we use a relation name R at places where an atom is expected, then we mean the (unique) R-atom of q.

If q is a Boolean conjunctive query, $\vec{x} = \langle x_1, \dots, x_\ell \rangle$ is a sequence of distinct variables that occur in q, and $\vec{a} = \langle a_1, \dots, a_\ell \rangle$ is a sequence of constants, then $q_{[\vec{x} \mapsto \vec{a}]}$ denotes the query obtained from q by replacing all occurrences of x_i with a_i, for all $1 \leq i \leq \ell$.

Typed uncertain databases. For every variable x, we assume an infinite set of constants, denoted $\mathsf{type}(x)$, such that $x \neq y$ implies $\mathsf{type}(x) \cap \mathsf{type}(y) = \emptyset$. Let q be a self-join-free Boolean conjunctive query, and let $\mathbf{db}$ be an uncertain database. We say that $\mathbf{db}$ is *typed relative to* q if for every atom $R(x_1, \dots, x_n)$ in q, for every $i \in \{1, \dots, n\}$, if x_i is a variable, then for every fact $R(a_1, \dots, a_n)$ in $\mathbf{db}$, $a_i \in \mathsf{type}(x_i)$ and the constant a_i does not occur in q. Significantly, since q is self-join-free, the assumption that uncertain databases are typed is without loss of generality.

Purified uncertain databases. Let q be a Boolean conjunctive query, and let $\mathbf{db}$ be an uncertain database. We say that a fact $A \in \mathbf{db}$ is *relevant for q in* $\mathbf{db}$ if for some valuation θ over $\mathsf{vars}(q)$, $A \in \theta(q) \subseteq \mathbf{db}$. We say that $\mathbf{db}$ is *purified relative to* q if every fact $A \in \mathbf{db}$ is relevant for q in $\mathbf{db}$.

Frugal repairs. For every uncertain database $\mathbf{db}$, Boolean conjunctive query q, and $X \subseteq \mathsf{vars}(q)$, we define a preorder $\preceq_q^X$ on $\mathsf{rset}(\mathbf{db})$, as follows. For every two repairs $\mathbf{r}_1, \mathbf{r}_2$, we define $\mathbf{r}_1 \preceq_q^X \mathbf{r}_2$ if for every valuation θ over X, $\mathbf{r}_1 \models \theta(q)$ implies $\mathbf{r}_2 \models \theta(q)$. Here, $\theta(q)$ is the query obtained from q by replacing all occurrences of each $x \in X$ with $\theta(x)$; variables not in X remain unaffected (i.e., θ is understood to be the identity on variables not in X). Clearly, $\preceq_q^X$ is a preorder (i.e., it is reflexive and transitive), and its minimal elements are called *$\preceq_q^X$-frugal repairs*.[1]

Functional dependencies. Let q be a Boolean conjunctive query. A *functional dependency for* q is an expression $X \rightarrow Y$ where $X, Y \subseteq \mathsf{vars}(q)$. We say that an uncertain database $\mathbf{db}$ *satisfies* $X \rightarrow Y$ *for* q, denoted $\mathbf{db} \Vdash_q X \rightarrow Y$, if for all valuations θ, μ over $\mathsf{vars}(q)$ such that $\theta(q), \mu(q) \subseteq \mathbf{db}$, if $\theta[X] = \mu[X]$, then $\theta[Y] = \mu[Y]$.

Example 1. The relation R shown next does not satisfy the standard functional dependency $2 \rightarrow 3$, because its tuples agree on the second position, but disagree on the third position. Nevertheless, for $q = \exists y \exists z R(\underline{a}, y, z)$, we have $R \Vdash_q y \rightarrow z$. The second tuple of R is not relevant for the query, because a and d are distinct constants; the relation R' is purified relative to q.

R	$\underline{1}$	2	3
	a	b	c
	d	b	f

R'	$\underline{1}$	2	3
	a	b	c

[1] $\mathbf{r}_1$ is minimal if for all $\mathbf{r}_2$, if $\mathbf{r}_2 \preceq_q^X \mathbf{r}_1$ then $\mathbf{r}_1 \preceq_q^X \mathbf{r}_2$.

Consistent query answering. For every Boolean conjunctive query q, the decision problem $\mathsf{CERTAINTY}(q)$ takes as input an uncertain database $\mathbf{db}$, and asks whether q is satisfied by every repair of $\mathbf{db}$. It is straightforward that for every Boolean first-order query q, $\mathsf{CERTAINTY}(q)$ is in $\mathbf{coNP}$.

The following two lemmas are useful in the study of the complexity of $\mathsf{CERTAINTY}(q)$.

LEMMA 1 ([19]). *Let q be a Boolean conjunctive query. Let $\mathbf{db}$ be an uncertain database. It is possible to compute in polynomial time an uncertain database $\mathbf{db}'$ that is purified relative to q such that every repair of $\mathbf{db}$ satisfies q if and only if every repair of $\mathbf{db}'$ satisfies q.*

LEMMA 2. *Let q be a Boolean conjunctive query, and $X \subseteq \mathsf{vars}(q)$. Let $\mathbf{db}$ be an uncertain database. Then, every repair of $\mathbf{db}$ satisfies q if and only if every $\preceq_q^X$-frugal repair of $\mathbf{db}$ satisfies q.*

4. ATTACK GRAPHS

Attack graphs were introduced in [16] for studying first-order expressibility of $\mathsf{CERTAINTY}(q)$ for acyclic (in the sense of [3]) self-join-free conjunctive queries q. Here, we extend the notion of attack graph to all (cyclic or acyclic) self-join-free conjunctive queries.

Let q be a self-join-free Boolean conjunctive query. We define $\mathcal{K}(q)$ as the following set of functional dependencies:

$$\mathcal{K}(q) := \{\mathsf{key}(F) \rightarrow \mathsf{vars}(F) \mid F \in q\}.$$

For every atom $F \in q$, we define $F^{+,q}$ as the following set of variables:

$$F^{+,q} := \{x \in \mathsf{vars}(q) \mid \mathcal{K}(q \setminus \{F\}) \models \mathsf{key}(F) \rightarrow x\}.$$

The *attack graph of* q is a directed graph whose vertices are the atoms of q. There is a directed edge from F to G ($F \neq G$) if there exists a sequence

$$F_0, F_1, \dots, F_n \tag{1}$$

of (not necessarily distinct) atoms of q such that

- $F_0 = F$ and $F_n = G$; and
- for all $i \in \{0, \dots, n-1\}$, $\mathsf{vars}(F_i) \cap \mathsf{vars}(F_{i+1}) \not\subseteq F^{+,q}$.

A directed edge from F to G in the attack graph of q is also called an *attack from F to G*, denoted by $F \overset{q}{\rightsquigarrow} G$. The sequence (1) is called a *witness* for the attack $F \overset{q}{\rightsquigarrow} G$. We will often add variables to a witness: if we write $F_0 \overset{z_1}{\frown} F_1 \overset{z_2}{\frown} F_2 \dots \overset{z_n}{\frown} F_n$, then it is understood that for $i \in \{1, \dots, n\}$, $z_i \in \mathsf{vars}(F_{i-1}) \cap \mathsf{vars}(F_i)$ and $z_i \notin {F_0}^{+,q}$. If $F \overset{q}{\rightsquigarrow} G$, then we also say that F *attacks* G (or that G is attacked by F).

An attack from F to G is called *weak* if $\mathcal{K}(q) \models \mathsf{key}(F) \rightarrow \mathsf{key}(G)$; otherwise it is *strong*. A directed cycle in the attack graph of q is called *weak* if all attacks in the cycle are weak; otherwise the cycle is called *strong*.

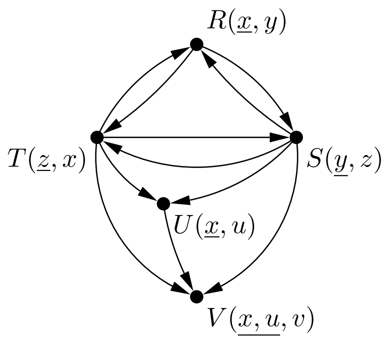


Figure 1: Attack graph of the query in Example 2.

Example 2. Consider $q = \{R(\underline{x}, y), S(\underline{y}, z), T(\underline{z}, x), U(\underline{x}, u), V(\underline{x, u}, v)\}$. By a little abuse of notation, we denote each atom by its relation name (e.g., R is used to denote the atom $R(\underline{x}, y)$). We have $R^{+,q} = \{x, u, v\}$. A witness for $R \overset{q}{\rightsquigarrow} T$ is $R \overset{y}{\frown} S \overset{z}{\frown} T$. The complete attack graph is shown in Fig. 1. All attacks are weak.

The above notion of attack graph is purely syntactic. Semantically, an attack from an R-atom to an S-atom in the attack graph of q means that there exists an uncertain database $\mathbf{db}$ such that every repair of $\mathbf{db}$ satisfies q, and such that two key-equal R-facts join exclusively with two S-facts that are not key-equal. For the query of Example 2, such a database could be $\mathbf{db} = \{R(\underline{1}, a), R(\underline{1}, b), S(\underline{a}, \alpha), S(\underline{b}, \beta), \dots\}$, in which the two R-facts are key-equal, $R(\underline{1}, a)$ joins exclusively with $S(\underline{a}, \alpha)$, and $R(\underline{1}, b)$ joins exclusively with $S(\underline{b}, \beta)$, and the two S-facts are not key-equal. Therefore, the attack graph of Fig. 1 contains a directed edge from the R-atom to the S-atom.

Equipped with the notion of attack graph, we can now present our threefold solution to the complexity classification task of CERTAINTY(q) for q ranging over the class of self-join-free Boolean conjunctive queries.

THEOREM 1. *Let q be a self-join-free Boolean conjunctive query.*

1. *If the attack graph of q is acyclic, then* CERTAINTY(q) *is in* $\mathbf{FO}$.
2. *If the attack graph of q is cyclic but contains no strong cycle, then* CERTAINTY(q) *is in* $\mathbf{P}$ *and is* $\mathbf{L}$*-hard.*
3. *If the attack graph of q contains a strong cycle, then* CERTAINTY(q) *is* $\mathbf{coNP}$*-complete.*

Furthermore, it can be decided in polynomial time in the size of q which of the above three cases applies.

The following lemma establishes that the three if-conditions in Theorem 1 can be decided in polynomial time.

LEMMA 3. *Given a self-join-free Boolean conjunctive query q, the following questions can be answered in polynomial time in the size of q:*

1. *Does the attack graph of q contain a strong cycle?*
2. *Does the attack graph of q contain a weak cycle?*

The rest of the paper completes the proof of Theorem 1. We first present some properties of attack graphs that will be useful in subsequent sections.

LEMMA 4. *Let q be a self-join-free Boolean conjunctive query. If $F \overset{q}{\rightsquigarrow} G$ and $G \overset{q}{\rightsquigarrow} H$, then either $F \overset{q}{\rightsquigarrow} H$ or $G \overset{q}{\rightsquigarrow} F$ (or both).*

LEMMA 5. *Let q be a self-join-free Boolean conjunctive query.*

1. *If the attack graph of q contains a cycle, then it contains a cycle of size two.*
2. *If the attack graph of q contains a strong cycle, then it contains a strong cycle of size two.*

LEMMA 6. *Let q be a self-join-free Boolean conjunctive query. Let $x \in \mathsf{vars}(q)$ and let a be an arbitrary constant.*

1. *If the attack graph of q is acyclic, then the attack graph of $q_{[x \mapsto a]}$ is acyclic.*
2. *If the attack graph of q contains no strong cycle, then the attack graph of $q_{[x \mapsto a]}$ contains no strong cycle.*

We conclude this section with two definitions. The following definition is taken from [2] and applies to directed graphs.

