Counting Database Repairs that Satisfy Conjunctive Queries with Self-Joins

Dany Maslowski
University of Mons (UMONS)
dany.maslowski@umons.ac.be

Jef Wijsen
University of Mons (UMONS)
jef.wijsen@umons.ac.be

ABSTRACT

An uncertain database is defined as a relational database in which primary keys need not be satisfied. A block is a maximal subset of tuples of the same relation that agree on the primary key. A repair (or possible world) of an uncertain database is obtained by selecting exactly one tuple from each block. From a probabilistic database perspective, an uncertain database is a restricted kind of block-independent disjoint (BID) probabilistic database, where the restriction is that the probabilities of tuples in a block are equal and sum up to one.

For every fixed Boolean query q, the counting problem $\natural\mathsf{CERTAINTY}(q)$ takes as input an uncertain database **db** and asks to determine the number of repairs that satisfy q. A Boolean conjunctive query is self-join-free if no relation name occurs more than once in it. In previous work, it was proved that for every self-join-free Boolean conjunctive query q, the problem $\natural\mathsf{CERTAINTY}(q)$ is either in **FP** or $\natural$**P**-complete, and it is decidable which of the two cases applies. This complexity dichotomy has its analogue in BID probabilistic databases.

The current paper investigates the complexity of the problem $\natural\mathsf{CERTAINTY}(q)$ for Boolean conjunctive queries with self-joins. Our most appealing result is that for every Boolean conjunctive query q (possibly with self-joins) in which all primary keys consist of a single attribute, $\natural\mathsf{CERTAINTY}(q)$ is either in **FP** or $\natural$**P**-complete, and it is decidable which of the two cases applies. Significantly, no analogous dichotomy for conjunctive queries with self-joins is known for BID probabilistic databases.

Categories and Subject Descriptors

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

General Terms

Theory, Algorithms

Keywords

Conjunctive queries; consistent query answering; database repairing; primary keys; probabilistic databases

1. INTRODUCTION

Primary key violations are a natural way for modeling uncertainty in the relational data model. Tuples of the same relation with the same primary key value are mutually exclusive alternatives for each other. This representation of uncertainty is also used in probabilistic databases, where each tuple is associated with a probability and distinct tuples with the same primary key value are disjoint probabilistic events [12, p. 35].

In this paper, the term *uncertain database* is used for databases with primary key constraints that need not be satisfied. A *repair* (or possible world) of an uncertain database **db** is a maximal subset of **db** that satisfies all primary key constraints. Semantics of querying follows the conventional paradigm of *consistent query answering* [2, 3]: Given a Boolean query q, the decision problem $\mathsf{CERTAINTY}(q)$ takes as input an uncertain database **db** and asks whether q is satisfied by every repair of **db**. Unless specified otherwise, whenever we say "query" in the remainder of this paper, we mean "Boolean query."

The counting variant of $\mathsf{CERTAINTY}(q)$, which has been denoted $\natural\mathsf{CERTAINTY}(q)$, takes as input an uncertain database **db** and asks to determine the number of repairs of **db** that satisfy query q. Maslowski and Wijsen [11] have recently proved that for every self-join-free conjunctive query q, the counting problem $\natural\mathsf{CERTAINTY}(q)$ is either in **FP** or $\natural$**P**-complete, and it is decidable which of the two cases applies. A conjunctive query is self-join-free if no relation name occurs more than once in it. The aim of the current paper is to investigate the complexity of $\natural\mathsf{CERTAINTY}(q)$ for conjunctive queries q with self-joins. Our most appealing result is that the aforementioned complexity dichotomy carries over to self-joins under the condition that all primary keys are simple. A primary key is simple if it consists of a single attribute.

Example 1. The primary key **conf** is underlined in the conference database of Fig. 1. Maximal sets of tuples that agree on their primary key, called *blocks*, are separated by dashed lines. There is uncertainty about the frequency of

10.5441/002/icdt.2014.18

R	**conf**	**rank**	**frequency**
	ICDT	A	biennial
	ICDT	A	annual
	KDD	A	annual
	KDD	B	annual

Figure 1: Uncertain database.

R	$\underline{A}$	B
	1	b
	1	c

S	$\underline{A}$	B
	2	b
	2	c

Figure 2: Uncertain database with four repairs.

ICDT[1], and about the rank of KDD. The database has four repairs. The query

$$\exists y \exists z_1 \exists z_2 (\mathbf{R}(\underline{\text{‘ICDT’}}, y, z_1) \land \mathbf{R}(\underline{\text{‘KDD’}}, y, z_2))$$

(Do ICDT and KDD have equal ranks?) is true in only two repairs.

Moving from self-join-free conjunctive queries to conjunctive queries with self-joins is a major challenge. To get a flavor of an additional hurdle incurred by self-joins, consider the following conjunctive queries, where b and c are distinct constants:

$$\begin{aligned} q_1 &= \exists x R(\underline{x}, b) \land \exists y S(\underline{y}, c) \\ q_2 &= \exists x R(\underline{x}, b) \land \exists y R(\underline{y}, c) \end{aligned}$$

The query q_1 is self-join-free, while q_2 contains a self-join. These queries are composed of the three conjuncts $\exists x R(\underline{x}, b)$, $\exists y S(\underline{y}, c)$, and $\exists y R(\underline{y}, c)$. For the uncertain database of Fig. 2, it is straightforward to verify that each conjunct evaluates to true on 2 repairs (out of a total of 4 repairs). That is, the fraction of repairs satisfying each individual conjunct is $\frac{2}{4} = \frac{1}{2}$. Since the two conjuncts of q_1 refer to distinct relations, their truth values are independent of one another. Therefore, it is correct to conclude that the fraction of repairs satisfying q_1 is $\frac{1}{2} \cdot \frac{1}{2} = \frac{1}{4}$. On the other hand, the two conjuncts of q_2 are not independent, because for the database of Fig. 2, any repair that satisfies $\exists x R(\underline{x}, b)$ will falsify $\exists y R(\underline{y}, c)$, and vice versa.

For a query q, the problem $\natural\mathsf{CERTAINTY}(q)$ is a special case of probabilistic query answering. Let N be the total number of repairs of a given uncertain database $\mathbf{db}$. If a fact A (or, by extension, a Boolean query) evaluates to true in m repairs, then its probability, denoted $P(A)$, is m/N. For example, in Fig. 1, the probability of the fact $\mathbf{R}(\underline{\text{ICDT}}, \text{A}, \text{biennial})$ is 2/4, because it belongs to 2 repairs out of 4. It can now be easily verified that for all distinct facts A, B of $\mathbf{db}$, the following hold:

- If the facts A and B belong to a same block, then $P(A \land B) = 0$. In probabilistic terms, distinct facts of the same block represent *disjoint* (i.e., exclusive) events.
- If the facts A and B belong to distinct blocks, then $P(A \land B) = P(A) \cdot P(B)$. In probabilistic terms, facts of distinct blocks are *independent*.

[1]Until 2009, ICDT was actually organized every two years.

R	**conf**	**rank**	**frequency**	P
	ICDT	A	biennial	0.3
	ICDT	A	annual	0.6
	KDD	A	annual	0.5
	KDD	B	annual	0.5

Figure 3: Representation of a BID probabilistic database.

Probabilistic databases satisfying the above two properties have been coined *block-independent-disjoint* (BID) by Dalvi, Ré, and Suciu [4]. BID probabilistic databases can be represented by listing the probability of each fact, as illustrated in Fig. 3. The main differences between uncertain databases and BID probabilistic databases are twofold:

- In an uncertain database, all facts of a same block have the same probability. In BID probabilistic databases, facts of a same block need not have the same probability. For example, in the BID probabilistic database of Fig. 3, the two facts about ICDT have distinct probabilities (0.3 and 0.6).
- In an uncertain database, the probabilities of facts in a same block sum up to 1. In BID probabilistic databases, this sum can be strictly less than 1.

A detailed comparison of both data models can be found in [14].

The tractability/intractability frontier of evaluating self-join-free conjunctive queries on BID probabilistic databases has been revealed by Dalvi et al. [5]. As mentioned before, on uncertain databases, this frontier was recently charted by Maslowski and Wijsen [11]. In both these works, conjunctive queries are self-join-free. Self-joins in BID databases have not been systematically studied in depth [12, p. 88].

The situation is different for tuple-independent probabilistic databases. In such a database, there is no notion of block and all tuples represent independent events. The tractability/intractability frontier of evaluating unions of conjunctive queries (possibly with self-joins) on tuple-independent probabilistic databases has been revealed by Dalvi and Suciu [6].

The remainder of this paper is organized as follows. The next section further discusses related work. Section 3 introduces basic notions. Section 4 recalls a dichotomy in the complexity of $\natural\mathsf{CERTAINTY}(q)$ for self-join-free conjunctive queries. In Section 5, we start our investigation on the complexity of $\natural\mathsf{CERTAINTY}(q)$ when q is a conjunctive query with a single relation name, denoted R. Such queries are not self-join-free, unless they consist of a single conjunct. We provide two lemmas that can be used for showing (in)tractability of $\natural\mathsf{CERTAINTY}(q)$. Section 6 explains that the assumption of a single relation is without loss of generality, as multiple relations can be easily encoded into a single one. In Section 7, we show our most appealing result: if q is a conjunctive query in which all primary keys are simple, then $\natural\mathsf{CERTAINTY}(q)$ is either in $\mathbf{FP}$ or $\natural\mathbf{P}$-complete, and it is decidable which of the two cases applies. Section 8 concludes the paper and raises challenges for future research.

