Rewriting Consistent Answers on Annotated Data

PHOKION KOLAITIS, University of California Santa Cruz & IBM Research, USA
NINA PARDAL, University of Huddersfield, University of Sheffield, United Kingdom
JONNI VIRTEMA, University of Sheffield, United Kingdom
JEF WIJSEN, University of Mons, Belgium

We embark on a study of the consistent answers of queries over databases annotated with values from a naturally ordered positive semiring. In this setting, the consistent answers of a query are defined as the minimum of the semiring values that the query takes over all repairs of an inconsistent database. The main focus is on self-join free conjunctive queries and key constraints, which is the most extensively studied case of consistent query answering over standard databases. We introduce a variant of first-order logic with a limited form of negation, define suitable semiring semantics, and then establish the main result of the paper: the consistent query answers of a self-join free conjunctive query under key constraints are rewritable in this logic if and only if the attack graph of the query contains no cycles. This result generalizes an analogous result of Koutris and Wijsen for ordinary databases, but also yields new results for a multitude of semirings, including the bag semiring, the tropical semiring, and the fuzzy semiring. Further, for the bag semiring, we show that computing the consistent answers of any self-join free conjunctive query whose attack graph has a strong cycle is not only NP-hard but also it is NP-hard to even approximate the consistent answers with a constant relative approximation guarantee.

CCS Concepts: • **Information systems → Relational database query languages**; • **Theory of computation → Incomplete, inconsistent, and uncertain databases**; **Logic and databases**.

Additional Key Words and Phrases: consistent query answering, repairs, semirings, conjunctive queries, key constraints

ACM Reference Format:
Phokion Kolaitis, Nina Pardal, Jonni Virtema, and Jef Wijsen. 2025. Rewriting Consistent Answers on Annotated Data. *Proc. ACM Manag. Data* 3, 2 (PODS), Article 110 (May 2025), 26 pages. https://doi.org/10.1145/3725247

1 Introduction and summary of results

Database repairs and the consistent answers of queries provide a principled approach to coping with inconsistent databases, i.e., databases that violate one or more integrity constraints in a given set Σ. This area of research started with the influential work by Arenas, Bertossi, and Chomicki [3] and since then has had a steady presence in database theory. Intuitively, a *repair* of an inconsistent database $\mathfrak{D}$ is a consistent database $\mathfrak{D}'$ (i.e., $\mathfrak{D}'$ satisfies every constraint in Σ) and differs from $\mathfrak{D}$ in a "minimal" way. By definition, the *consistent answers* $\textsc{Cons}(q, \Sigma, \mathfrak{D})$ of a query q over an inconsistent database $\mathfrak{D}$ is the intersection of the evaluations $q(\mathfrak{D}')$ of q over all repairs $\mathfrak{D}'$ of $\mathfrak{D}$. Thus, every set Σ of integrity constraints and every query q give rise to the following algorithmic problem $\textsc{Cons}(q, \Sigma)$: given a database $\mathfrak{D}$, compute $\textsc{Cons}(q, \Sigma, \mathfrak{D})$.

Authors' Contact Information: Phokion Kolaitis, University of California Santa Cruz & IBM Research, Santa Cruz, California, USA, kolaitis@soe.ucsc.edu; Nina Pardal, University of Huddersfield, University of Sheffield, Huddersfield, United Kingdom, n.pardal@hud.ac.uk; Jonni Virtema, University of Sheffield, Sheffield, United Kingdom, j.t.virtema@sheffield.ac.uk; Jef Wijsen, University of Mons, Mons, Belgium, jef.wijsen@umons.ac.be.

ACM 2836-6573/2025/5-ART110
https://doi.org/10.1145/3725247

Since, in general, an inconsistent database may have a multitude of different repairs, computing the consistent answers can be an intractable problem. In fact, computing the consistent answers can be a coNP-hard problem, even for conjunctive queries q and for key constraints. This state of affairs motivated a series of investigations aiming to delineate the boundary between tractability and intractability in the computation of the consistent answers. One of the most striking results along these lines is a *trichotomy* theorem obtained by Koutris and Wijsen [26]. Specifically, Koutris and Wijsen showed that for a self-join free conjunctive query q and a set Σ of key constraints with one key constraint per relation of q, the problem $\textsc{Cons}(q, \Sigma)$ exhibits one of the following three behaviours: $\textsc{Cons}(q, \Sigma)$ is first-order rewritable, or it is polynomial-time computable but it is not first-order rewritable, or it is coNP-complete. Extending this trichotomy theorem to more expressive classes of queries and to richer types of integrity constraints has been the topic of active investigations during the past several years [24, 25, 27, 28].

A different direction in database research has focused on $\mathbb{K}$-databases, i.e., databases in which the tuples in the relations are annotated with values from some fixed semiring $\mathbb{K} = (K, +, \times, 0, 1)$. Ordinary databases correspond to databases over the Boolean semiring $\mathbb{B} = (\{0, 1\}, \vee, \wedge, 0, 1)$, while bag databases correspond to databases over the semiring $\mathbb{N} = (N, +, \times, 0, 1)$ of the non-negative integers. The catalyst for this investigation was the paper by Green, Karvounarakis, and Tannen [19], which developed a powerful framework for data provenance based on semirings of polynomials. While the original framework [19] applied only to the provenance of queries expressible in negation-free first-order logic, subsequent investigations extended the study of provenance to richer languages, including full first-order logic [17], Datalog [12], and least fixed-point logic [11]. Furthermore, several other topics in database theory have been examined in the context of semiring semantics, including the conjunctive query containment problem [18, 23], the evaluation of Datalog queries [22, 38], and the interplay between local consistency and global consistency for relations over semirings [6].