Definition 1. A directed graph is *strongly connected* if there is a directed path from any vertex to any other. The maximal strongly connected subgraphs of a graph are vertex-disjoint and are called its *strong components*. If S_1 and S_2 are strong components such that an edge leads from a vertex in S_1 to a vertex in S_2, then S_1 is a *predecessor* of S_2 and S_2 is a *successor* of S_1. A strong component is called *initial* if it has no predecessor.

Example 3. In the attack graph of Fig. 1, the atoms $R(\underline{x}, y)$, $S(\underline{y}, z)$, and $T(\underline{z}, x)$ together form an initial strong component.

So far we have defined an attack from an atom to another atom. The following definition introduces attacks from an atom to a variable of the query.

Definition 2. Let q be a self-join-free Boolean conjunctive query. Let R be a relation name with signature $[1, 1]$ such that R does not occur in q. For $F \in q$ and $z \in \mathsf{vars}(q)$, we say that F *attacks* z, denoted $F \overset{q}{\rightsquigarrow} z$, if $F \overset{q'}{\rightsquigarrow} R(\underline{z})$ where $q' = q \cup \{R(\underline{z})\}$.

Example 4. Clearly, if $F_0 \overset{z_1}{\frown} F_1 \dots \overset{z_n}{\frown} F_n$ is a witness for $F_0 \overset{q}{\rightsquigarrow} F_n$, then $F_0 \overset{q}{\rightsquigarrow} z_i$ for every $i \in \{1, \dots, n\}$. Notice also that if $q = \{R(\underline{x}, y)\}$, then the attack graph of q contains no edge, yet $R \overset{q}{\rightsquigarrow} y$.

5. FIRST-ORDER EXPRESSIBILITY

In this section, we prove the first item in the statement of Theorem 1, as well as the $\mathbf{L}$-hard lower complexity bound stated in the second item. Taken together, this leads to the following characterization of first-order expressibility of CERTAINTY(q).

THEOREM 2. *Let q be a self-join-free Boolean conjunctive query. Then the following are equivalent:*

1. CERTAINTY(q) *is in* $\mathbf{FO}$*;*
2. *the attack graph of q is acyclic.*

That is, acyclicity of the attack graph of q is both a necessary and sufficient condition for first-order expressibility of CERTAINTY(q). In Section 5.1, we show the contrapositive of the implication 1 $\Longrightarrow$ 2. In Section 5.2, we show the implication 2 $\Longrightarrow$ 1.

5.1 Necessary Condition

Let $q_0 = \{R_0(\underline{x}, y), S_0(\underline{y}, x)\}$. In [17], it was shown that $\mathsf{CERTAINTY}(q_0)$ is not in $\mathbf{FO}$. The following lemma shows a stronger result.

LEMMA 7. *Let* $q_0 = \{R_0(\underline{x}, y), S_0(\underline{y}, x)\}$. *Then* $\mathsf{CERTAINTY}(q_0)$ *is* $\mathbf{L}$*-hard.*

LEMMA 8. *Let q be a self-join-free Boolean conjunctive query. If the attack graph of q is cyclic, then* $\mathsf{CERTAINTY}(q)$ *is* $\mathbf{L}$*-hard (and hence not in* $\mathbf{FO}$*).*

PROOF. (*Outline.*) The proof shows that if the attack graph of q is cyclic, then there exists a first-order many-one reduction from $\mathsf{CERTAINTY}(q_0)$ to $\mathsf{CERTAINTY}(q)$. The desired result then follows from Lemma 7. The detailed proof is available in [10]. □

5.2 Sufficient Condition

In this section, we show that $\mathsf{CERTAINTY}(q)$ is in $\mathbf{FO}$ if the attack graph of q is acyclic.

LEMMA 9. *Let q be a self-join-free Boolean conjunctive query. Let F be an atom of q such that in the attack graph of q, the indegree of F is zero. Let* $k = |\mathsf{key}(F)|$ *and let* $\vec{x} = (x_1, \dots, x_k)$ *be a sequence containing (exactly once) each variable of* $\mathsf{key}(F)$*. Then the following are equivalent for every uncertain database* $\mathbf{db}$*:*

1. *q is true in every repair of* $\mathbf{db}$*;*
2. *for some* $\vec{a} \in (\mathsf{adom}(\mathbf{db}))^k$*, it is the case that* $q_{[\vec{x}\mapsto\vec{a}]}$ *is true in every repair of* $\mathbf{db}$.

Lemma 9 immediately leads to the following result.

LEMMA 10. *Let q be a self-join-free Boolean conjunctive query. If the attack graph of q is acyclic, then* $\mathsf{CERTAINTY}(q)$ *is in* $\mathbf{FO}$.

PROOF. Assume that the attack graph of q is acyclic. The proof runs by induction on $|q|$. If $|q| = 0$, then $\mathsf{CERTAINTY}(q)$ is obviously in $\mathbf{FO}$.

Let $\mathbf{db}$ be an instance of $\mathsf{CERTAINTY}(q)$. Since the attack graph of q is acyclic, we can assume an atom $R(\underline{\vec{x}}, \vec{y})$ that is not attacked in the attack graph of q. By Lemma 9, the following are equivalent:

1. q is true in every repair of $\mathbf{db}$.
2. For some fact $R(\underline{\vec{a}}, \vec{b}) \in \mathbf{db}$, there exists of a valuation θ over $\mathsf{vars}(\vec{x})$ such that $\theta(\vec{x}) = \vec{a}$ and such that for all key-equal facts $R(\underline{\vec{a}}, \vec{b}')$ in $\mathbf{db}$, the valuation θ can be extended to a valuation θ^+ over $\mathsf{vars}(\vec{x}) \cup \mathsf{vars}(\vec{y})$ such that $\theta^+(\vec{y}) = \vec{b}$ and $\theta^+(q')$ is true in every repair of $\mathbf{db}$, where $q' = q \setminus \{R(\underline{\vec{x}}, \vec{y})\}$.

From Lemma 6, it follows that the attack graph of $\theta^+(q')$ is acyclic, and hence $\mathsf{CERTAINTY}(\theta^+(q'))$ is in $\mathbf{FO}$ by the induction hypothesis. It is then clear that the latter condition 2 can be checked in $\mathbf{FO}$. □

For a self-join-free Boolean conjunctive query q, the problem $\mathsf{CERTAINTY}(q)$ can be equivalently defined as the set containing every uncertain database $\mathbf{db}$ such that every repair of $\mathbf{db}$ satisfies q. If $\mathsf{CERTAINTY}(q)$ is in $\mathbf{FO}$, then the set $\mathsf{CERTAINTY}(q)$ is definable in first-order logic (by definition of the complexity class $\mathbf{FO}$). If $\mathsf{CERTAINTY}(q)$ is in $\mathbf{FO}$, then its first-order definition is commonly called *first-order rewriting*. Such a first-order rewriting is actually an implementation, in first-order logic, of the algorithm in the proof of Lemma 10. This is illustrated next.

Example 5. Let $q = \{R(\underline{x}, y), S(\underline{y}, b)\}$, where b is a constant. The attack graph of q contains a single directed edge, from the R-atom to the S-atom. The first-order definition of $\mathsf{CERTAINTY}(q)$ is as follows:

$$\begin{aligned}&\exists x \exists y (R(\underline{x}, y) \wedge \\ &\quad \forall y \left(R(\underline{x}, y) \rightarrow \left(S(\underline{y}, b) \wedge \forall z \left(S(\underline{y}, z) \rightarrow z = b\right)\right)\right)).\end{aligned}$$

6. POLYNOMIAL-TIME TRACTABILITY

In this section, we prove the $\mathbf{coNP}$-hard lower complexity bound stated in the third item of Theorem 1, as well as the $\mathbf{P}$ upper complexity bound stated in the second item of Theorem 1. Those complexity bounds are recalled in the following two theorems.

THEOREM 3. *Let q be a self-join-free Boolean conjunctive query. If the attack graph of q contains a strong cycle, then* $\mathsf{CERTAINTY}(q)$ *is* $\mathbf{coNP}$*-hard.*

THEOREM 4. *Let q be a self-join-free Boolean conjunctive query. If the attack graph of q contains no strong cycle, then* $\mathsf{CERTAINTY}(q)$ *is in* $\mathbf{P}$.

The proof of Theorem 3 is available in [10], and differs little from the proof of Theorem 2 in [19]. In the remainder of this section, we elaborate on the proof of Theorem 4.

A very high-level outline of the proof of Theorem 4 is as follows. Let q be a self-join-free Boolean conjunctive query such that the attack graph of q contains no strong cycle. The proof will run by induction on syntax. If the attack graph of q contains an atom F without incoming attacks, then Lemma 9 tells us that the answer to $\mathsf{CERTAINTY}(q)$ can be obtained in polynomial time from the answers to a polynomial number of problems $\mathsf{CERTAINTY}(q')$, all of which are in polynomial time by induction hypothesis. The more difficult case is if all atoms have incoming attacks in the attack graph of q. We will show that in this case, $\mathsf{CERTAINTY}(q)$ can be reduced in polynomial time to some problem $\mathsf{CERTAINTY}(q'')$ which is in polynomial time by induction hypothesis. The query q'' is obtained from q by a technique called *"dissolution of Markov cycles."*

Road map. The detailed proof of Theorem 4 is technically involved. We start by introducing in Section 6.1 an extension of the data model that allows some syntactic simplifications, expressed in Section 6.2. In Section 6.3, we introduce the notion of *Markov cycle*, and show how the "dissolution" of Markov cycles is helpful in the proof of Theorem 4, which is given in Section 6.4. The dissolution of Markov cycles is explained in detail in Section 6.5.

6.1 Relations Known to Be Consistent

We conservatively extend our data model. We first distinguish between two kinds of relation names: those that can be inconsistent, and those that cannot.

Relations known to be consistent. Every relation name has a unique and fixed *mode*, which is an element in $\{i, c\}$. It will come in handy to think of i and c as inconsistent and consistent respectively. We often write R^c to denote that R is a relation name with mode c. If q is a self-join-free Boolean conjunctive query, then $[\![q]\!]$ denotes the subset of q containing each atom whose relation name has mode c. The *inconsistency count* of q, denoted $\mathsf{incnt}(q)$, is the number of relation names with mode i in q. Modes carry over to atoms and facts: the mode of an atom $R(\underline{\vec{x}}, \vec{y})$ or a fact $R(\underline{\vec{a}}, \vec{b})$ is the mode of R.

The intended semantics is that if a relation name R has mode c, then the set of R-facts of an uncertain database will always be consistent.

Certain query answering with consistent and inconsistent relations. The problem $\mathsf{CERTAINTY}(q)$ now takes as input an uncertain database $\mathbf{db}$ such that for every relation name R in q, if R has mode c, then the set of R-facts of $\mathbf{db}$ is consistent. The problem is to determine whether every repair of $\mathbf{db}$ satisfies q.