2. MORE RELATED WORK

The investigation of $\mathsf{CERTAINTY}(q)$ was pioneered by Fuxman and Miller [7, 8], who defined a class of self-join-free conjunctive queries q for which $\mathsf{CERTAINTY}(q)$ is first-order expressible. Since then, the following complexity classification problem has gained considerable research interest: Given a conjunctive query q on input, determine the complexity classes to which the problem $\mathsf{CERTAINTY}(q)$ belongs, or does not belong. For conjunctive queries with self-joins, this complexity classification problem remains largely open, which is likely due to the difficulty of treating self-joins. For self-join-free conjunctive queries, the following are known:

- Given an acyclic self-join-free conjunctive query q, it is decidable whether or not $\mathsf{CERTAINTY}(q)$ is first-order expressible [13].
- For each self-join-free conjunctive query q with exactly two atoms, $\mathsf{CERTAINTY}(q)$ is either in **P** or **coNP**-complete, and it is decidable which of the two cases applies [9]. The sufficient condition for **coNP**-completeness has later on been generalized to more than two atoms [14].
- For each self-join-free conjunctive query q in which all primary keys are either simple or contain all attributes of the relation, $\mathsf{CERTAINTY}(q)$ is either in **P** or **coNP**-complete, and it is decidable which of the two cases applies [10].

It remains an intriguing open conjecture that for each self-join-free conjunctive query q, $\mathsf{CERTAINTY}(q)$ is either in **P** or **coNP**-complete.

As explained in Section 1, the counting version of the decision problem $\mathsf{CERTAINTY}(q)$, denoted $\natural\mathsf{CERTAINTY}(q)$, is intimately related to the problem of evaluating conjunctive queries on block-independent-disjoint (BID) probabilistic databases. It was proved that for every self-join-free conjunctive query q, $\natural\mathsf{CERTAINTY}(q)$ is either in **FP** or $\natural$**P**-complete [11], using a proof inspired by an analogous complexity dichotomy in BID probabilistic databases [5]. No such complexity dichotomy is known for conjunctive queries with self-joins.

For unions of conjunctive queries with self-joins, a complexity dichotomy is known for tuple-independent probabilistic databases [6]. The latter dichotomy is fundamentally different from the dichotomy proved in Theorem 3 of this paper, because tuple-independent probabilistic databases have no notion of block.

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 $\vec{x}$.

Let U be a set of variables. A *valuation over* U is a total mapping θ from U to the set of constants. Such valuation θ is extended to be the identity on constants and on variables not in U.

Atoms and key-equal facts. Each *relation name* R of arity n, $n \geq 1$, has a unique *primary key* which is a set $\{1, 2, \dots, k\}$ where $1 \leq k \leq n$. We say that R has *signature* $[n, k]$ if R has arity n and primary key $\{1, 2, \dots, k\}$. The relation name R is *simple-key* if $k = 1$. The relation name R is *all-key* if $n = k$. Elements of the primary key are called *primary-key positions*, while $k+1$, $k+2$, $\dots$, 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 we call $R(s_1, \dots, s_n)$ an *R-atom* (or simply atom), where each s_i is either a constant or a variable ($1 \leq i \leq n$). Such atom is commonly written as $R(\underline{\vec{x}}, \vec{y})$ where the primary key value $\vec{x} = s_1, \dots, s_k$ is underlined and $\vec{y} = s_{k+1}, \dots, s_n$. A *fact* is an 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 $\vec{a}_1 = \vec{a}_2$.

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 database, 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 **db** of facts using only the relation names of the schema. We write **adom**(**db**) for the active domain of **db** (i.e., the set of constants that occur in **db**). A *block* of **db** is a maximal set of key-equal facts of **db**. An uncertain database **db** is *consistent* if it does not contain two distinct facts that are key-equal (i.e., if every block of **db** is a singleton). A *repair* of **db** is a maximal (with respect to set containment) consistent subset of **db**.

Boolean conjunctive query. 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. By $\mathsf{vars}(q)$, we denote the set of variables that occur in q. The set q represents the first-order sentence

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

where $\{u_1, \dots, u_k\} = \mathsf{vars}(q)$. The query q is *satisfied* by uncertain database **db**, denoted $\mathbf{db} \models q$, if there exists a valuation θ over $\mathsf{vars}(q)$ such that for each $i \in \{1, \dots, n\}$, $R_i(\underline{\theta(\vec{x}_i)}, \theta(\vec{y}_i)) \in \mathbf{db}$. We say that q has a *self-join* if some relation name occurs more than once in q (i.e., if $R_i = R_j$ for some $1 \leq i < j \leq n$). If q has no self-join, then it is called *self-join-free*. We say that q is *unirelational* if it is empty or refers to only one relation name (i.e., if $R_i = R_j$ for all i, j).

Since every relation name has a fixed signature, relevant primary key constraints are implicitly present in all queries; moreover, primary keys will be underlined.

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 $1 \leq i \leq \ell$.

Complex part of a Boolean conjunctive query. The following definition is borrowed from [11]. Let q be a Boolean conjunctive query. A variable $x \in \mathsf{vars}(q)$ is called a *liaison variable* if x has at least two occurrences in q.[2] The *complex part* of a Boolean conjunctive query q, denoted $[\![q]\!]$, contains every atom $F \in q$ such that some non-primary-key position in F contains a liaison variable or a constant.

[2]Liaison variables are sometimes called "join variables" in the literature. Notice nevertheless that in the singleton query $\{R(\underline{x}, x)\}$, which is not a genuine join, the variable x is a liaison variable.

Example 2. The variable y is the only liaison variable in $q = \{R(\underline{x}, y), R(\underline{y}, z), S(\underline{y}, u, a)\}$, in which a is a constant. The complex part of q is $[\![q]\!] = \{R(\underline{x}, y), S(\underline{y}, u, a)\}$. The complex part of $\{R(\underline{y}, w), R(\underline{x}, u), T(\underline{x, y})\}$, where T is all-key, is empty.

If some atom $F = R(\underline{\vec{x}}, y_1, \dots, y_\ell)$ of a Boolean conjunctive query q does *not* belong to q's complex part, then $y_1, \dots, y_\ell$ are distinct variables that have only one occurrence in q. Intuitively, such variables can be disregarded when evaluating the query q, because they do not impose any join condition. This intuition underlies the following helping lemma.

LEMMA 1. *Let q be a Boolean conjunctive query. Let* **db** *be an uncertain database. Let* $\mathbf{r}_1$, $\mathbf{r}_2$ *be two repairs of* **db**. *For every valuation θ over* $\mathsf{vars}([\![q]\!])$, *if* $\mathbf{r}_1 \models \theta(q)$ *and* $\theta([\![q]\!]) \subseteq \mathbf{r}_2$, *then* $\mathbf{r}_2 \models \theta(q)$.

PROOF. Let θ be a valuation over $\mathsf{vars}([\![q]\!])$ such that $\mathbf{r}_1 \models \theta(q)$ and $\theta([\![q]\!]) \subseteq \mathbf{r}_2$. Since $\mathbf{r}_1 \models \theta(q)$, we can extend θ to a valuation μ_1 over $\mathsf{vars}(q)$ such that $\mu_1(q) \subseteq \mathbf{r}_1$. Every atom $F \in q \setminus [\![q]\!]$ is of the form $R(\underline{\vec{x}}, y_1, \dots, y_\ell)$ where each y_i is a variable that occurs only once in q. Let μ_2 be the extension of θ such that for every atom $F \in q \setminus [\![q]\!]$, we have that $\mu_2(F)$ is the fact of $\mathbf{r}_2$ that is key-equal to $\mu_1(F)$. Obviously, μ_2 is well defined and $\mu_2(q) \subseteq \mathbf{r}_2$. It follows $\mathbf{r}_2 \models \theta(q)$. $\square$

Counting repairs. For any fixed Boolean conjunctive query q, we define $\natural\mathsf{CERTAINTY}(q)$ as the following counting problem: Given an uncertain database **db** on input, determine the number of repairs of **db** that satisfy q.

Let **db** be an uncertain database and q a Boolean query. We write $\mathsf{rset}(\mathbf{db})$ for the set of repairs of **db**, and $\mathsf{rset}(\mathbf{db}, q)$ for the subset of $\mathsf{rset}(\mathbf{db})$ containing each repair that satisfies q. The cardinality of these sets are denoted by $\natural\mathsf{rset}(\mathbf{db})$ and $\natural\mathsf{rset}(\mathbf{db}, q)$ respectively. Thus, for a fixed Boolean conjunctive query q, $\natural\mathsf{CERTAINTY}(q)$ is the problem that takes as input an uncertain database **db** and asks to determine $\natural\mathsf{rset}(\mathbf{db}, q)$. The following is straightforward.