All constructs and results shown in previous sections assumed that all relation names had mode i. Nevertheless, Proposition 1 (which has an easy proof) indicates that in the tractability study of $\mathsf{CERTAINTY}(q)$, relation names with mode c can be simulated by means exclusively of relation names with mode i. Therefore, having relation names with mode c will be convenient, but is not fundamental.

PROPOSITION 1. *Let q be a self-join free Boolean conjunctive query. Let $R^c(\underline{\vec{x}}, \vec{y})$ be an atom with mode c in q. Let R_1 and R_2 be two relation names, both with mode i and with the same signature as R, such that neither R_1 nor R_2 occurs in q. Let $q' = (q \setminus \{R^c(\underline{\vec{x}}, \vec{y})\}) \cup \{R_1(\underline{\vec{x}}, \vec{y}), R_2(\underline{\vec{x}}, \vec{y})\}$. Then $\mathsf{CERTAINTY}(q)$ and $\mathsf{CERTAINTY}(q')$ are equivalent under first-order reductions.*

If relation names with mode c are allowed for syntactic convenience, the definition of $F^{+,q}$ needs slight change:

$$F^{+,q} := \{x \in \mathsf{vars}(q) \mid \mathcal{K}((q \setminus F) \cup [\![q]\!]) \models \mathsf{key}(F) \rightarrow x\}.$$

Modulo this redefinition, the notion of attack graph remains unchanged.

Proposition 1 explains how to replace atoms with mode c. Conversely, the following lemma states that in pursuing a proof for Theorem 4, there are cases where a self-join-free Boolean conjunctive query can be extended with atoms of mode c.

LEMMA 11. *Let q be a self-join-free Boolean conjunctive query. Let $x, z \in \mathsf{vars}(q)$ such that $\mathcal{K}(q) \models x \rightarrow z$ and for every $F \in q$, if $\mathcal{K}(q) \models x \rightarrow \mathsf{key}(F)$, then $F \overset{q}{\not\rightsquigarrow} x$ and $F \overset{q}{\not\rightsquigarrow} z$. Let $q' = q \cup \{T^c(\underline{x}, z)\}$, where T is a fresh relation name with mode c. Then,*

1. *there exists a polynomial-time many-one reduction from $\mathsf{CERTAINTY}(q)$ to $\mathsf{CERTAINTY}(q')$; and*
2. *if the attack graph of q contains no strong cycle, then the attack graph of q' contains no strong cycle either.*

Saturated queries. Given a self-join-free Boolean conjunctive query, the reduction of Lemma 11 can be repeated until it can no longer be applied. The query so obtained will be called *saturated*.

Definition 3. Let q be a self-join-free Boolean conjunctive query. We say that q is *saturated* if whenever $x, z \in \mathsf{vars}(q)$ such that $\mathcal{K}(q) \models x \rightarrow z$ and $\mathcal{K}([\![q]\!]) \not\models x \rightarrow z$, then there exists an atom $F \in q$ with $\mathcal{K}(q) \models x \rightarrow \mathsf{key}(F)$ such that $F \overset{q}{\rightsquigarrow} x$ or $F \overset{q}{\rightsquigarrow} z$.

Example 6. Let $q = \{R(\underline{x}, y), S_1(\underline{y}, z), S_2(\underline{y}, z), T^c(\underline{x, z}, w), U(\underline{w}, x)\}$. We have $\mathcal{K}(q) \models y \rightarrow z$ and $\mathcal{K}([\![q]\!]) \not\models y \rightarrow z$. The set $\{F \in q \mid \mathcal{K}(q) \models y \rightarrow \mathsf{key}(F)\}$ equals $\{S_1, S_2\}$. We have neither $S_1 \overset{q}{\rightsquigarrow} y$ nor $S_1 \overset{q}{\rightsquigarrow} z$. Likewise, neither $S_2 \overset{q}{\rightsquigarrow} y$ nor $S_2 \overset{q}{\rightsquigarrow} z$. Hence, q is not saturated. By Lemma 11, there exists a polynomial-time many-one reduction from $\mathsf{CERTAINTY}(q)$ to $\mathsf{CERTAINTY}(q')$ with $q' = q \cup \{S^c(\underline{y}, z)\}$, where S is a fresh relation name with mode c. It can be verified that the query q' is saturated.

6.2 Syntactic Simplifications

The following lemma shows that any proof of Theorem 4 can assume some syntactic simplifications without loss of generality.

LEMMA 12. *Let q be a self-join-free Boolean conjunctive query. There exists a polynomial-time many-one reduction from $\mathsf{CERTAINTY}(q)$ to $\mathsf{CERTAINTY}(q')$ for some self-join-free Boolean conjunctive query q' with the following properties:*

- $\mathsf{incnt}(q') \leq \mathsf{incnt}(q)$*;*
- *no atom in q' contains two occurrences of the same variable;*
- *constants occur in q' exclusively at the primary-key position of simple-key atoms;*
- *every atom with mode i in q' is simple-key;*
- *q' is saturated; and*
- *if the the attack graph of q contains no strong cycle, then the attack graph of q' contains no strong cycle either.*

6.3 Dissolving Markov Cycles

The following definition introduces Markov graphs.

Definition 4. Let q be a self-join-free Boolean conjunctive query such that every atom with mode i in q is simple-key. For every $x \in \mathsf{vars}(q)$, we define

$$\mathsf{C}_q(x) := \{F \in q \mid F \text{ has mode } i \text{ and } \mathsf{key}(F) = \{x\}\}.$$

Notice that $\mathsf{C}_q(x)$ can be empty.

The *Markov graph* of q is a directed graph whose vertex set is $\mathsf{vars}(q)$. There is a directed edge from x to y, denoted $x \xrightarrow{q,\mathsf{M}} y$, if $x \neq y$ and $\mathcal{K}(\mathsf{C}_q(x) \cup [\![q]\!]) \models x \rightarrow y$. If the query q is clear from the context, then $x \xrightarrow{q,\mathsf{M}} y$ can be shortened into $x \xrightarrow{\mathsf{M}} y$. We write $x \xrightarrow{q,\mathsf{M}*} y$ (or $x \xrightarrow{\mathsf{M}*} y$ if q is clear from the context) if the Markov graph of q contains a directed path from x to y.[2] Notice that for every $x \in \mathsf{vars}(q)$, $x \xrightarrow{q,\mathsf{M}*} x$.

An elementary directed cycle $\mathcal{C}$ in the Markov graph of q is said to be *premier* if there exists a variable $x \in \mathsf{vars}(q)$ such that

1. $\{x\} = \mathsf{key}(F_0)$ for some atom F_0 with mode i that belongs to an initial strong component of the attack graph of q; and
2. for some y in $\mathcal{C}$, we have $x \xrightarrow{q,\mathsf{M}*} y$ and $\mathcal{K}(q) \models y \rightarrow x$.

The term *Markov edge* is used for an edge in the Markov graph; likewise for *Markov path* and *Markov cycle*.

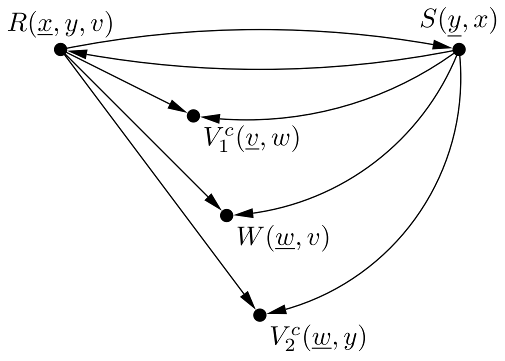
Example 7. Let $q = \{R(\underline{x}, y, v),\ S(\underline{y}, x),\ V_1^c(\underline{v}, w),\ W(\underline{w}, v)\ V_2^c(\underline{w}, y)\}$. All atoms in q are simple-key. Then, $[\![q]\!] = \{V_1^c(\underline{v}, w), V_2^c(\underline{w}, y)\}$.

We have $\mathsf{C}_q(x) = \{R(\underline{x}, v, y)\}$. Since $\mathcal{K}(\mathsf{C}_q(x) \cup [\![q]\!]) \models x \rightarrow \{y, v, w\}$, the Markov graph of q contains directed edges from x to each of y, v, and w.

We have $\mathsf{C}_q(v) = \emptyset$. Since $\mathcal{K}(\mathsf{C}_q(v) \cup [\![q]\!]) \models v \rightarrow \{y, w\}$, the Markov graph of q contains directed edges from v to both y and w. The complete Markov graph of q is shown in Fig. 2 (right).

The attack graph of q is shown in Fig. 2 (left). The two atoms $R(\underline{x}, y, v)$ and $S(\underline{y}, x)$ together constitute an initial strong component of the attack graph. It is then straightforward that each cycle in the Markov graph of q that contains x or y, must be premier. Further, the cycle v, w, v in the Markov graph of q is also premier, because there is a Markov path from x to v, and $\mathcal{K}(q) \models v \rightarrow x$.

[2]The term Markov refers to the intuition that in a Markov path, each variable functionally determines the next variable in the path, independently of preceding variables.



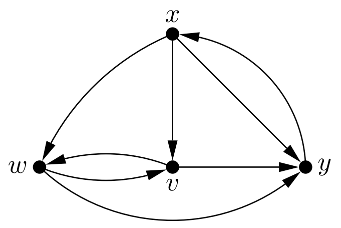


Figure 2: Attack graph (left) and Markov graph (right) of the query $\{R(\underline{x},y,v)$, $S(\underline{y},x)$, $V_1^c(\underline{v},w)$, $W(\underline{w},v)$ $V_2^c(\underline{w},y)\}$.

Let q be like in Definition 4 and assume that the Markov graph of q contains an elementary directed cycle $\mathcal{C}$. Lemma 13 states that $\mathsf{CERTAINTY}(q)$ can be reduced in polynomial time to $\mathsf{CERTAINTY}(q^*)$, where q^* is obtained from q by "dissolving" the Markov cycle $\mathcal{C}$ as defined in Definition 5. Moreover, we will show (Lemma 14) that if $\mathcal{C}$ is premier and the attack graph of q contains no strong cycle, then the attack graph of q^* will contain no strong cycle either. The reduction that "dissolves" Markov cycles will be the central idea in our polynomial-time algorithm for $\mathsf{CERTAINTY}(q)$ when the attack graph of q contains no strong cycle.

Definition 5. Let q be a self-join-free Boolean conjunctive query such that every atom with mode i in q is simple-key. Let $\mathcal{C}$ be an elementary directed cycle of length $k \geq 2$ in the Markov graph of q. Then, $\mathsf{dissolve}(\mathcal{C},q)$ denotes the self-join-free Boolean conjunctive query defined next. Let $x_0,\dots,x_{k-1}$ be the variables in $\mathcal{C}$, and let $q_0 = \bigcup_{i=0}^{k-1} \mathsf{C}_q(x_i)$. Let $\vec{y}$ be a sequence of variables containing exactly once each variable of $\mathsf{vars}(q_0) \setminus \{x_0,\dots,x_{k-1}\}$. Let $q_1 = \{T(\underline{u},x_0,\dots,x_{k-1},\vec{y})\} \cup \{U_i^c(\underline{x_i},u)\}_{i=0}^{k-1}$, where u is a fresh variable, T is a fresh relation name with mode i, and $U_1,\dots,U_{k-1}$ are fresh relation names with mode c. Then, we define

$$\mathsf{dissolve}(\mathcal{C},q) := (q \setminus q_0) \cup q_1.$$

Notice that $\mathsf{dissolve}(\mathcal{C},q)$ is unique up to a renaming of the variable u and the relation names in q_1.

Example 8. Let q be the query of Fig. 2. Let $\mathcal{C}$ be the cycle x,w,y,x in the Markov graph of q. Using the notation of Definition 5, we have

$$\begin{aligned} q_0 &= \{R(\underline{x},y,v), S(\underline{y},x), W(\underline{w},v)\}, \\ q_1 &= \{T(\underline{u},x,w,y,v), U_1^c(\underline{x},u), U_2^c(\underline{w},u), U_3^c(\underline{y},u)\}. \end{aligned}$$

Hence, $\mathsf{dissolve}(\mathcal{C},q) = \{V_1^c(\underline{v},w),\ V_2^c(\underline{w},y),\ T(\underline{u},x,w,y,v),$ $U_1^c(\underline{x},u), U_2^c(\underline{w},u), U_3^c(\underline{y},u)\}$.

LEMMA 13. *Let q be a self-join-free Boolean conjunctive query such that every atom with mode i in q is simple-key. Let $\mathcal{C}$ be an elementary directed cycle in the Markov graph of q, and let $q^* = \mathsf{dissolve}(\mathcal{C},q)$. Then, there exists a polynomial-time many-one reduction from $\mathsf{CERTAINTY}(q)$ to $\mathsf{CERTAINTY}(q^*)$.*

The reduction of Lemma 13 will be explained in Section 6.5. To use the reduction in a proof of Theorem 4, two more results are needed:

- First, we need to show that the "dissolution" of Markov cycles can be done while keeping the attack graph free of strong cycles (this is Lemma 14). This turns out to be true only for Markov cycles that are premier (as defined in Definition 4).
- Second, we need to show the existence of premier Markov cycles that can be "dissolved" (this is Lemma 15).

LEMMA 14. *Let q be a self-join-free Boolean conjunctive query such that every atom with mode i in q is simple-key. Let $\mathcal{C}$ be an elementary directed cycle in the Markov graph of q such that $\mathcal{C}$ is premier, and let $q^* = \mathsf{dissolve}(\mathcal{C},q)$. If the attack graph of q contains no strong cycle, then the attack graph of q^* contains no strong cycle either.*

LEMMA 15. *Let q be a self-join-free Boolean conjunctive query such that*

- *for every atom $F \in q$, if F has mode i, then F is simple-key and $\mathsf{key}(F) \neq \emptyset$;*
- *q is saturated;*
- *the attack graph of q contains no strong cycle; and*
- *the attack graph of q contains an initial strong component with two or more atoms.*

Then, the Markov graph of q contains an elementary directed cycle that is premier and such that for every y in $\mathcal{C}$, $\mathsf{C}_q(y) \neq \emptyset$.

The condition $\mathsf{C}_q(y) \neq \emptyset$, for every y in $\mathcal{C}$, guarantees that $\mathsf{dissolve}(\mathcal{C},q)$ will contain strictly less atoms of mode i than q. This condition will be used in the proof of Theorem 4 which runs by induction on the number of atoms with mode i. The following example shows that Lemma 15 is no longer true if q is not saturated.

Example 9. Continuing Example 6. The query q of Example 6 is not saturated, but satisfies all other conditions in the statement of Lemma 15. In particular, the attack graph of q contains a weak cycle $R \overset{q}{\rightsquigarrow} U \overset{q}{\rightsquigarrow} R$, which is part of an initial strong component. The Markov graph of q consists of a single path $w \xrightarrow{q,\mathsf{M}} x \xrightarrow{q,\mathsf{M}} y \xrightarrow{q,\mathsf{M}} z$, and hence is acyclic.

The query q' of Example 6 is saturated, and we have $x \xrightarrow{q',\mathsf{M}} w \xrightarrow{q',\mathsf{M}} x$, a Markov cycle which can be shown to be premier.

6.4 The Proof of Theorem 4

PROOF OF THEOREM 4. Assume that the attack graph of q contains no strong cycle. The proof runs by induction on increasing

$\mathsf{incnt}(q)$. The desired result is obvious if $\mathsf{incnt}(q) = 0$. Assume that $\mathsf{incnt}(q) > 0$ in the remainder of the proof. Let $\mathbf{db}$ be an uncertain database that is input to $\mathsf{CERTAINTY}(q)$.

First, we reduce in polynomial time $\mathsf{CERTAINTY}(q)$ to $\mathsf{CERTAINTY}(q')$ with q' like in Lemma 12. We now distinguish two cases.

Case q' contains an atom F with mode i that has zero indegree in the attack graph of q. We can assume either $F = R(\underline{x}, \vec{y})$ or $F = R(\underline{a}, \vec{y})$, where $\vec{y}$ is a sequence of distinct variables. In the remainder, we treat the case $F = R(\underline{x}, \vec{y})$ (the case $F = R(\underline{a}, \vec{y})$ is even simpler).

Let $q'' = q' \setminus \{R(\underline{x}, \vec{y})\}$. By Lemma 9, every repair of $\mathbf{db}$ satisfies q' if and only if $\mathbf{db}$ includes an R-block $\mathbf{b}$ (there are only polynomially many such blocks) such for every $R(\underline{a}, \vec{b}) \in \mathbf{b}$, every repair of $\mathbf{db}$ satisfies ${q''}_{[x,\vec{y} \mapsto a,\vec{b}]}$. By Lemma 6, the attack graph of ${q''}_{[x,\vec{y} \mapsto a,\vec{b}]}$ contains no strong cycle. From $\mathsf{incnt}({q''}_{[x,\vec{y} \mapsto a,\vec{b}]}) = \mathsf{incnt}(q') - 1 < \mathsf{incnt}(q)$, it follows that $\mathsf{CERTAINTY}({q''}_{[x,\vec{y} \mapsto a,\vec{b}]})$ is in $\mathbf{P}$ by the induction hypothesis. It follows that $\mathsf{CERTAINTY}(q)$ is in $\mathbf{P}$ as well.

Case every atom F with mode i in q' has an incoming attack in the attack graph of q'. It will be the case that no constant occurs in an atom of mode i in q'.

Then, the attack graph of q' must contain an initial strong component with two or more atoms. By Lemma 15, the Markov graph of q' contains an elementary directed cycle $\mathcal{C}$ that is premier and such that for every y in $\mathcal{C}$, $\mathsf{C}_{q'}(y) \neq \emptyset$. By Lemma 13, we can reduce in polynomial time $\mathsf{CERTAINTY}(q')$ to $\mathsf{CERTAINTY}(q^*)$ where $q^* = \mathsf{dissolve}(\mathcal{C}, q')$. Since the attack graph of q' contains no strong cycle, it follows by Lemma 14 that the attack graph of q^* contains no strong cycle either.

Let $k \geq 2$ be the size of $\mathcal{C}$. It is easy to verify that $\mathsf{incnt}(q^*) \leq (\mathsf{incnt}(q') - k) + 1 < \mathsf{incnt}(q')$. By the induction hypothesis, $\mathsf{CERTAINTY}(q^*)$ is in $\mathbf{P}$. Since there exists a polynomial-time reduction from $\mathsf{CERTAINTY}(q)$ to $\mathsf{CERTAINTY}(q^*)$, we conclude that $\mathsf{CERTAINTY}(q)$ is in $\mathbf{P}$ as well. $\square$

6.5 The Reduction of Lemma 13

This section first describes the reduction of Lemma 13, and then proves the lemma.

Relevance of subsets of repairs. In Section 3, we distinguished database facts that are relevant for a query from those that are not. This notion is extended next.

Definition 6. Let q be a self-join-free Boolean conjunctive query, and let $\mathbf{db}$ be an uncertain database. A consistent subset $\mathbf{s}$ of $\mathbf{db}$ is said to be *grelevant for q in* $\mathbf{db}$ (generalized relevant) if it can be extended into a repair $\mathbf{r}$ of $\mathbf{db}$ such that some fact of $\mathbf{s}$ is relevant for q in $\mathbf{r}$.

It can be seen that $A \in \mathbf{db}$ is relevant for q in $\mathbf{db}$ if and only if $\{A\}$ is grelevant for q in $\mathbf{db}$. Therefore, "grelevant" is a notion that generalises "relevant."

LEMMA 16. *Let q be a self-join-free Boolean conjunctive query, and let $\mathbf{db}$ be an uncertain database. Let $\mathbf{s}$ be a consistent subset of $\mathbf{db}$ that is not grelevant for q in $\mathbf{db}$. Let $\mathbf{db}_0 = \bigcup\{\mathsf{block}(A, \mathbf{db}) \mid A \in \mathbf{s}\}$. Then, the following are equivalent:*

1. *every repair of* $\mathbf{db}$ *satisfies q;*
2. *every repair of* $\mathbf{db} \setminus \mathbf{db}_0$ *satisfies q.*

PROOF. $\boxed{1 \Longrightarrow 2}$ By contraposition. Let $\mathbf{r}$ be a repair of $\mathbf{db} \setminus \mathbf{db}_0$ that falsifies q. Then, $\mathbf{r} \cup \mathbf{s}$ is a repair of $\mathbf{db}$. If $\mathbf{r} \cup \mathbf{s} \models q$, then it must be the case that $\mathbf{s}$ is grelevant for q in $\mathbf{db}$, a contradiction. We conclude by contradiction that $\mathbf{r} \cup \mathbf{s} \not\models q$. $\boxed{2 \Longrightarrow 1}$ Trivial. $\square$

Introductory example. The following example illustrates the main ideas behind the reduction of Lemma 13.

Example 10. Let q be a self-join-free Boolean conjunctive query. Assume that q includes $q_0 = \{R(\underline{x}, y), S(\underline{y}, z), V(\underline{z}, x)\}$. Then, the Markov graph of q contains a cycle $x \xrightarrow{\mathsf{M}} y \xrightarrow{\mathsf{M}} z \xrightarrow{\mathsf{M}} x$. Let $\mathbf{db}$ be an uncertain database that is purified relative to q. Let $\mathbf{db}_0$ be the subset of $\mathbf{db}$ containing all R-facts, S-facts, and V-facts of $\mathbf{db}$. Assume that the following three tables represent all facts of $\mathbf{db}_0$ (for convenience, we use variables as attribute names, and we blur the distinction between a relation name R and a table representing a set of R-facts).

R	$\underline{x}$	y	S	$\underline{y}$	z	V	$\underline{z}$	x	
	1	a		a	α		α	1	$\mathbf{db}_{01}$
				a	κ		κ	1	
	2	b		b	β		β	2	$\mathbf{db}_{02}$
	2	c		c	γ		γ	2	
	3	d		d	δ		δ	3	$\mathbf{db}_{03}$
	3	e		e	ϵ		ϵ	3	
	4	e		e	δ		δ	4	
	4	f		f	ϕ		ϕ	4	

As indicated, we can partition $\mathbf{db}_0$ into three subsets $\mathbf{db}_{01}$, $\mathbf{db}_{02}$, and $\mathbf{db}_{03}$ whose active domains have, pairwise, no constants in common. Consider each of these three subsets in turn.