THEOREM 1. *For every Boolean conjunctive query q, the counting problem* $\natural\mathsf{CERTAINTY}(q)$ *is in* $\natural\mathbf{P}$.

PROOF. [3] For any fixed Boolean conjunctive query q, the following problem is in **NP**: Given an uncertain database **db** on input, determine whether some repair of **db** satisfies q. The problem is in **NP**, because if the answer is "yes" for some uncertain database **db**, then a succinct certificate is a repair of **db** that satisfies q. Since the above problem is in **NP**, its counting variant is in $\natural\mathbf{P}$. $\square$

In the technical treatment, it is often more convenient to work with the fraction of repairs rather than the absolute number of repairs that satisfy some query. To this extent, we define $\mathsf{rfrac}(\mathbf{db}, q) = \frac{\natural\mathsf{rset}(\mathbf{db},q)}{\natural\mathsf{rset}(\mathbf{db})}$, the fraction of repairs satisfying q. Since $\natural\mathsf{rset}(\mathbf{db})$ can be computed in polynomial time in the size of **db**, the problems of determining $\natural\mathsf{rset}(\mathbf{db}, q)$ and $\mathsf{rfrac}(\mathbf{db}, q)$ are polynomially equivalent.

[3]The proof suggests that $\natural\mathsf{CERTAINTY}(q)$ might better have been named $\natural\mathsf{POSSIBILITY}(q)$.

4. DICHOTOMY FOR SELF-JOIN-FREE CONJUNCTIVE QUERIES

In earlier work [11], we showed that for every self-join-free Boolean conjunctive query q, $\natural\mathsf{CERTAINTY}(q)$ is either in **FP** or $\natural\mathbf{P}$-hard, and it is decidable which of the two cases applies. This result is recalled next and will be used later on.

```
Function IsSafe(q) Determine whether q is safe
Input: q is a self-join-free Boolean conjunctive query.
Result: Boolean in {true, false}.
begin
    SE0a: if $|q| = 1$ and $\mathsf{vars}(q) = \emptyset$ then
        return true;
    SE0b: if $[\![q]\!] = \emptyset$ then
        return true;
    SE1: if $q = q_1 \cup q_2$ with $q_1 \neq \emptyset \neq q_2$, $\mathsf{vars}(q_1) \cap \mathsf{vars}(q_2) = \emptyset$
    then
        return $IsSafe(q_1) \wedge IsSafe(q_2)$;
    /* $a$ is an arbitrary constant                                    */
    SE2: if $[\![q]\!] \neq \emptyset$ and $\bigcap_{F \in [\![q]\!]} \mathsf{key}(F) \neq \emptyset$ then
        select $x \in \bigcap_{F \in [\![q]\!]} \mathsf{key}(F)$;
        return $IsSafe(q_{[x \mapsto a]})$;
    SE3: if there exists $F \in q$ such that $\mathsf{key}(F) = \emptyset \neq \mathsf{vars}(F)$
    then
        select $F \in q$ such that $\mathsf{key}(F) = \emptyset \neq \mathsf{vars}(F)$;
        select $x \in \mathsf{vars}(F)$;
        return $IsSafe(q_{[x \mapsto a]})$;
    if none of the above then
        return false;
```

Function IsSafe takes a self-join-free conjunctive query q on input, and always terminates with either **true** or **false**. The function is recursive. The base rules (SE0a and SE0b) apply if q consists of a single fact, or if the complex part of q is empty. The recursive rule SE1 applies if q can be partitioned into two subqueries which have no variables in common. The recursive rule SE2 applies if all atoms in the complex part of q contain the same variable at some of their primary-key positions. The recursive rule SE3 applies if all primary-key positions of some atom are occupied by constants and some non-primary-key position contains a variable.

Definition 1. A self-join-free Boolean conjunctive query q is called *safe* if Function IsSafe returns **true** on input q; otherwise q is *unsafe*.

The main result of [11] can now be stated.

THEOREM 2 ([11]). *Let q be a self-join-free Boolean conjunctive query.*

1. *If q is safe, then* $\natural\mathsf{CERTAINTY}(q)$ *is in* **FP**.
2. *If q is unsafe, then* $\natural\mathsf{CERTAINTY}(q)$ *is* $\natural\mathbf{P}$*-hard.*

The aim of the current paper is to establish a complexity dichotomy like Theorem 2 for conjunctive queries with self-joins.

5. UNIRELATIONAL QUERIES

Recall that a Boolean conjunctive query q is called unirelational if it does not refer to two distinct relation names. In this section, we focus on the complexity of the problem $\natural$CERTAINTY(q) when q is unirelational, using exclusively relation name R. We will impose no restriction on the signature of R. Sections 5.1 and 5.2 establish syntactic conditions on q that guarantee intractability and tractability of $\natural$CERTAINTY(q).

Lemmas 2 and 5 are the deepest new results in this paper. They are the main tools for proving the complexity dichotomies in Section 7.

5.1 Intractability Result

We first recall the notion of minimality of Boolean conjunctive queries. We then prove a powerful lemma which is useful for establishing intractability of $\natural$CERTAINTY(q) if q is a unirelational Boolean conjunctive query. The lemma will be illustrated by an example.

Definition 2. A Boolean conjunctive query q is minimal if there exists no Boolean conjunctive query q' such that $|q'| < |q|$ and q' is equivalent to q.

LEMMA 2. *Let q be a unirelational Boolean conjunctive query using relation name R such that q is minimal and no two distinct atoms of q agree on all primary-key positions. Let q' be the self-join-free Boolean conjunctive query obtained from q by replacing each occurrence of R with a new relation name of the same signature as R. If q' is unsafe, then $\natural$CERTAINTY(q) is $\natural\mathbf{P}$-hard.*

Example 3. Let $q = \{R(\underline{x}, y), R(\underline{y}, x)\}$. The query q satisfies the premise of Lemma 2 because it is minimal and contains no two distinct atoms that agree on their primary key. The self-join-free query $q' = \{S(\underline{x}, y), T(\underline{y}, x)\}$ is obtained by replacing in q each occurrence of R with a new relation name. It can be easily verified that Function IsSafe returns **false** on input q', i.e., q' is unsafe. By Lemma 2, $\natural$CERTAINTY(q) is $\natural\mathbf{P}$-hard.

The following proof of Lemma 2 uses three sublemmas for readability reasons. After the proof, we will explain how the restrictions on q in the premise of Lemma 2 can be easily met.

PROOF OF LEMMA 2. Assume q' is unsafe. By Theorem 2, $\natural$CERTAINTY(q') is $\natural\mathbf{P}$-hard. To establish $\natural\mathbf{P}$-hardness of $\natural$CERTAINTY(q), it suffices to show a polynomial-time Turing reduction from the $\natural\mathbf{P}$-hard problem $\natural$CERTAINTY(q') to $\natural$CERTAINTY(q). This is the object of the remainder of the proof.

Let $R, S_1, \dots, S_m$ be relation names of same signature $[n, k]$ such that

$$\begin{aligned} q &= \{R(x_{i1}, \dots, x_{in})\}_{i=1}^{m}, \text{ and} \\ q' &= \{S_i(x_{i1}, \dots, x_{in})\}_{i=1}^{m}. \end{aligned}$$

Let σ be the following mapping on S_i-facts. For $i \in \{1, \dots, m\}$, if $G = S_i(a_1, \dots, a_n)$, then

$$\sigma(G) = R((a_1, x_{i1}), \dots, (a_n, x_{in})).$$

The pairs (a, s), where a is a constant and s a symbol, denote constants, such that $(a_1, s_1) = (a_2, s_2)$ if and only if $a_1 = a_2$ and $s_1 = s_2$. If $a \neq s$, then (a, s) denotes a new constant not occurring elsewhere. If $a = s$, then (a, s) is the constant a.

Clearly, for all $i, j \in \{1, \dots, m\}$ such that $i \neq j$, for all S_i-facts G_1, G_2, for each S_j-fact H,

- $G_1 = G_2$ iff $\sigma(G_1) = \sigma(G_2)$;
- G_1 and G_2 are key-equal iff $\sigma(G_1)$ and $\sigma(G_2)$ are key-equal;
- $\sigma(G_1)$ and $\sigma(H)$ are not key-equal because by the premise in the statement of Lemma 2, it is the case that $\langle x_{i1}, \dots, x_{ik}\rangle \neq \langle x_{j1}, \dots, x_{jk}\rangle$.

Let $\mathbf{db}$ be an uncertain database that uses only relation names among $S_1, \dots, S_m$. We define $\sigma(\mathbf{db}) := \{\sigma(F) \mid F \in \mathbf{db}\}$.

It can now be easily seen that $\mathsf{rset}(\sigma(\mathbf{db})) = \{\sigma(\mathbf{r}) \mid \mathbf{r} \in \mathsf{rset}(\mathbf{db})\}$ and $\natural\mathsf{rset}(\sigma(\mathbf{db})) = \natural\mathsf{rset}(\mathbf{db})$. It suffices now to show that for each repair $\mathbf{r}$ of $\mathbf{db}$,

$$\mathbf{r} \models q' \iff \sigma(\mathbf{r}) \models q.$$

$\boxed{\Longrightarrow}$ Assume $\mathbf{r} \models q'$. We can assume a valuation ν such that $\nu(q') \subseteq \mathbf{r}$. Let $\hat{\nu}$ be the valuation such that for every symbol x, we have $\hat{\nu}(x) = (\nu(x), x)$. Notice that $\hat{\nu}$ is indeed the identity on constants, because for every constant a, $\hat{\nu}(a) = (a, a) = a$. The following paragraph shows that $\hat{\nu}(q) \subseteq \sigma(\mathbf{r})$.

Let $1 \leq i \leq m$. Let $a_1, \dots, a_n$ be constants such that $\nu(x_{i1}) = a_1, \dots, \nu(x_{in}) = a_n$. Consequently, $\hat{\nu}(x_{i1}) = (a_1, x_{i1}), \dots, \hat{\nu}(x_{in}) = (a_n, x_{in})$. From $\nu(q') \subseteq \mathbf{r}$, it follows $S_i(a_1, \dots, a_n) \in \mathbf{r}$, hence $R(\hat{\nu}(x_{i1}), \dots, \hat{\nu}(x_{in})) \in \sigma(\mathbf{r})$.

$\boxed{\Longleftarrow}$ Assume $\sigma(\mathbf{r}) \models q$. We can assume a valuation θ over $\mathsf{vars}(q)$ such that $\theta(q) \subseteq \sigma(\mathbf{r})$. We can assume the existence of a total function $\pi : \{1, \dots, m\} \to \{1, \dots, m\}$ such that for each $i \in \{1, \dots, m\}$, we can assume some fact $S_{\pi(i)}(a_{i1}, \dots, a_{in})$ in $\mathbf{r}$ such that

$$R(\theta(x_{i1}), \dots, \theta(x_{in})) = \sigma\big(S_{\pi(i)}(a_{i1}, \dots, a_{in})\big).$$

Since

$$\sigma\big(S_{\pi(i)}(a_{i1}, \dots, a_{in})\big) = R((a_{i1}, x_{\pi(i)1}), \dots, (a_{in}, x_{\pi(i)n})),$$

we obtain that for $1 \leq i \leq m$ and $1 \leq j \leq n$,

$$\theta(x_{ij}) = (a_{ij}, x_{\pi(i)j}).$$

SUBLEMMA 1. *If $x_{ij} = x_{k\ell}$, then $a_{ij} = a_{k\ell}$ and $x_{\pi(i)j} = x_{\pi(k)\ell}$.*

PROOF OF SUBLEMMA 1. Let $x_{ij} = x_{k\ell}$. Since $\theta(x_{ij}) = \theta(x_{k\ell})$, it follows $(a_{ij}, x_{\pi(i)j}) = (a_{k\ell}, x_{\pi(k)\ell})$. Consequently, $a_{ij} = a_{k\ell}$ and $x_{\pi(i)j} = x_{\pi(k)\ell}$. $\square$

SUBLEMMA 2. *If x_{ij} is a constant, then $x_{ij} = a_{ij} = x_{\pi(i)j}$.*

PROOF OF SUBLEMMA 2. Let x_{ij} be a constant. Since $\theta(x_{ij}) = (a_{ij}, x_{\pi(i)j})$ and θ is the identity on constants, $x_{ij} = a_{ij} = x_{\pi(i)j}$. $\square$

We distinguish two cases.

Case π is a permutation. We will use the following easy sublemma.

SUBLEMMA 3. *Let π be a permutation of some nonempty finite set S. There exists integer $p \geq 0$ such that for all $a \in S$, we have $\pi^{-1}(a) = \pi^p(a)$.*

PROOF. Let C be the set of positive integers such that $k \in C$ if and only if π contains a cycle with k elements. Let $p = \left(\prod_{k \in C} k\right) - 1$. Obviously, for every $a \in S$, we have $\pi^p(a) = \pi^{-1}(a)$. □

Let ω be the valuation such that for all $i \in \{1, \dots, m\}$, $j \in \{1, \dots, n\}$, $\omega(x_{ij}) = a_{\pi^{-1}(i)j}$. By Sublemma 3, we can assume an integer $p \geq 0$ such that for all $i \in \{1, \dots, m\}$, we have $\pi^{-1}(i) = \pi^p(i)$. Consequently,

$$\omega(x_{ij}) = a_{\pi^{-1}(i)j} = a_{\pi^p(i)j}.$$

We show that ω is well defined.

1. Assume $x_{ij} = x_{k\ell}$. By repeated application of Sublemma 1, we obtain $a_{\pi^p(i)j} = a_{\pi^p(k)\ell}$.
2. Assume x_{ij} is a constant. By repeated application of Sublemma 2, we obtain $x_{ij} = a_{\pi^p(i)j}$.

We show next that $\omega(q') \subseteq \mathbf{r}$, hence $\mathbf{r} \models q'$.

Since π is a permutation,

$$\{1, \dots, m\} = \{\pi(1), \dots, \pi(m)\}.$$

Let $i \in \{1, \dots, m\}$. It suffices to show that the fact

$$S_{\pi(i)}(\omega(x_{\pi(i)1}), \dots, \omega(x_{\pi(i)n}))$$

belongs to $\mathbf{r}$. This is straightforward since

$$S_{\pi(i)}(\omega(x_{\pi(i)1}), \dots, \omega(x_{\pi(i)n})) = S_{\pi(i)}(a_{i1}, \dots, a_{in}),$$

and $S_{\pi(i)}(a_{i1}, \dots, a_{in})$ belongs to $\mathbf{r}$ by definition.

Case π is not a permutation. Let μ be the substitution such that for all x_{ij}, we have $\mu(x_{ij}) = x_{\pi(i)j}$.

We show that ω is well defined.

1. Assume $x_{ij} = x_{k\ell}$. By application of Sublemma 1, we obtain $x_{\pi(i)j} = x_{\pi(k)\ell}$.
2. Assume x_{ij} is a constant. By application of Sublemma 2, we obtain $x_{ij} = x_{\pi(i)j}$.

Then $\mu(q) \subsetneq q$, hence q is not minimal, a contradiction. This concludes the proof of Lemma 2. □

Let q be a unirelational Boolean conjunctive query referring to relation name R. Lemma 2 only applies if q is minimal and contains no two distinct atoms that agree on all primary-key positions. We argue next that these restrictions can be easily met.

For a given unirelational Boolean conjunctive query q with relation name R, we can first chase q by the primary key of R. See [1, p. 174] for a definition of the chase. Two cases can occur:

1. The chase attempts to equate two distinct constants. If this happens, there exists no consistent database that satisfies q. For any uncertain database $\mathbf{db}$, the number of repairs satisfying q is 0. For example, chasing $\{R(\underline{x}, a), R(\underline{x}, b)\}$ by the primary key of R will equate the distinct constants a and b; obviously, no repair can satisfy this query.
2. The chase terminates with a query q'. Clearly, q' contains no two distinct atoms that agree on all primary-key positions (otherwise the chase could be continued). By Proposition 8.4.2 in [1, p. 175], we know that q and q' evaluate to the same truth value on any consistent database. Finally, by Theorem 6.2.6 in [1, p. 119], we can compute a minimal Boolean conjunctive query $q'' \subseteq q'$ such that q'' is equivalent to q'. Clearly, q'' satisfies the premise of Lemma 2 and for every uncertain database $\mathbf{db}$, $\natural\mathsf{rset}(\mathbf{db}, q) = \natural\mathsf{rset}(\mathbf{db}, q'')$.

Thus the following lemma holds.

LEMMA 3. *For every unirelational Boolean conjunctive query q with relation name R, exactly one of the following statements is true, and it can be decided which one is true:*

1. *for every uncertain database $\mathbf{db}$, $\natural\mathsf{rset}(\mathbf{db}, q) = 0$; or*
2. *there exists a computable unirelational Boolean conjunctive query q' with relation name R such that*
   - *for every uncertain database $\mathbf{db}$, $\natural\mathsf{rset}(\mathbf{db}, q') = \natural\mathsf{rset}(\mathbf{db}, q)$; and*
   - *q' is minimal and contains no two distinct atoms that agree on all primary-key positions.*

5.2 Tractability Results

The two lemmas of this section provide computation rules for $\mathsf{rfrac}(\mathbf{db}, q)$ where q is a unirelational Boolean conjunctive query, and $\mathbf{db}$ is an uncertain database. Lemma 4 provides a recursive rule that is the analogue of rule SE3 in Function IsSafe. The other recursive rules of Function IsSafe do not carry over to conjunctive queries with self-joins. Instead, Lemma 5 provides a new base rule whose proof makes use of the inclusion-exclusion principle.