1. $\mathbf{db}_{01}$ has two repairs, both satisfying q_0. For every repair $\mathbf{r}$ of $\mathbf{db}$, either $\mathbf{r} \models {q_0}_{[x,y,z \mapsto 1,a,\alpha]}$ or $\mathbf{r} \models {q_0}_{[x,y,z \mapsto 1,a,\kappa]}$.

2. $\mathbf{db}_{02}$ has two repairs, both satisfying q_0. For every repair $\mathbf{r}$ of $\mathbf{db}$, either $\mathbf{r} \models {q_0}_{[x,y,z \mapsto 2,b,\beta]}$ or $\mathbf{r} \models {q_0}_{[x,y,z \mapsto 2,c,\gamma]}$.

3. $\mathbf{db}_{03}$ has 16 repairs, and for $\mathbf{s} := \{R(\underline{3}, d), S(\underline{d}, \delta), V(\underline{\delta}, 4), R(\underline{4}, e), S(\underline{e}, \epsilon), V(\underline{\epsilon}, 3), S(\underline{f}, \phi), V(\underline{\phi}, 4)\}$, we have that $\mathbf{s}$ is a repair of $\mathbf{db}_{03}$ that falsifies q_0. It can be seen that $\mathbf{s}$ is not grelevant for q in $\mathbf{db}$. Then, by Lemma 16, every repair of $\mathbf{db}$ satisfies q if and only if every repair of $\mathbf{db} \setminus \mathbf{db}_{03}$ satisfies q. That is, $\mathbf{db}_{03}$ can henceforth be ignored.

The following table T summarizes our findings. In the first column (named with a fresh variable u), the values 01 and 02 refer to $\mathbf{db}_{01}$ and $\mathbf{db}_{02}$ respectively. The table includes two blocks (separated by a dashed line for clarity). The first block indicates that for every repair $\mathbf{r}$ of $\mathbf{db}$, either $\mathbf{r} \models {q_0}_{[x,y,z \mapsto 1,a,\alpha]}$ or $\mathbf{r} \models {q_0}_{[x,y,z \mapsto 1,a,\kappa]}$. Likewise for the second block.

T	$\underline{u}$	x	y	z
	01	1	a	α
	01	1	a	κ
	02	2	b	β
	02	2	c	γ

The table U_x shown below is the projection of T on attributes x and u. This table must be consistent, because by construction, the active domains of $\mathbf{db}_{01}$ and $\mathbf{db}_{02}$ are disjoint. Likewise for U_y

and U_z.

U_x	$\underline{x}$	u
	1	01
	2	02

U_y	$\underline{y}$	u
	a	01
	b	02
	c	02

U_z	$\underline{z}$	u
	α	01
	κ	01
	β	02
	γ	02

Let $\mathbf{db}'$ be the database that extends $\mathbf{db}$ with all the facts shown in the tables T, U_x, U_y, and U_z.[3] Let $q^* = (q \setminus q_0) \cup \{T(\underline{u}, x, y, z)$, $U_x^c(\underline{x}, u)$, $U_y^c(\underline{y}, u)$, $U_z^c(\underline{z}, u)\}$. From our construction, it follows that every repair of $\mathbf{db}$ satisfies q if and only if every repair of $\mathbf{db}'$ satisfies q^*.

Gblocks and gpurification. The following definition strengthens the notion of purification introduced earlier in Section 3.

Definition 7. Let q be a self-join-free Boolean conjunctive query such that all atoms with mode i in q are simple-key. Let $\mathbf{db}$ be an uncertain database that is purified and typed relative to q. A *gblock* (generalized block) of $\mathbf{db}$ relative to q is a maximal (with respect to $\subseteq$) subset $\mathbf{g}$ of $\mathbf{db}$ such that all facts in $\mathbf{g}$ have mode i and agree on their primary-key position (but may disagree on their relation name). Notice that a gblock has at most polynomially many repairs (in the size of $\mathbf{db}$).[4] We say that $\mathbf{db}$ is *gpurified relative to* q if for every gblock $\mathbf{g}$ of $\mathbf{db}$, every repair of $\mathbf{g}$ is grelevant for q in $\mathbf{db}$.

Clearly, every gblock is the union of one or more blocks. Two facts of the same gblock have the same primary-key value, but can have distinct relation names.

Example 11. Let $q = \{R(\underline{x}, y), S(\underline{x}, y)\}$. Let $\mathbf{db} = \{R(\underline{a}, 1)$, $R(\underline{a}, 2)$, $S(\underline{a}, 1)$, $S(\underline{a}, 2)\}$. Then, $\mathbf{db}$ is purified and typed relative to q. All facts of $\mathbf{db}$ together constitute a gblock. The uncertain database $\mathbf{db}$ is not gpurified, since $\mathbf{s} = \{R(\underline{a}, 1), S(\underline{a}, 2)\}$ is a repair of the gblock, and also a repair of $\mathbf{db}$. However, neither $R(\underline{a}, 1)$ nor $S(\underline{a}, 2)$ is relevant for q in $\mathbf{s}$.

Example 12. Let $q = \{R_1(\underline{x}, y), R_2(\underline{x}, z), S(\underline{y, z})\}$, where the signature of S is $[2, 2]$. Let $\mathbf{db}$ be the uncertain database containing the following facts.

R_1	$\underline{x}$	y
	a	1
	a	2

R_2	$\underline{x}$	z
	a	3
	a	4

S	$\underline{y}$	$\underline{z}$
	1	3
	2	4

Then, $\mathbf{db}$ is purified and typed relative to q. All R_1-facts and R_2-facts together constitute a gblock. A repair of this gblock is $\mathbf{s} = \{R_1(\underline{a}, 1), R_2(\underline{a}, 4)\}$. The uncertain database $\mathbf{db}$ is not gpurified. Indeed, the only repair of $\mathbf{db}$ that extends $\mathbf{s}$ is $\{R_1(\underline{a}, 1), R_2(\underline{a}, 4)$, $S(\underline{1, 3})$, $S(\underline{2, 4})\}$ (call it $\mathbf{r}$). Neither $R_1(\underline{a}, 1)$ nor $R_2(\underline{a}, 4)$ is relevant for q in $\mathbf{r}$.

The following lemma is similar to Lemma 1 and has an easy proof.

LEMMA 17. *Let q be a self-join-free Boolean conjunctive query such that all atoms with mode i in q are simple-key. Let* $\mathbf{db}$ *be an uncertain database that is purified and typed relative to q. It is possible to compute in polynomial time an uncertain database* $\mathbf{db}'$ *that is gpurified relative to q such that every repair of* $\mathbf{db}$ *satisfies q if and only if every repair of* $\mathbf{db}'$ *satisfies q.*

[3]Facts of $\mathbf{db}_0$ can be omitted from $\mathbf{db}'$, but that is not important.

[4]Indeed, since $\mathbf{db}$ is purified relative to q, every gblock of $\mathbf{db}$ contains at most $|q|$ distinct relation names, and hence has at most $|\mathbf{db}|^{|q|}$ distinct repairs.

Specification of the reduction of Lemma 13. Let q and $\mathcal{C}$ be as in the statement of Lemma 13. Assume that the elementary directed cycle $\mathcal{C}$ in the Markov graph of q is $x_0 \xrightarrow{\mathsf{M}} x_1 \cdots \xrightarrow{\mathsf{M}} x_{k-1} \xrightarrow{\mathsf{M}} x_0$. In what follows, let $\mathsf{dissolve}(\mathcal{C}, q)$ be as in Definition 5, with q_0, q_1, $\vec{y}$, u, T, and $U_0, \dots, U_{k-1}$ as defined there. Moreover, we write $\oplus$ for addition modulo k, and $\ominus$ for subtraction modulo k. For every $i \in \{0, \dots, k-1\}$, we define X_i as follows:

$$X_i := \mathsf{vars}(\mathsf{C}_q(x_i)).$$

The reduction of Lemma 13 will be described under the following simplifying assumptions which can be made without loss of generality:

- every uncertain database $\mathbf{db}$ that is input to $\mathsf{CERTAINTY}(q)$ is typed, purified, and gpurified relative to q. This assumption is without loss of generality as argued in Section 3, and by Lemmas 1 and 17; and
- for every $i \in \{0, \dots, k-1\}$, no atom of $\mathsf{C}_q(x_i)$ contains constants or double occurrences of the same variable. This assumption is without loss of generality by Lemma 12.

Under these notations and assumptions, we describe the reduction of Lemma 13. Let $\mathbf{db}$ be an uncertain database that is input to $\mathsf{CERTAINTY}(q)$. Define a directed k-partite graph, denoted $\mathcal{G}(\mathbf{db})$, as follows:

1. the vertex set of $\mathcal{G}(\mathbf{db})$ is $\bigcup_{i=0}^{k-1} \mathsf{type}(x_i)$; and
2. there is a directed edge from $a \in \mathsf{type}(x_i)$ to $b \in \mathsf{type}(x_{i\oplus 1})$ if for some valuation θ over $\mathsf{vars}(q)$, we have that $\theta(q) \subseteq \mathbf{db}$ and $\theta(x_i) = a$ and $\theta(x_{i\oplus 1}) = b$. In this case, we say that $\theta[X_i]$ *realizes* the edge (a, b), where $\theta[X_i]$ denotes the restriction of θ on X_i.

Notice that distinct valuations can realize the same edge of $\mathcal{G}(\mathbf{db})$ (but if $\mathbf{db}$ is consistent, then every edge in $\mathcal{G}(\mathbf{db})$ is realized at most once).

Example 13. Let $q = \{R_1(\underline{x_0}, y_1), R_2(\underline{x_0}, y_2), S^c(\underline{y_1, y_2}, x_1)$, $R_3(\underline{x_0}, y_3)$, $V(\underline{x_1}, x_0)\}$. Then we have $x_0 \xrightarrow{\mathsf{M}} x_1$ and $X_0 = \{x_0, y_1, y_2, y_3\}$. Assume an uncertain database $\mathbf{db}$ containing, among others, the following facts.

R_1	$\underline{x_0}$	y_1
	a	c_1

R_2	$\underline{x_0}$	y_2
	a	c_2
	a	c_3

S	$\underline{y_1}$	$\underline{y_2}$	x_1
	c_1	c_2	1
	c_1	c_3	1

R_3	$\underline{x_0}$	y_3
	a	β
	a	γ

The graph $\mathcal{G}(\mathbf{db})$ contains a directed edge $(a, 1)$, which is realized by $\{x_0 \mapsto a, y_1 \mapsto c_1, y_2 \mapsto c_2, y_3 \mapsto \beta\}$. The edge $(a, 1)$ is also realized by $\{x_0 \mapsto a, y_1 \mapsto c_1, y_2 \mapsto c_3, y_3 \mapsto \gamma\}$.