LEMMA 4. *Let q be a unirelational Boolean conjunctive query such that for some atom $F \in q$, for some variable x, $\mathsf{key}(F) = \emptyset$ and $x \in \mathsf{vars}(F)$. Then, for every uncertain database $\mathbf{db}$,*

$$\mathsf{rfrac}(\mathbf{db}, q) = \sum_{a \in \mathbf{adom}(\mathbf{db})} \mathsf{rfrac}(\mathbf{db}, q_{[x \mapsto a]}) \ .$$

PROOF. Obviously,

$$\mathsf{rset}(\mathbf{db}, q) = \bigcup_{c \in \mathbf{adom}(\mathbf{db})} \mathsf{rset}(\mathbf{db}, q_{[x \mapsto c]}) \ .$$

It suffices thus to show that for $c_1, c_2 \in \mathbf{adom}(\mathbf{db})$ with $c_1 \neq c_2$,

$$\mathsf{rset}(\mathbf{db}, q_{[x \mapsto c_1]}) \cap \mathsf{rset}(\mathbf{db}, q_{[x \mapsto c_2]}) = \emptyset \ . \qquad (1)$$

Assume towards a contradiction some repair $\mathbf{r}$ of $\mathbf{db}$ such that $\mathbf{r} \in \mathsf{rset}(\mathbf{db}, q_{[x \mapsto c_1]}) \cap \mathsf{rset}(\mathbf{db}, q_{[x \mapsto c_2]})$. Assume without loss of generality that F is of the form $R(\underline{\vec{a}}, x, \dots)$. Then, $R(\underline{\vec{a}}, c_1, \dots)$, $R(\underline{\vec{a}}, c_2, \dots) \in \mathbf{r}$, hence $\mathbf{r}$ contains two distinct, key-equal facts, a contradiction. We conclude by contradiction that (1) holds. This concludes the proof of Lemma 4. □

LEMMA 5. *Let q be a unirelational Boolean conjunctive query with relation name R. Let Q be a partition of q such that for all $o, p \in Q$,*[4]

[4]Q is a partition of q if the elements of Q are pairwise disjoint, nonempty subsets of q such that $\bigcup_{p \in Q} p = q$.

1. *if $o \neq p$, then $\mathsf{vars}(o) \cap \mathsf{vars}(p) = \emptyset$; and*

2. *for all $F, G \in [\![p]\!]$, $\mathsf{key}(F) = \mathsf{key}(G)$.*

Then, $\natural\mathsf{CERTAINTY}(q)$ is in **FP**.

Example 4. Let R be a relation name of signature $[3, 2]$. Let a, b be constants. Let

$$\begin{aligned} q &= \{R(\underline{x,y},z), R(\underline{y,x},z), R(\underline{a,z},u), R(\underline{w,a},b)\} \\ p_1 &= \{R(\underline{x,y},z), R(\underline{y,x},z), R(\underline{a,z},u)\} \\ p_2 &= \{R(\underline{w,a},b)\} \end{aligned}$$

We have $[\![p_1]\!] = \{R(\underline{x,y},z), R(\underline{y,x},z)\}$. It is now straightforward to verify that $Q = \{p_1, p_2\}$ is a partition of q that satisfies the premise of Lemma 5. Consequently, $\natural\mathsf{CERTAINTY}(q)$ is in **FP**.

The following proof of Lemma 5 uses three sublemmas for readability reasons.

PROOF OF LEMMA 5. Let $\mathbf{db}$ be an uncertain database of R-facts that is the input of $\natural\mathsf{CERTAINTY}(q)$. For every $P \subseteq Q$, we define $\mathsf{FALS}(P)$ as follows:

$$\mathsf{FALS}(P) := \{\mathbf{r} \in \mathsf{rset}(\mathbf{db}) \mid \forall p \in P(\mathbf{r} \not\models p)\}.$$

We will use the following sublemma.

SUBLEMMA 4. *$\bigcup_{p \in Q} \mathsf{FALS}(\{p\}) = \{\mathbf{r} \in \mathsf{rset}(\mathbf{db}) \mid \mathbf{r} \not\models q\}$.*

PROOF OF SUBLEMMA 4. Let $\mathbf{r}$ be an arbitrary repair of $\mathbf{db}$.

$\boxed{\subseteq}$ Assume $\mathbf{r} \in \bigcup_{p \in Q} \mathsf{FALS}(\{p\})$. We can assume $p_0 \in Q$ such that $\mathbf{r} \in \mathsf{FALS}(\{p_0\})$. Then $\mathbf{r} \not\models p_0$. Since $p_0 \subseteq q$, it follows $\mathbf{r} \not\models q$.

$\boxed{\supseteq}$ Assume $\mathbf{r} \not\models q$. By the first item in the statement of Lemma 5, we can assume $p_0 \in Q$ such that $\mathbf{r} \not\models p_0$. Then $\mathbf{r} \in \mathsf{FALS}(\{p_0\})$. It follows $\mathbf{r} \in \bigcup_{p \in Q} \mathsf{FALS}(\{p\})$. This concludes the proof of Sublemma 4. □

Let $N = |\mathsf{rset}(\mathbf{db})|$ and $M = |\{\mathbf{r} \in \mathsf{rset}(\mathbf{db}) \mid \mathbf{r} \not\models q\}|$. Then, $\natural\mathsf{rset}(\mathbf{db}, q) = N - M$. It suffices to show that M can be computed in polynomial time in the size of $\mathbf{db}$.

By Sublemma 4 and the inclusion-exclusion principle,

$$M = \sum_{k=1}^{|Q|} \Big((-1)^{k-1} \sum_{P \subseteq Q, |P| = k} |\mathsf{FALS}(P)| \Big). \tag{2}$$

In the remainder, we show how to compute in polynomial time $|\mathsf{FALS}(P)|$ for each nonempty subset P of Q.

We define $\sigma : 2^{\mathbf{db}} \to 2^{Q}$ such that for every subset $\mathbf{db}_0 \subseteq \mathbf{db}$, the set $\sigma(\mathbf{db}_0)$ contains $p \in Q$ if there exists a valuation θ over $\mathsf{vars}(p)$ such that

1. $\theta([\![p]\!]) \subseteq \mathbf{db}_0$; and

2. $\theta(p)$ is a consistent subset of $\mathbf{db}$.

The second condition implies that some repair of $\mathbf{db}$ satisfies $\theta(p)$. Obviously, σ is computable in polynomial time data complexity.

SUBLEMMA 5. *Let $P \subseteq Q$. Let $\{\mathbf{db}_1, \mathbf{db}_2, \ldots, \mathbf{db}_n\}$ be a partition of $\mathbf{db}$ such that the following hold:*

1. *every block of $\mathbf{db}$ is entirely contained in some (unique) $\mathbf{db}_i$ ($1 \leq i \leq n$); and*

2. *for every $p \in P$, for every valuation θ over $\mathsf{vars}(p)$, if $\theta([\![p]\!]) \subseteq \mathbf{db}$, then for some $1 \leq i \leq n$, $\theta([\![p]\!]) \subseteq \mathbf{db}_i$.*

Then,

$$|\mathsf{FALS}(P)| = \prod_{1=1}^{n} |\{\mathbf{r}_i \in \mathsf{rset}(\mathbf{db}_i) \mid \sigma(\mathbf{r}_i) \cap P = \emptyset\}|. \tag{3}$$

PROOF OF SUBLEMMA 5. We first show that for every repair $\mathbf{r}$ of $\mathbf{db}$,

$$\mathbf{r} \in \mathsf{FALS}(P) \iff \begin{array}{l} \text{for each } 1 \leq i \leq n, \\ \sigma(\mathbf{r} \cap \mathbf{db}_i) \cap P = \emptyset. \end{array} \tag{4}$$

Let $\mathbf{r}$ be an arbitrary repair of $\mathbf{db}$.

$\boxed{\Longleftarrow}$ Proof by contraposition. Assume $\mathbf{r} \notin \mathsf{FALS}(P)$. We can assume $p_0 \in P$ such that $\mathbf{r} \models p_0$. Then we can assume a valuation θ such that $\theta(p_0) \subseteq \mathbf{r}$, hence $\theta([\![p_0]\!]) \subseteq \mathbf{r}$. By the second item in the premise of Sublemma 5, there exists $1 \leq i \leq n$ such that $\theta([\![p_0]\!]) \subseteq \mathbf{db}_i$. By the definition of σ, we have $p_0 \in \sigma(\mathbf{r} \cap \mathbf{db}_i)$, hence $\sigma(\mathbf{r} \cap \mathbf{db}_i) \cap P \neq \emptyset$.

$\boxed{\Longrightarrow}$ Proof by contraposition. Assume that for some $1 \leq i \leq n$, $\sigma(\mathbf{r} \cap \mathbf{db}_i) \cap P \neq \emptyset$. We can assume $p_0 \in P$ such that $p_0 \in \sigma(\mathbf{r} \cap \mathbf{db}_i)$. By the definition of σ, we can assume a valuation θ over $\mathsf{vars}(p_0)$ such that $\theta([\![p_0]\!]) \subseteq \mathbf{r} \cap \mathbf{db}_i$ and $\theta(p_0)$ is a consistent subset of $\mathbf{db}$. Consequently, $\theta([\![p_0]\!]) \subseteq \mathbf{r}$ and we can assume a repair $\mathbf{r}_0$ of $\mathbf{db}$ such that $\theta(p_0) \subseteq \mathbf{r}_0$. Let θ' be the restriction of θ to $\mathsf{vars}([\![p_0]\!])$. From $\theta(p_0) \subseteq \mathbf{r}_0$, it follows $\mathbf{r}_0 \models \theta'(p_0)$. From $\theta([\![p_0]\!]) \subseteq \mathbf{r}$, it follows $\theta'([\![p_0]\!]) \subseteq \mathbf{r}$. From $\mathbf{r}_0 \models \theta'(p_0)$ and $\theta'([\![p_0]\!]) \subseteq \mathbf{r}$, it follows $\mathbf{r} \models \theta'(p_0)$ by Lemma 1, hence $\mathbf{r} \models p_0$. Since $\mathbf{r} \models p_0$, $\mathbf{r} \notin \mathsf{FALS}(P)$. This concludes the proof of (4).

By the first item in the premise of Sublemma 5, every repair $\mathbf{r}$ of $\mathbf{db}$ can be written as a disjoint union $\mathbf{r} = \mathbf{r}_1 \uplus \mathbf{r}_2 \uplus \cdots \uplus \mathbf{r}_n$ where for every $1 \leq i \leq n$, $\mathbf{r}_i := \mathbf{r} \cap \mathbf{db}_i$ is a repair of $\mathbf{db}_i$. By (4), such a repair $\mathbf{r}$ belongs to $\mathsf{FALS}(P)$ if and only if for $1 \leq i \leq n$, $\sigma(\mathbf{r}_i) \cap P = \emptyset$. Therefore, it is correct to conclude (3). This terminates the proof of Sublemma 5 □

For every $P \subseteq Q$, we define the binary relation $\sim_P$ on $\mathbf{db}$ as the transitive closure of:

$F_1 \sim_P F_2$ if one of the following holds:

1. the facts F_1 and F_2 are key-equal; or
2. for some $p \in P$, for some valuation θ over $\mathsf{vars}(p)$, we have $F_1, F_2 \in \theta([\![p]\!]) \subseteq \mathbf{db}$.

Let $P \subseteq Q$. The equivalence classes of $\sim_P$ can be computed in polynomial time, because for every $p \in P$, there are at most polynomially many (in the size of $\mathbf{db}$) distinct valuations $\theta : \mathsf{vars}(p) \to \mathbf{adom}(\mathbf{db})$. The following sublemma states that every equivalence class has at most polynomially many repairs.

SUBLEMMA 6. *Let $P \subseteq Q$. Let $\{\mathbf{db}_1, \mathbf{db}_2, \ldots, \mathbf{db}_n\}$ be the partition of $\mathbf{db}$ induced by $\sim_P$. For every $1 \leq i \leq n$, $\natural\mathsf{rset}(\mathbf{db}_i)$ is polynomially bounded in the size of $\mathbf{db}$.*

PROOF OF SUBLEMMA 6. Let $1 \leq i \leq n$. Let F be an arbitrary fact of $\mathbf{db}_i$. We show hereinafter that for every $H \in \mathbf{db}_i$, if b is a constant that occurs at some primary-key position in H, then either b occurs in q or b occurs at some primary-key position in F (or both). Consequently, we can assume integer ℓ, which does not depend on $\mathbf{db}$, such that the number of distinct blocks in $\mathbf{db}_i$ is less than or equal to ℓ. Then $\natural\mathsf{rset}(\mathbf{db}_i)$ is bounded by $|\mathbf{db}|^{\ell}$.

Let $H \in \mathbf{db}_i$. Then, there exists a sequence

$$F_1, F_2, \ldots, F_m$$

of facts in $\mathbf{db}_i$ such that $F_1 = F$, $F_m = H$, and for each $j \in \{1, \ldots, m-1\}$, one of the following holds:

1. the facts F_j and F_{j+1} are key-equal; or
2. for some $p \in P$, for some valuation θ over $\mathsf{vars}(p)$, we have $F_j, F_{j+1} \in \theta(\llbracket p \rrbracket) \subseteq \mathbf{db}$.

It suffices to show that for every $j \in \{1, \ldots, m\}$, if b is a constant that occurs at some primary-key position in F_j, then either b occurs in q or b occurs at some primary-key position in F_1 (or both). The proof runs by induction on increasing j. The desired result holds obviously for $j = 1$. For the induction step, $j \to j+1$, let b be a constant such that b occurs at some primary-key position in F_{j+1} and b does not occur in q. Two cases can occur.

1. F_j and F_{j+1} are key-equal. Clearly, b occurs at some primary-key position in F_j.
2. There exists $p \in P$ and a valuation θ over $\mathsf{vars}(p)$ such that $F_j, F_{j+1} \in \theta(\llbracket p \rrbracket) \subseteq \mathbf{db}$. We can assume $G_1, G_2 \in \llbracket p \rrbracket$ such that $F_j = \theta(G_1)$ and $F_{j+1} = \theta(G_2)$. Since b does not occur in q, we can assume $x \in \mathsf{key}(G_2)$ such that $\theta(x) = b$. From $\mathsf{key}(G_1) = \mathsf{key}(G_2)$, it follows $x \in \mathsf{key}(G_1)$, hence b occurs at some primary-key position in F_j.

Since b occurs at some primary-key position in F_j, by the induction hypothesis, b occurs at some primary-key position in F_1. This concludes the proof of Sublemma 6. $\square$

By the definition of $\sim_P$, the set of equivalence classes of $\sim_P$ satisfies the first and the second item in Sublemma 5. By Sublemma 6, the righthand expression in equation (3) can be computed in polynomial time. It follows that the righthand expression in equation (2) can be computed in polynomial time. This concludes the proof of Lemma 5. $\square$

We point out that the lemmas in this section are not sufficient to establish the (in)tractability of each problem $\natural\mathsf{CERTAINTY}(q)$ where q is a unirelational Boolean conjunctive query. Consider the minimal query $q = \{R(\underline{x, y}, a), R(\underline{z, x}, a)\}$, where a is a constant. Since the query $\{S(\underline{x, y}, a), T(\underline{z, x}, a)\}$ is safe, Lemma 2 provides no information on the complexity of $\natural\mathsf{CERTAINTY}(q)$. Further, q does not satisfy the premises of Lemmas 4 and 5.

6. ENCODING MULTIPLE RELATIONS IN A SINGLE RELATION

The results in Section 5 assume a single relation name denoted R. This assumption is not severe, because it is easy to encode multiple relations into a single relation. Although the following definition is somewhat technical, the idea is simple and illustrated by Fig. 4, which shows how to encode two relations R and S with different signatures in one relation N. The R-fact $R(\underline{a, b}, c)$ is encoded as $N(\underline{R, a, b}, c, 0)$,

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

S	$\underline{1}$	2	3
	a	b	c
	e	f	c

N	$\underline{1}$	$\underline{2}$	$\underline{3}$	4	5
	R	a	b	c	0
	R	a	b	d	0
	S	a	0	b	c
	S	e	0	f	c

Figure 4: The relations R and S are encoded in N. The signatures of R, S, and N are $[3,2]$, $[3,1]$, and $[5,3]$ respectively. The leftmost position of N is used for tagging tuples with their relation of origin. The symbol 0 is used for padding.

and the S-fact $S(\underline{a}, b, c)$ is encoded as $N(\underline{S, a, 0}, b, c)$. The leftmost position in N is used for storing the original relation of the encoded fact; the encoding preserves primary-key positions; extra positions are padded with 0's. The encoding naturally extends to conjunctive queries.

Definition 3. Let $\mathbf{S}$ be a database schema (i.e., a finite set of relation names). Let ℓ and m be the smallest integers such that for every relation name $R \in \mathbf{S}$, if R has signature $[n, k]$, then $k \leq \ell$ and $n - k \leq m$.

Let N be a relation name of signature $[\ell + m + 1, \ell + 1]$ such that $N \notin \mathbf{S}$. Let 0 be a constant, arbitrarily picked in the set of constants. For every relation name R in $\mathbf{S}$, for every R-atom $F = R(\underline{s_1, \ldots, s_k}, t_1, \ldots, t_j)$, we define

$$\mathsf{enc}_{\mathbf{S}}(F) = N(\underline{R, s_1, \ldots, s_\ell}, t_1, \ldots, t_m),$$

where $s_{k+1} = \cdots = s_\ell = 0 = t_{j+1} = \cdots = t_m$, i.e., extra positions are padded with 0's.

The function $\mathsf{enc}_{\mathbf{S}}$ naturally extends to sets. Let A be a finite set of atoms using only the relation names in $\mathbf{S}$. Typically, A will be a Boolean conjunctive query or a set of facts. We define $\mathsf{enc}_{\mathbf{S}}(A) = \{\mathsf{enc}_{\mathbf{S}}(F) \mid F \in A\}$.

The following lemmas are fairly straightforward.