Let $[\![\mathbf{db}]\!]$ be the subset of $\mathbf{db}$ that contains all facts with mode c. Significantly, the edges in $\mathcal{G}(\mathbf{db})$ outgoing from some constant $a \in \mathsf{type}(x_j)$ (for some $j \in \{0, \dots, k-1\}$) are fully determined by $[\![\mathbf{db}]\!]$ and the gblock of $\mathbf{db}$ containing all facts whose relation name is in $\mathsf{C}_q(x_j)$ and whose primary-key position contains the constant a (call this gblock $\mathbf{g}_a$). Since $\mathbf{db}$ is gpurified, for every repair $\mathbf{s}$ of $\mathbf{g}_a$, there exists a unique constant $b \in \mathsf{type}(x_{j\oplus 1})$ such that

$$\mathbf{s} \cup [\![\mathbf{db}]\!] \models (\mathsf{C}_q(x_j) \cup [\![q]\!])_{[x_j, x_{j\oplus 1} \mapsto a, b]},$$

in which case $\mathcal{G}(\mathbf{db})$ will contain a directed edge from a to b. Uniqueness of b follows from $\mathcal{K}(\mathsf{C}_q(x_j) \cup [\![q]\!]) \models x_j \rightarrow x_{j\oplus 1}$ and [18, Lemma 4.3].

Since $\mathbf{db}$ is gpurified, $\mathcal{G}(\mathbf{db})$ is a vertex-disjoint union of strong components such that no edge leads from one strong component to another strong component (i.e., all strong components are initial).[5] In what follows, let D be a strong component of $\mathcal{G}(\mathbf{db})$. Since $\mathcal{G}(\mathbf{db})$ is k-partite, the length of any cycle in $\mathcal{G}(\mathbf{db})$ must be a multiple of k, i.e., must be in $\{k, 2k, 3k, \dots\}$. Let $\mathbf{db}_D$ be the subset of $\mathbf{db}$ that contains $R(\underline{a}, \vec{b})$ whenever R is of mode i and the constant a is a vertex in D (and $\vec{b}$ is any sequence of constants). Obviously, every block of $\mathbf{db}$ is either included in $\mathbf{db}_D$ or disjoint with $\mathbf{db}_D$.

Clearly, D must contain a cycle. Among the cycles in D of length exactly k, we now distinguish the cycles that *support* q from those that do not, as defined next. Let such cycle in D be

$$a_0, a_1, \dots, a_{k-1}, a_0 \tag{2}$$

where for $i \in \{0, \dots, k-1\}$, $a_i \in \mathsf{type}(x_i)$. For $i \in \{0, \dots, k-1\}$, let Δ_i be the set of all valuations over X_i that realize $(a_i, a_{i\oplus 1})$. We say that the cycle (2) *supports* q if for for all $i, j \in \{0, \dots, k-1\}$, for all $\mu_i \in \Delta_i$ and $\mu_j \in \Delta_j$, it is the case that μ_i and μ_j agree on all variables in $X_i \cap X_j$. Notice that $X_i \cap X_j$ can be empty. The cycle (2) may not support q, because μ_i and μ_j can disagree on variables in $X_i \cap X_j \cap \mathsf{vars}(\vec{y})$, as illustrated next.

Example 14. Let $q = \{R(\underline{x_0}, x_1, y),\ S(\underline{x_1}, x_0, y)\}$. We have $x_0 \xrightarrow{\mathsf{M}} x_1 \xrightarrow{\mathsf{M}} x_0$. Let $\mathbf{db}$ be the uncertain database containing the following facts.

R	$\underline{x_0}$	x_1	y
	a	1	α
	a	1	β

S	$\underline{x_1}$	x_0	y
	1	a	α
	1	a	β

The edge set of $\mathcal{G}(\mathbf{db})$ is $\{(a, 1),\ (1, a)\}$. Both $(a, 1)$ and $(1, a)$ are realized by the valuations $\{x_0 \mapsto a,\ x_1 \mapsto 1,\ y \mapsto \alpha\}$ and $\{x_0 \mapsto a,\ x_1 \mapsto 1,\ y \mapsto \beta\}$, which disagree on y. Hence, the cycle $a, 1, a$ does not support q.

On the other hand, we can assume without loss of generality that μ_i and μ_j agree on all variables in $X_i \cap X_j \cap \{x_0, \dots, x_{k-1}\}$. In particular, if $x_i \in X_j$, then $\mu_j(x_i) = \mu_i(x_i) = a_i$. To see why this is the case, assume that $x_i \in X_j$, where $i, j \in \{0, \dots, k-1\}$ and $i \neq j$. Then, it must be that $x_j \xrightarrow{\mathsf{M}} x_i$. Two cases can occur:

- if $j = i\ominus 1$, then μ_j realizes the edge $(a_{i\ominus 1}, a_i)$ and $\mu_j(x_i) = a_i$; and
- if $j \neq i\ominus 1$, then $x_j \xrightarrow{\mathsf{M}} x_i \xrightarrow{\mathsf{M}} x_{i\oplus 1} \cdots \xrightarrow{\mathsf{M}} x_{j\ominus 1} \xrightarrow{\mathsf{M}} x_j$ is a shorter Markov cycle.

The second case can be avoided by picking $\mathcal{C}$ to be the shorter cycle, as illustrated by Example 15. It can be seen that such choice of $\mathcal{C}$ is without loss of generality. In particular, in Lemma 15, if $\mathcal{C}$ was premier, then the shorter cycle will also be premier.

Example 15. Let $q = \{R(\underline{x_0}, x_1),\ S(\underline{x_1}, x_2, x_0),\ V(\underline{x_2}, x_0)\}$. Then, $x_0 \xrightarrow{\mathsf{M}} x_1 \xrightarrow{\mathsf{M}} x_2 \xrightarrow{\mathsf{M}} x_0$. We have $X_0 = \{x_0, x_1\}$, $X_1 = \{x_1, x_2, x_0\}$, and $X_2 = \{x_2, x_0\}$. Assume an uncertain database $\mathbf{db}$ with the following facts.

R	$\underline{x_0}$	x_1
	a	1
	b	1

S	$\underline{x_1}$	x_2	x_0
	1	β	a
	1	β	b

V	$\underline{x_2}$	x_0
	β	a
	β	b

[5]Strong components are defined by Definition 1.

The graph $\mathcal{G}(\mathbf{db})$ contains an elementary directed cycle $a, 1, \beta, a$. The edge $(a, 1)$ is realized by $\mu_0 = \{x_0 \mapsto a,\ x_1 \mapsto 1\}$. The edge $(1, \beta)$ is realized, among others, by $\mu_1 = \{x_1 \mapsto 1, x_2 \mapsto \beta, x_0 \mapsto b\}$. Notice that μ_0 and μ_1 disagree on x_0. Although it is easy to deal with this situation where two valuations disagree on a variable in the Markov cycle, it is even easier to avoid this situation by working with the shorter Markov cycle $x_0 \xrightarrow{\mathsf{M}} x_1 \xrightarrow{\mathsf{M}} x_0$.

We now distinguish two cases.

Case D contains either an elementary directed cycle of size k that does not support q, or an elementary directed cycle of size strictly greater than k. We show in the next paragraph how to construct a repair $\mathbf{s}$ of $\mathbf{db}_D$ such that $\mathbf{s}$ is not grelevant for q in $\mathbf{db}$. Then, by Lemma 16, every repair of $\mathbf{db}$ satisfies q if and only if every repair of $\mathbf{db} \setminus \mathbf{db}_D$ satisfies q. In this case, the reduction deletes from $\mathbf{db}$ all facts of $\mathbf{db}_D$.

The construction of $\mathbf{s}$ proceeds as follows. Pick an elementary cycle in D that has size strictly greater than k, or that has size k but does not support q. The cycle picked will henceforth be denoted by $\mathcal{E}$. Construct a maximal sequence

$$(V_0, E_0), b_1, (V_1, E_1), b_2, (V_2, E_2), \dots, b_n, (V_n, E_n)$$

where

1. V_0 is the set of vertices in $\mathcal{E}$, and E_0 is the set of directed edges in $\mathcal{E}$; and
2. for every $i \in \{1, \dots, n\}$,
   (a) $b_i \notin V_{i-1}$ and for some $c \in V_{i-1}$, (b_i, c) is a directed edge in $\mathcal{G}(\mathbf{db})$; and
   (b) $V_i = V_{i-1} \cup \{b_i\}$ and $E_i = E_{i-1} \cup \{(b_i, c)\}$.

The resulting graph (V_n, E_n) is such that V_n is equal to the vertex set of D, and E_n contains exactly one outgoing edge for each vertex in V_n. The graph (V_n, E_n) contains no directed cycle other than $\mathcal{E}$. To construct $\mathbf{s}$, for each $j \in \{0, \dots, k-1\}$, for each vertex $a \in V_n \cap \mathsf{type}(x_j)$, select some valuation μ that realizes the edge in E_n outgoing from a, and add $\mu(\mathsf{C}_q(x_j))$ to $\mathbf{s}$. If $\mathcal{E}$ has size k, then the valuations μ should be selected such that for some vertices a, b in $\mathcal{E}$, the valuations chosen for a and b disagree on some variable of $\mathsf{vars}(\vec{y})$. It is not hard to see that the set $\mathbf{s}$ so obtained is a repair of $\mathbf{db}_D$ that is not grelevant for q in $\mathbf{db}$.

We illustrate the above construction by two examples.

Example 16. In Example 14, one can choose $\mathbf{s} = \{R(\underline{a}, 1, \alpha), S(\underline{1}, a, \beta)\}$. The treatment of a directed cycle of size strictly greater than k is illustrated by $\mathbf{db}_{03}$ in Example 10.

Example 17. Let $q = \{R(\underline{x_0}, y_1, y_2), V(\underline{x_1}, y_2), S_1^{\mathsf{c}}(\underline{y_1}, \underline{y_2}, x_1), S_2^{\mathsf{c}}(\underline{y_2}, x_0)\}$. We have $x_0 \xrightarrow{\mathsf{M}} x_1 \xrightarrow{\mathsf{M}} x_0$, $X_0 = \{x_0, y_1, y_2\}$, and $X_1 = \{x_1, y_2\}$. Let $\mathbf{db}$ be an uncertain database with the following facts.

R	$\underline{x_0}$	y_1	y_2
	a	1	2
	a	3	4
	a	1	6

V	$\underline{x_1}$	y_2
	γ	2
	γ	4
	β	6

S_1^{c}	$\underline{y_1}$	$\underline{y_2}$	x_1
	1	2	γ
	3	4	γ
	1	6	β

S_2^{c}	$\underline{y_2}$	x_0
	2	a
	4	a
	6	a

The following table lists the edges in $\mathcal{G}(\mathbf{db})$, by type, along with the valuations that realize each edge.

Edges in $\mathsf{type}(x_0) \times \mathsf{type}(x_1)$

Edge	Realized by
(a, γ)	$\{x_0 \mapsto a, y_1 \mapsto 1, y_2 \mapsto 2\}{=}\mu_1$
	$\{x_0 \mapsto a, y_1 \mapsto 3, y_2 \mapsto 4\}{=}\mu_2$
(a, β)	$\{x_0 \mapsto a, y_1 \mapsto 1, y_2 \mapsto 6\}{=}\mu_3$

Edges in $\mathsf{type}(x_1) \times \mathsf{type}(x_0)$

Edge	Realized by
(γ, a)	$\{x_1 \mapsto \gamma, y_2 \mapsto 2\} \ {=}\mu_4$
	$\{x_1 \mapsto \gamma, y_2 \mapsto 4\} \ {=}\mu_5$
(β, a)	$\{x_1 \mapsto \beta, y_2 \mapsto 6\} \ {=}\mu_6$

Then, $\mathcal{G}(\mathbf{db})$ contains two elementary cycles, a, γ, a and a, β, a, both of length 2. The cycle a, β, a supports q. The cycle a, γ, a does not support q, because μ_1 and μ_5 disagree on y_2. Therefore, the edges (a, γ) and (γ, a), along with μ_1 and μ_5, will be used in the construction of a consistent set $\mathbf{s}$ that is not grelevant for q in $\mathbf{db}$. For the remaining vertex β, we add the edge (β, a), which is only realized by μ_6. Then, $\mathbf{s}$ contains the R-fact $R(\underline{a}, 1, 2)$ (because of μ_1), and the V-facts $V(\underline{\gamma}, 4)$ and $V(\underline{\beta}, 6)$ (because of μ_5 and μ_6 respectively). In this example, there is only one repair that contains $\mathbf{s}$, and this repair falsifies q.