Lemma 6. *Let $\mathbf{S}$ be a database schema. Let F, G be atoms whose relation names belong to $\mathbf{S}$. Then,*

1. *$F = G$ iff $\mathsf{enc}_{\mathbf{S}}(F) = \mathsf{enc}_{\mathbf{S}}(G)$.*
2. *F and G are key-equal facts iff $\mathsf{enc}_{\mathbf{S}}(F)$ and $\mathsf{enc}_{\mathbf{S}}(G)$ are key-equal facts.*

Lemma 7. *Let $\mathbf{S}$ be a database schema. Let q be a Boolean conjunctive query using only the relation names in $\mathbf{S}$. Then, the problems $\natural\mathsf{CERTAINTY}(q)$ and $\natural\mathsf{CERTAINTY}(\mathsf{enc}_{\mathbf{S}}(q))$ are equivalent under polynomial-time many-one reductions.*

Proof. In the first part of the proof, we will show a polynomial-time many-one reduction from $\natural\mathsf{CERTAINTY}(q)$ to $\natural\mathsf{CERTAINTY}(\mathsf{enc}_{\mathbf{S}}(q))$. Let $\mathbf{db}$ be an uncertain database using only relation names in $\mathbf{S}$ (it is trivial to extend the proof to deal with relation names outside $\mathbf{S}$), which is an instance of $\natural\mathsf{CERTAINTY}(q)$. We will show

$$\natural\mathsf{rset}(\mathbf{db}, q) = \natural\mathsf{rset}(\mathsf{enc}_{\mathbf{S}}(\mathbf{db}), \mathsf{enc}_{\mathbf{S}}(q)).$$

Since $\mathsf{enc}_{\mathbf{S}}(\mathbf{db})$ can be computed in polynomial time, this suffices to conclude that $\natural\mathsf{CERTAINTY}(q)$ can be polynomially reduced to $\natural\mathsf{CERTAINTY}(\mathsf{enc}_{\mathbf{S}}(q))$.

From Lemma 6, it follows $\mathsf{rset}(\mathsf{enc}_{\mathbf{S}}(\mathbf{db})) = \{\mathsf{enc}_{\mathbf{S}}(\mathbf{r}) \mid \mathbf{r} \in \mathsf{rset}(\mathbf{db})\}$ and $\natural\mathsf{rset}(\mathsf{enc}_{\mathbf{S}}(\mathbf{db})) = \natural\mathsf{rset}(\mathbf{db})$. Consequently, it is sufficient to show that for every repair $\mathbf{r}$ of $\mathbf{db}$,

$$\mathbf{r} \models q \iff \mathsf{enc}_{\mathbf{S}}(\mathbf{r}) \models \mathsf{enc}_{\mathbf{S}}(q).$$

Let $\mathbf{r}$ be a repair of $\mathbf{db}$. Clearly, $\mathsf{vars}(q) = \mathsf{vars}(\mathsf{enc}_{\mathbf{S}}(q))$.

$\boxed{\Longrightarrow}$ Assume $\mathbf{r} \models q$. We can assume a valuation θ over $\mathsf{vars}(q)$ such that $\theta(q) \subseteq \mathbf{r}$, hence $\mathsf{enc}_{\mathbf{S}}(\theta(q)) \subseteq \mathsf{enc}_{\mathbf{S}}(\mathbf{r})$. It suffices to show $\theta(\mathsf{enc}_{\mathbf{S}}(q)) \subseteq \mathsf{enc}_{\mathbf{S}}(\mathbf{r})$. To this extent, let F be an arbitrary atom of q. It suffices to show $\theta(\mathsf{enc}_{\mathbf{S}}(F)) \in \mathsf{enc}_{\mathbf{S}}(\mathbf{r})$. From $\mathsf{enc}_{\mathbf{S}}(\theta(q)) \subseteq \mathsf{enc}_{\mathbf{S}}(\mathbf{r})$, it follows $\mathsf{enc}_{\mathbf{S}}(\theta(F)) \in \mathsf{enc}_{\mathbf{S}}(\mathbf{r})$. Since the equality $\mathsf{enc}_{\mathbf{S}}(\theta(F)) = \theta(\mathsf{enc}_{\mathbf{S}}(F))$ is obvious, it follows $\theta(\mathsf{enc}_{\mathbf{S}}(F)) \in \mathsf{enc}_{\mathbf{S}}(\mathbf{r})$.

$\boxed{\Longleftarrow}$ Assume $\mathsf{enc}_{\mathbf{S}}(\mathbf{r}) \models \mathsf{enc}_{\mathbf{S}}(q)$. We can assume a valuation θ over $\mathsf{vars}(q)$ such that $\theta(\mathsf{enc}_{\mathbf{S}}(q)) \subseteq \mathsf{enc}_{\mathbf{S}}(\mathbf{r})$. It suffices to show $\theta(q) \subseteq \mathbf{r}$. To this extent, let F be an arbitrary atom of q. It suffices to show $\theta(F) \in \mathbf{r}$. From $\theta(\mathsf{enc}_{\mathbf{S}}(q)) \subseteq \mathsf{enc}_{\mathbf{S}}(\mathbf{r})$, it follows $\theta(\mathsf{enc}_{\mathbf{S}}(F)) \in \mathsf{enc}_{\mathbf{S}}(\mathbf{r})$. Since $\theta(\mathsf{enc}_{\mathbf{S}}(F)) = \mathsf{enc}_{\mathbf{S}}(\theta(F))$ is obvious, it follows $\mathsf{enc}_{\mathbf{S}}(\theta(F)) \in \mathsf{enc}_{\mathbf{S}}(\mathbf{r})$. We can assume a fact $G \in \mathbf{r}$ such that $\mathsf{enc}_{\mathbf{S}}(\theta(F)) = \mathsf{enc}_{\mathbf{S}}(G)$. By Lemma 6, $\theta(F) = G$, hence $\theta(F) \in \mathbf{r}$.

In the second part of the proof, we show a polynomial-time many-one reduction from the problem $\natural\mathsf{CERTAINTY}(\mathsf{enc}_{\mathbf{S}}(q))$ to $\natural\mathsf{CERTAINTY}(q)$. Let N be the relation name such that all atoms in $\mathsf{enc}_{\mathbf{S}}(q)$ are N-atoms. Assume that the arity of N is n. Let $\mathbf{db}$ be a database of N-facts. Let $\mathbf{db}_0$ be the subset of $\mathbf{db}$ containing each fact $N(R, s_1, \dots, s_{n-1})$ such that R is a relation name in $\mathbf{S}$. We can compute in polynomial time the (unique) database $\mathbf{db}_0'$ with schema $\mathbf{S}$ such that $\mathsf{enc}_{\mathbf{S}}(\mathbf{db}_0') = \mathbf{db}_0$. From the first part of the proof, it follows $\mathsf{rfrac}(\mathbf{db}_0, \mathsf{enc}_{\mathbf{S}}(q)) = \mathsf{rfrac}(\mathbf{db}_0', q)$. From the obvious observation that $\mathsf{rfrac}(\mathbf{db}, \mathsf{enc}_{\mathbf{S}}(q)) = \mathsf{rfrac}(\mathbf{db}_0, \mathsf{enc}_{\mathbf{S}}(q))$, it follows $\mathsf{rfrac}(\mathbf{db}, \mathsf{enc}_{\mathbf{S}}(q)) = \mathsf{rfrac}(\mathbf{db}_0', q)$. This concludes the proof. $\square$

7. A COMPLEXITY DICHOTOMY

Recall that a relation name R is simple-key if its primary key is a singleton (i.e., if R's signature is $[n, 1]$ for some n). In this section, we use the tool lemmas of Section 5 to prove a complexity dichotomy in the class containing all problems $\natural\mathsf{CERTAINTY}(q)$ where q is a Boolean conjunctive query in which all relation names are simple-key.

Definition 4. We say that a class $\mathcal{P}$ of function problems *exhibits an effective* $\mathbf{FP}$-$\natural\mathbf{P}$-*dichotomy* if all problems in $\mathcal{P}$ are either in $\mathbf{FP}$ or $\natural\mathbf{P}$-hard and it is decidable whether a given problem in $\mathcal{P}$ is in $\mathbf{FP}$ or $\natural\mathbf{P}$-hard.

If we use the encoding of Section 6 to store multiple simple-key relations in a single relation N, then the signature of N will be $[n, 2]$ for some n, where the leftmost position of N is used for storing relation names. This leads to the following definition.

Definition 5. We define $\mathcal{B}$ as the class that contains all problems $\natural\mathsf{CERTAINTY}(q)$ where q is a unirelational Boolean conjunctive query whose relation name has signature $[n, 2]$ (for some $n \geq 2$) such that for every $F \in q$, the first position of F is a constant.