Case every elementary directed cycle in D has length k and supports q. In this case, we will encode each cycle of D as a set of T-facts, as follows. Consider any cycle of the form (2) in D, and take the cross product

$$\Delta_0 \times \Delta_2 \times \cdots \times \Delta_{k-1}, \tag{3}$$

which is of polynomial size (in the size of $\mathbf{db}$). Since we are in the case where any cycle of the form (2) supports q, for every tuple $(\mu_0, \mu_1, \ldots, \mu_{k-1})$ in the cross product (3), the set $\mu := \bigcup_{i=0}^{k-1} \mu_i$ is a well defined valuation over $\{x_0, \ldots, x_{k-1}\} \cup \mathsf{vars}(\vec{y})$. In this case, for each such tuple, the reduction adds the following $k + 1$ facts:

$$\begin{array}{l} T(\underline{D}, a_0, \ldots, a_{k-1}, \mu(\vec{y})) \\ U_0^c(\underline{a_0}, D) \\ \quad\vdots \\ U_{k-1}^c(\underline{a_{k-1}}, D) \end{array}$$

in which D is used as a constant. Recall that $a_i = \mu(x_i)$ for $i \in \{0, \ldots, k-1\}$. Notice that if the sequence $\vec{y}$ is empty, then the reduction will add exactly one T-fact for every cycle of the form (2). Otherwise, the reduction may add multiple T-facts for the same cycle, as illustrated next.

Example 18. Let $q = \{R(\underline{x_0}, x_1, y), S(\underline{x_1}, x_0)\}$. Then we have $x_0 \xrightarrow{\mathsf{M}} x_1 \xrightarrow{\mathsf{M}} x_0$ with $X_0 = \{x_0, x_1, y\}$ and $X_1 = \{x_0, x_1\}$. Let $\mathbf{db}$ be the uncertain database containing the following facts.

R	$\underline{x_0}$	x_1	y
	a	1	α
	a	1	β

S	$\underline{x_1}$	x_0
	1	a

The edge set of $\mathcal{G}(\mathbf{db})$ is $\{(a, 1), (1, a)\}$. The edge $(a, 1)$ is realized by both $\{x_0 \mapsto a, x_1 \mapsto 1, y \mapsto \alpha\}$ and $\{x_0 \mapsto a, x_1 \mapsto 1, y \mapsto \beta\}$. The edge $(1, a)$ is realized only by $\{x_0 \mapsto a, x_1 \mapsto 1\}$. The cycle $a, 1, a$ in $\mathcal{G}(\mathbf{db})$ supports q. The reduction will add the following T-facts (for some identifier D):

T	$\underline{u}$	x_0	x_1	y
	D	a	1	α
	D	a	1	β

Example 19. Take the query q of Example 17, with the following uncertain database $\mathbf{db}$.

R	$\underline{x_0}$	y_1	y_2
	a	1	2
	a	1	6
	a	3	6

V	$\underline{x_1}$	y_2
	γ	2
	β	6

S_1^c	$\underline{y_1}$	$\underline{y_2}$	x_1
	1	2	γ
	1	6	β
	3	6	β

S_2^c	$\underline{y_2}$	x_0
	2	a
	6	a

Then, $\mathcal{G}(\mathbf{db})$ contains two elementary cycles, a, γ, a and a, β, a, both of length 2 and both supporting q. The reduction will add the following T-facts (for some identifier D):

T	$\underline{u}$	x_0	x_1	y_1	y_2
	D	a	γ	1	2
	D	a	β	1	6
	D	a	β	3	6

Each relation U_i^c encodes that every constant in the set $\mathsf{type}(x_i) \cap \mathsf{adom}(\mathbf{db})$ occurs in a unique strong component of $\mathcal{G}(\mathbf{db})$. The meaning of the T-facts is as follows. Let $V = \{x_0, \ldots, x_{k-1}\} \cup \mathsf{vars}(\vec{y})$. Let Θ_D be the set of all valuations over V such that

$$T(\underline{D}, \mu(x_1), \ldots, \mu(x_{k-1}), \mu(\vec{y}))$$

has been added by the reduction. Then the following hold (recall $q_0 = \bigcup_{i=0}^{k-1} \mathsf{C}_q(x_i)$):

- for every repair $\mathbf{r}$ of $\mathbf{db}$, there exists $\mu \in \Theta_D$ such that $\mathbf{r} \models \mu(q_0)$; and
- for every $\mu \in \Theta_D$, there exists a repair $\mathbf{r}$ of $\mathbf{db}$ such that
  1. $\mathbf{r} \models \mu(q_0)$; and
  2. for each $\mu' \in \Theta_D$, if $\mu' \neq \mu$, then $\mathbf{r} \not\models \mu'(q_0)$.

The cycles in D can be found in polynomial time by solving reachability problems, as explained in [19, Theorem 4] and [9]. The crux is that the number of cycles in $\mathcal{G}(\mathbf{db})$ of length exactly k is polynomially bounded. Any longer cycle consists of an elementary path $a_0, a_1, \ldots, a_{k-1}, a_0'$ of length k ($a_0 \neq a_0'$), concatenated with an elementary path from a_0' to a_0 that contains no vertex in $\{a_1, \ldots, a_{k-1}\}$. Notice incidentally that the reduction needs to know the existence (or not) of cycles of size strictly greater than k in any strong component D, but the vertices on such cycle need not be remembered.

It can now be seen that, in general, the above reduction results in a database $\mathbf{db}'$ that is as in the following lemma.

LEMMA 18. *Let q and $\mathcal{C}$ be as in the statement of Lemma 13. Let $q^* = \mathsf{dissolve}(q, \mathcal{C})$, and let the variable u be as in Definition 5. Let $\mathbf{db}$ be an uncertain database that is input to* $\mathsf{CERTAINTY}(q)$. *We can compute in polynomial time an uncertain database $\mathbf{db}'$ that is a legal input to* $\mathsf{CERTAINTY}(q^*)$ *such that the following hold:*

1. *for every repair $\mathbf{r}$ of $\mathbf{db}$, there exists a repair $\mathbf{r}'$ of $\mathbf{db}'$ such that for every valuation θ over $\mathsf{vars}(q^*)$, if $\theta(q^*) \subseteq \mathbf{r}'$, then $\theta(q) \subseteq \mathbf{r}$; and*
2. *for every repair $\mathbf{r}'$ of $\mathbf{db}'$, there exists a repair $\mathbf{r}$ of $\mathbf{db}$ such that for every valuation θ over $\mathsf{vars}(q)$, if $\theta(q) \subseteq \mathbf{r}$, then there exists a constant D such that $\theta_{[u \mapsto D]}(q^*) \subseteq \mathbf{r}'$.*

We can now prove Lemma 13.

PROOF OF LEMMA 13. Let $\mathbf{db}$ be an uncertain database that is input to $\mathsf{CERTAINTY}(q)$. By Lemma 18, we can compute in polynomial time an uncertain database $\mathbf{db}'$ that is a legal input to $\mathsf{CERTAINTY}(q^*)$ such that $\mathbf{db}'$ satisfies conditions 1 and 2 in the statement of Lemma 18. It suffices to show that the following are equivalent.

1. Every repair of $\mathbf{db}$ satisfies q.
2. Every repair of $\mathbf{db}'$ satisfies q^*.

$\boxed{1 \Longrightarrow 2}$ Proof by contraposition. Assume a repair $\mathbf{r}'$ of $\mathbf{db}'$ such that $\mathbf{r}' \not\models q^*$. By item 2 in the statement of Lemma 18, we can assume a repair $\mathbf{r}$ of $\mathbf{db}$ such that for every valuation θ over $\mathsf{vars}(q)$, if $\theta(q) \subseteq \mathbf{r}$, then there exists a constant D such that $\theta_{[u \mapsto D]}(q^*) \subseteq \mathbf{r}'$. Obviously, if $\mathbf{r} \models q$, then $\mathbf{r}' \models q^*$, a contradiction. We conclude by contradiction that $\mathbf{r} \not\models q$. $\boxed{2 \Longrightarrow 1}$ Proof by contraposition. Assume a repair $\mathbf{r}$ of $\mathbf{db}$ such that $\mathbf{r} \not\models q$. By item 1 in the statement of Lemma 18, we can assume a repair $\mathbf{r}'$ of $\mathbf{db}'$ such that for every valuation θ over $\mathsf{vars}(q^*)$, if $\theta(q^*) \subseteq \mathbf{r}'$, then $\theta(q) \subseteq \mathbf{r}$. Obviously, $\mathbf{r}' \not\models q^*$. $\square$

7. CONCLUSION

This paper settles a long-standing open question in consistent query answering, by providing a solution to the complexity classification task of $\mathsf{CERTAINTY}(q)$ for q ranging over the class of self-join-free Boolean conjunctive queries. In particular, we showed that, given q, there exists a procedure that looks at the structure of the attack graph of q and decides whether $\mathsf{CERTAINTY}(q)$ is in $\mathbf{FO}$, in $\mathbf{P} \setminus \mathbf{FO}$, or $\mathbf{coNP}$-complete.

The exciting question that still remains open is whether our results can be extended beyond self-join-free conjunctive queries, to conjunctive queries with self-joins and unions of conjunctive queries.

Acknowledgments

This work is supported in part by the NSF through NSF grant NSF IIS-1115188.

8. REFERENCES

[1] M. Arenas, L. E. Bertossi, and J. Chomicki. Consistent query answers in inconsistent databases. In *PODS*, pages 68–79. ACM Press, 1999.

[2] B. Aspvall, M. F. Plass, and R. E. Tarjan. A linear-time algorithm for testing the truth of certain quantified boolean formulas. *Inf. Process. Lett.*, 8(3):121–123, 1979.

[3] C. Beeri, R. Fagin, D. Maier, and M. Yannakakis. On the desirability of acyclic database schemes. *J. ACM*, 30(3):479–513, 1983.

[4] L. E. Bertossi. *Database Repairing and Consistent Query Answering*. Synthesis Lectures on Data Management. Morgan & Claypool Publishers, 2011.

[5] A. A. Bulatov. Complexity of conservative constraint satisfaction problems. *ACM Trans. Comput. Log.*, 12(4):24, 2011.

[6] G. Fontaine. Why is it hard to obtain a dichotomy for consistent query answering? In *LICS*, pages 550–559. IEEE Computer Society, 2013.

[7] A. Fuxman and R. J. Miller. First-order query rewriting for inconsistent databases. In T. Eiter and L. Libkin, editors, *ICDT*, volume 3363 of *Lecture Notes in Computer Science*, pages 337–351. Springer, 2005.

[8] P. G. Kolaitis and E. Pema. A dichotomy in the complexity of consistent query answering for queries with two atoms. *Inf. Process. Lett.*, 112(3):77–85, 2012.

[9] P. Koutris and D. Suciu. A dichotomy on the complexity of consistent query answering for atoms with simple keys. In Schweikardt et al. [15], pages 165–176.

[10] P. Koutris and J. Wijsen. A trichotomy in the data complexity of certain query answering for conjunctive queries. *CoRR*, abs/1501.07864, 2015.

[11] L. Libkin. *Elements of Finite Model Theory*. Springer, 2004.

[12] D. Maslowski and J. Wijsen. A dichotomy in the complexity of counting database repairs. *J. Comput. Syst. Sci.*, 79(6):958–983, 2013.

[13] D. Maslowski and J. Wijsen. Counting database repairs that satisfy conjunctive queries with self-joins. In Schweikardt et al. [15], pages 155–164.

[14] G. J. Minty. On maximal independent sets of vertices in claw-free graphs. *J. Comb. Theory, Ser. B*, 28(3):284–304, 1980.

[15] N. Schweikardt, V. Christophides, and V. Leroy, editors. *Proc. 17th International Conference on Database Theory (ICDT), Athens, Greece, March 24-28, 2014*. OpenProceedings.org, 2014.

[16] J. Wijsen. On the first-order expressibility of computing certain answers to conjunctive queries over uncertain databases. In J. Paredaens and D. V. Gucht, editors, *PODS*, pages 179–190. ACM, 2010.

[17] J. Wijsen. A remark on the complexity of consistent conjunctive query answering under primary key violations. *Inf. Process. Lett.*, 110(21):950–955, 2010.

[18] J. Wijsen. Certain conjunctive query answering in first-order logic. *ACM Trans. Database Syst.*, 37(2):9, 2012.

[19] J. Wijsen. Charting the tractability frontier of certain conjunctive query answering. In R. Hull and W. Fan, editors, *PODS*, pages 189–200. ACM, 2013.

[20] J. Wijsen. A survey of the data complexity of consistent query answering under key constraints. In C. Beierle and C. Meghini, editors, *Foundations of Information and Knowledge Systems - 8th International Symposium, FoIKS 2014, Bordeaux, France, March 3-7, 2014. Proceedings*, volume 8367 of *Lecture Notes in Computer Science*, pages 62–78. Springer, 2014.

APPENDIX

A. PROOF OF LEMMA 9

We first show two helping lemmas.

LEMMA 19. *Let q be a self-join-free Boolean conjunctive query. Let $X \subseteq \mathsf{vars}(q)$ and let $G \in q$ be an R-atom such for every $x \in X$, $G \overset{q}{\not\rightsquigarrow} x$. Let $\mathbf{r}$ be a repair of some database such that $\mathbf{r} \models q$. Let $A \in \mathbf{r}$ be an R-fact that is relevant for q in $\mathbf{r}$. Let B be key-equal to A and $\mathbf{r}_B = (\mathbf{r} \setminus \{A\}) \cup \{B\}$. Then, for every valuation ζ over X, if $\mathbf{r}_B \models \zeta(q)$, then $\mathbf{r} \models \zeta(q)$.*

PROOF. Let ζ be a valuation over X such that $\mathbf{r}_B \models \zeta(q)$. We can assume a valuation ζ^+ over $\mathsf{vars}(q)$ such that $\zeta^+[X] = \zeta[X]$ and $\zeta^+(q) \subseteq \mathbf{r}_B$. Thus, ζ^+ extends ζ to $\mathsf{vars}(q)$. We need to show $\mathbf{r} \models \zeta(q)$, which is obvious if $B \notin \zeta^+(q)$. Assume next $B \in \zeta^+(q)$. Since A is relevant for q in $\mathbf{r}$, we can assume a valuation μ over $\mathsf{vars}(q)$ such that $A \in \mu(q) \subseteq \mathbf{r}$. Let $q' = q \setminus \{G\}$. Let $\mathbf{r}' = \mathbf{r}_B \setminus \{B\} = \mathbf{r} \setminus \{A\}$. Since q' contains no R-atom (no self-join), $\zeta^+(q') \subseteq \mathbf{r}'$ and $\mu(q') \subseteq \mathbf{r}'$. Moreover, $\zeta^+[\mathsf{key}(G)] = \mu[\mathsf{key}(G)]$, because A and B are key-equal.

From $\mathcal{K}(q') \models \mathsf{key}(G) \rightarrow G^{+,q}$ and [18, Lemma 4.3], it follows $\zeta^+[G^{+,q}] = \mu[G^{+,q}]$.

Let τ be the complete edge-labeled undirected graph whose vertices are the atoms of q; an edge between H and H' is labeled by $\mathsf{vars}(H) \cap \mathsf{vars}(H')$.

Let τ' be the graph obtained from τ by cutting every edge whose label is included in $G^{+,q}$. Let q_G be the subset of q containing all atoms that are in τ''s strong component that contains G. Let $q_X = q \setminus q_G$.

Let κ be the valuation over $\mathsf{vars}(q)$ such that for every $x \in \mathsf{vars}(q)$,

$$\kappa(x) = \begin{cases} \mu(x) & \text{if } x \in \mathsf{vars}(q_G) \\ \zeta^+(x) & \text{if } x \in \mathsf{vars}(q_X) \end{cases}$$

We show that κ is well defined. Assume $x \in \mathsf{vars}(q_X) \cap \mathsf{vars}(q_G)$. Then, there exist atoms $F' \in q_X$ and $G' \in q_G$ such that $x \in \mathsf{vars}(F') \cap \mathsf{vars}(G')$. Since F' and G' belong to distinct strong components of τ', it follows $\mathsf{vars}(F') \cap \mathsf{vars}(G') \subseteq G^{+,q}$. Consequently, $x \in G^{+,q}$. Since $\zeta^+[G^{+,q}] = \mu[G^{+,q}]$, it follows that $\mu(x) = \zeta^+(x)$.

Obviously, $\kappa(q) \subseteq \mathbf{r}$. Finally, we show that for every $u \in X$, $\kappa(u) = \zeta(u)$. This is obvious if $u \in X \cap G^{+,q}$. Assume next that $u \in X \setminus G^{+,q}$. Since $G \overset{q}{\not\rightsquigarrow} u$ by the assumption in the statement of Lemma 19, it must be the case $u \in \mathsf{vars}(q_X)$, hence $\kappa(u) = \zeta^+(u) = \zeta(u)$. It follows $\mathbf{r} \models \zeta(q)$. This concludes the proof. □

The following helping lemma extends [18, Lemma B.1].

LEMMA 20. *Let q be a self-join-free Boolean conjunctive query. Let $F \in q$ such that F has zero indegree in the attack graph of q. Let $\mathbf{r}$ be a repair of some database. Let $A \in \mathbf{r}$ such that A is relevant for q in $\mathbf{r}$.*[6] *Let B be key-equal to A and $\mathbf{r}_B = (\mathbf{r} \setminus \{A\}) \cup \{B\}$. Then, for every valuation ζ over $\mathsf{key}(F)$, if $\mathbf{r}_B \models \zeta(q)$, then $\mathbf{r} \models \zeta(q)$.*

PROOF. The proof is obvious if A has the same relation name as F. Assume next that relation names in A and F are distinct. We can assume some atom $G \in q \setminus \{F\}$ such that A has the same relation name as G. Since $G \overset{q}{\not\rightsquigarrow} F$, we have that for each $x \in \mathsf{key}(F)$, $G \overset{q}{\not\rightsquigarrow} x$. The desired result then follows by Lemma 19. □

Assume that a query q contains an R-atom that has no incoming attack in the attack graph of q. Paraphrasing Lemma 20, if one replaces, in a repair $\mathbf{r}$, some relevant fact A with another fact B that belongs to the same block as A, then every R-fact of $\mathbf{r}$ that was not relevant in $\mathbf{r}$, will remain non-relevant in $(\mathbf{r} \setminus \{A\}) \cup \{B\}$. Notice, however, that the fact B may be non-relevant in the new repair $(\mathbf{r} \setminus \{A\}) \cup \{B\}$. The proof of Lemma 9 can now be given.

PROOF OF LEMMA 9. Let $X = \mathsf{key}(F)$. Let $\mathbf{db}$ be an uncertain database. Let $\mathbf{r}$ be a repair of $\mathbf{db}$ that is $\preceq_q^X$-frugal. Let $\mathbf{s}$ be any repair of $\mathbf{db}$. Construct a maximal sequence

$$(\mathbf{r}_0, \mathbf{s}_0), (\mathbf{r}_1, \mathbf{s}_1), \dots, (\mathbf{r}_n, \mathbf{s}_n) \tag{4}$$

where

1. $\mathbf{r}_0 = \mathbf{r}$ and $\mathbf{s}_0 = \mathbf{s}$;
2. for every $i \in \{1, \dots, n\}$, one of the following holds:
   (a) $\mathbf{r}_i = \mathbf{r}_{i-1}$ and $\mathbf{s}_i = (\mathbf{s}_{i-1} \setminus \{A\}) \cup \{B\}$ for distinct, key-equal facts A, B such that $A \in \mathbf{s}_{i-1}$, $B \in \mathbf{r}_{i-1}$, and A is relevant for q in $\mathbf{s}_{i-1}$; or
   (b) $\mathbf{s}_i = \mathbf{s}_{i-1}$ and $\mathbf{r}_i = (\mathbf{r}_{i-1} \setminus \{A\}) \cup \{B\}$ for distinct, key-equal facts A, B such that $A \in \mathbf{r}_{i-1}$, $B \in \mathbf{s}_{i-1}$, and A is relevant for q in $\mathbf{r}_{i-1}$.

That is, the construction repeatedly replaces a fact that is relevant in one repair with its distinct, key-equal fact in the other repair. The sequence (4) is finite, since the total number of distinct relevant facts distinguishes at each step. For the last element $(\mathbf{r}_n, \mathbf{s}_n)$, it holds that the set of facts that are relevant for q in $\mathbf{r}_n$ is equal the set of facts that are relevant for q in $\mathbf{s}_n$. It follows that for every valuation θ over X,

$$\mathbf{r}_n \models \theta(q) \iff \mathbf{s}_n \models \theta(q). \tag{5}$$

By Lemma 20, for every valuation θ over X,

$$\mathbf{r}_n \models \theta(q) \implies \mathbf{r} \models \theta(q), \tag{6}$$
$$\mathbf{s}_n \models \theta(q) \implies \mathbf{s} \models \theta(q). \tag{7}$$

From (6) and since $\mathbf{r}$ is $\preceq_q^X$-frugal, it follows that for every valuation θ over X,

$$\mathbf{r}_n \models \theta(q) \iff \mathbf{r} \models \theta(q). \tag{8}$$

From (8), (5), and (7), it follows that for every valuation θ over X,

$$\mathbf{r} \models \theta(q) \implies \mathbf{s} \models \theta(q).$$

Since $\mathbf{s}$ is an arbitrary repair, the desired result follows. □

[6]Recall from Section 3 that $A \in \mathbf{r}$ is *relevant* for q in $\mathbf{r}$ if $A \in \theta(q) \subseteq \mathbf{r}$ for some valuation θ over $\mathsf{vars}(q)$.