LEMMA 8. *The class $\mathcal{B}$ exhibits an effective* $\mathbf{FP}$-$\natural\mathbf{P}$-*dichotomy.*

PROOF. Let q be a Boolean conjunctive query such that $\natural\mathsf{CERTAINTY}(q)$ is in $\mathcal{B}$. By Lemma 3, two cases can occur, and it is decidable which case applies:

1. for every uncertain database $\mathbf{db}$, $\natural\mathsf{rset}(\mathbf{db}, q) = 0$; or

Function IsEasy(q) Determine whether the problem $\natural\mathsf{CERTAINTY}(q)$, which belongs to the class $\mathcal{B}$, is easy

```
Input: $q$ is a minimal unirelational Boolean conjunctive query
       with relation name $R$ of some signature $[n,2]$ such that
       (i) for every $R(\underline{s_1, s_2}, s_3, \dots, s_n)$ in $q$, it is the case that
       $s_1$ is a constant; and (ii) no two distinct atoms of $q$
       agree on both primary-key positions.
Result: Boolean in $\{\mathbf{true}, \mathbf{false}\}$.
begin
    if $q$ satisfies the premise of Lemma 5 then
        return true;
    /* $a$ is an arbitrary constant */
    if there exists $F \in q$ such that $\mathsf{key}(F) = \emptyset \neq \mathsf{vars}(F)$ then
        select $F \in q$ such that $\mathsf{key}(F) = \emptyset \neq \mathsf{vars}(F)$;
        select $x \in \mathsf{vars}(F)$;
        return $IsEasy(q_{[x \mapsto a]})$;
    if none of the above then
        return false;
```

2. there exists a computable unirelational Boolean conjunctive query q_m using the same relation name as q such that

   - for every uncertain database $\mathbf{db}$, $\natural\mathsf{rset}(\mathbf{db}, q_m) = \natural\mathsf{rset}(\mathbf{db}, q)$; and
   - q_m satisfies the premise of Lemma 2.

If the first case applies, $\natural\mathsf{CERTAINTY}(q)$ is in constant time data complexity. In what follows, we assume that the second case applies.

Let $\mathbf{db}_0$ be any database whose active domain is the singleton $\{a\}$. In the following computation, only the active domain matters, while the input $\mathbf{db}_0$ is immaterial. Start computing $\mathsf{rfrac}(q_m, \mathbf{db}_0)$ by recursively applying Lemma 4 as long as possible. In this way, since every sum ranges over the singleton $\{a\}$, we obtain a query $\bar{q}$ such that $\mathsf{rfrac}(q_m, \mathbf{db}_0) = \mathsf{rfrac}(\bar{q}, \mathbf{db}_0)$ and $\bar{q}$ does not satisfy the premise of Lemma 4. That is, for every atom $F \in \bar{q}$, either $\mathsf{key}(F) \neq \emptyset$ or $\mathsf{vars}(F) = \emptyset$. We show hereinafter that if the query $\bar{q}$ satisfies the premise of Lemma 5, then $\natural\mathsf{CERTAINTY}(q)$ is in $\mathbf{FP}$; otherwise $\natural\mathsf{CERTAINTY}(q)$ is $\natural\mathbf{P}$-hard.

For readability, the function IsEasy encodes the computation described in the previous paragraph. The function will always terminate with either **true** or **false**. The remainder of the proof actually shows the following: if IsEasy terminates with **true** on input q, then $\natural\mathsf{CERTAINTY}(q)$ is in $\mathbf{FP}$; otherwise $\natural\mathsf{CERTAINTY}(q)$ is $\natural\mathbf{P}$-hard.

First assume that the query $\bar{q}$ satisfies the premise of Lemma 5. For every uncertain database $\mathbf{db}$, we can compute $\mathsf{rfrac}(\mathbf{db}, q)$ as before by repeated application of Lemmas 4 and 5. The only difference is that the sum in the application of Lemma 4 now ranges over $\mathbf{adom}(\mathbf{db})$ instead of the singleton $\{a\}$. However, since $|\mathbf{adom}(\mathbf{db})|$ is polynomially bounded in the size of $\mathbf{db}$, the computation will terminate after a number of steps that is polynomial in the size of $\mathbf{db}$.

Assume next that the query $\bar{q}$ does not satisfy the premise of Lemma 5. Define the binary relation $\sim$ on $\bar{q}$ as the transitive closure of: $F \sim G$ if $\mathsf{vars}(F) \cap \mathsf{vars}(G) \neq \emptyset$. Let Q be the partition of $\bar{q}$ induced by $\sim$. By the hypothesis that $\bar{q}$ does not satisfy the premise of Lemma 5, it must be the case that for some element $p \in Q$, $[\![p]\!]$ contains two atoms $R(\underline{a, x}, \vec{u})$ and $R(\underline{b, y}, \vec{w})$ where x and y are distinct variables. Let q_m'

be the query obtained from q_m by replacing each occurrence of R with a new relation name. It can now be easily shown that Function IsSafe will return **false** on input q'_m. Thus, q'_m is unsafe. By Lemma 2, $\natural\mathsf{CERTAINTY}(q_m)$ is $\natural\mathbf{P}$-hard. It follows that $\natural\mathsf{CERTAINTY}(q)$ is $\natural\mathbf{P}$-hard. This concludes the proof. $\square$

Definition 6. We define $\mathcal{S}$ as the class containing all problems $\natural\mathsf{CERTAINTY}(q)$ where q is a Boolean conjunctive query in which all relation names are simple-key.

The most appealing result of this paper can now be proved.

THEOREM 3. *The class $\mathcal{S}$ exhibits an effective* $\mathbf{FP}$-$\natural\mathbf{P}$-*dichotomy.*

PROOF. Let q be a Boolean conjunctive query in which all relation names are simple-key. Let $\mathbf{S}$ the set of relation names in q. Let $q' = \mathsf{enc}_{\mathbf{S}}(q)$, which was defined in Definition 3. Since every relation name in q is simple-key, it follows that $\natural\mathsf{CERTAINTY}(q')$ belongs to $\mathcal{B}$. By Lemma 8, the problem $\natural\mathsf{CERTAINTY}(q')$ is either in $\mathbf{FP}$ or $\natural\mathbf{P}$-hard, and it is decidable which of the two cases applies. By Lemma 7, if $\natural\mathsf{CERTAINTY}(q')$ is in $\mathbf{FP}$, then $\natural\mathsf{CERTAINTY}(q)$ is in $\mathbf{FP}$; and if $\natural\mathsf{CERTAINTY}(q')$ is $\natural\mathbf{P}$-hard, then $\natural\mathsf{CERTAINTY}(q)$ is $\natural\mathbf{P}$-hard. This concludes the proof. $\square$

8. CONCLUSION

A relation name is simple-key if its primary key consists of a single attribute. Theorem 3 establishes that for every Boolean conjunctive query q (possibly with self-joins) in which all relation names are simple-key, the counting problem $\natural\mathsf{CERTAINTY}(q)$ is either in $\mathbf{FP}$ or $\natural\mathbf{P}$-hard (and in $\natural\mathbf{P}$ by Theorem 1), and it is decidable which of the two cases applies. Such effective $\mathbf{FP}$-$\natural\mathbf{P}$-dichotomy was so far only known for self-join-free Boolean conjunctive queries [11].

The complexity dichotomy in [11] was inspired by work in block-independent-disjoint (BID) probabilistic databases [5]. On the other hand, the complexity dichotomy of Theorem 3 does not correspond to any known result in BID probabilistic databases. It is an open issue to carry over this dichotomy from uncertain databases to BID probabilistic databases.

It is an open conjecture that for every Boolean conjunctive query q (without any restrictions concerning self-joins or cardinalities of primary keys), the problem $\natural\mathsf{CERTAINTY}(q)$ is either in $\mathbf{FP}$ or $\natural\mathbf{P}$-hard, and it is decidable which of the two cases applies. To those who want to tackle this conjecture, we point out that our tool lemmas in Section 5 impose no restriction on the signatures of relation names, and also apply to Boolean conjunctive queries containing relation names that are not simple-key. Nevertheless, we are also aware that our tool lemmas, despite their power, are still insufficient to solve this conjecture.

9. REFERENCES

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

[2] M. Arenas, L. E. Bertossi, and J. Chomicki. Consistent query answers in inconsistent databases. In V. Vianu and C. H. Papadimitriou, editors, *PODS*, pages 68–79. ACM Press, 1999.

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

[4] N. N. Dalvi, C. Ré, and D. Suciu. Probabilistic databases: diamonds in the dirt. *Commun. ACM*, 52(7):86–94, 2009.

[5] N. N. Dalvi, C. Re, and D. Suciu. Queries and materialized views on probabilistic databases. *J. Comput. Syst. Sci.*, 77(3):473–490, 2011.

[6] N. N. Dalvi and D. Suciu. The dichotomy of probabilistic inference for unions of conjunctive queries. *J. ACM*, 59(6):30, 2012.

[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] A. Fuxman and R. J. Miller. First-order query rewriting for inconsistent databases. *J. Comput. Syst. Sci.*, 73(4):610–635, 2007.

[9] 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.

[10] P. Koutris and D. Suciu. A dichotomy on the complexity of consistent query answering for atoms with simple keys. *CoRR*, abs/1212.6636, 2012.

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

[12] D. Suciu, D. Olteanu, C. Ré, and C. Koch. *Probabilistic Databases*. Synthesis Lectures on Data Management. Morgan & Claypool Publishers, 2011.

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

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