In this paper, we embark on an investigation of the consistent answers of queries under semiring semantics. Our main focus is on conjunctive queries with key constraints and on the rewritability of the certain answers of such queries. The first task is to address the following conceptual questions: How should the consistent answers of queries under semiring semantics be defined? What does it mean to say that the consistent answers of a query under semiring semantics are rewritable in first-order logic? To simplify the exposition, let us assume that the queries considered are *closed*, i.e., they have arity zero or, equivalently, the formulas that define them have no free variables. On ordinary databases, closed queries are called Boolean queries because they take value 0 or 1, but on $\mathbb{K}$-databases they can take any value in the universe K of the semiring $\mathbb{K}$.

To define the notion of the consistent answers of a query under semiring semantics, we first need to define the notion of a repair of a $\mathbb{K}$-database with respect to a set Σ of key constraints, where $\mathbb{K}$ is a fixed semiring. If $\mathfrak{D}$ is a $\mathbb{K}$-database, then the *support* of $\mathfrak{D}$ is the ordinary database $\mathrm{Supp}(\mathfrak{D})$ obtained from $\mathfrak{D}$ by setting value 1 to every tuple in one of the relations of $\mathfrak{D}$ that has a non-zero annotation. In Section 3, we argue that it is natural to define a *repair* of a $\mathbb{K}$-database $\mathfrak{D}$ to be a maximal sub-database $\mathfrak{D}'$ of $\mathfrak{D}$ whose support $\mathrm{Supp}(\mathfrak{D}')$ satisfies the key constraints at hand. Let q be a closed query with a set Σ of key constraints. If $\mathfrak{D}$ is a $\mathbb{K}$-database, then on every repair $\mathfrak{D}'$ of $\mathfrak{D}$, the query returns a value $q(\mathfrak{D}')$ from the universe K of the semiring $\mathbb{K}$. We define the *consistent answers* $\mathrm{mCA}_{\mathbb{K}}(q, \Sigma, \mathfrak{D})$ *of q on $\mathfrak{D}$* to be the minimum of the values $q(\mathfrak{D}')$, as $\mathfrak{D}'$ varies over all repairs of $\mathfrak{D}$. For this definition to be meaningful, we need to assume a total order on the universe K of $\mathbb{K}$. Thus, we define the consistent answers of queries for *naturally ordered positive semirings*, i.e., positive semirings $\mathbb{K}$ in which the *natural* preorder of the semiring is a total order (the precise definitions are given in the next section). The Boolean semiring, the bag semiring, the tropical semiring, and the fuzzy semiring are some of the main examples of naturally ordered positive semirings.

Before introducing the notion of first-order rewritability for semirings, let $\mathrm{q}_{\mathrm{path}}$ be the closed conjunctive query $\exists x \exists y \exists z\, (R(x;y) \wedge S(y;z))$, where the semicolon separates the key positions from the non-key ones, i.e., the first attribute of R is the key of R and the first attribute of S is the key of S. Let Σ be the set consisting of these two key constraints. Fuxman and Miller [15] showed that the consistent answers of $\mathrm{q}_{\mathrm{path}}$ on ordinary databases are rewritable in first-order logic. Specifically, they showed that for every ordinary database $\mathfrak{D}$,

$$\textsc{Cons}(q, \Sigma, \mathfrak{D}) = 1 \quad \text{if and only if} \quad \mathfrak{D} \models \exists x \exists z' (R(x;z') \wedge \forall z (R(x;z) \rightarrow \exists y S(z;y))).$$

Thus, $\textsc{Cons}(q, \Sigma, \mathfrak{D})$ can be computed by a single evaluation of a first-order sentence on $\mathfrak{D}$ and without evaluating repeatedly $q(\mathfrak{D}')$, as $\mathfrak{D}'$ ranges over the potentially exponentially many repairs of $\mathfrak{D}$. Let $\mathbb{K} = (K, +, \times, 0, 1)$ be a naturally ordered positive semiring. In Section 4, we show that for every $\mathbb{K}$-database $\mathfrak{D}$, the following holds for the consistent answers $\mathrm{mCA}_{\mathbb{K}}(\mathrm{q}_{\mathrm{path}}, \Sigma, \mathfrak{D})$:

$$\mathrm{mCA}_{\mathbb{K}}(\mathrm{q}_{\mathrm{path}}, \mathfrak{D}) = \sum_{a \in D} \min_{b \in D : R^{\mathfrak{D}}(a,b) \neq 0} (R^{\mathfrak{D}}(a,b) \times \min_{c \in D : S^{\mathfrak{D}}(b,c) \neq 0} S^{\mathfrak{D}}(b,c)),$$

where D is the active domain of $\mathfrak{D}$. Thus, $\mathrm{mCA}_{\mathbb{K}}(\mathrm{q}_{\mathrm{path}}, \Sigma, \mathfrak{D})$ can be evaluated directly on $\mathfrak{D}$ and without considering the repairs of $\mathfrak{D}$.

Motivated by the above properties of $\mathrm{q}_{\mathrm{path}}$, we introduce the logic $\mathcal{L}_{\mathbb{K}}$, which is a variant of first-order logic with a minimization operator ∇ and a limited form of negation $(\overline{\text{Supp}})$ that flattens non-zero annotations to zero. We give rigorous semantics to the formulas of the logic $\mathcal{L}_{\mathbb{K}}$ on every naturally ordered positive semiring and then investigate when the consistent answers of conjunctive queries are rewritable in this logic.

Let q be a self-join free closed conjunctive query with one key constraint per relation. Our main result asserts that the consistent answers of q are rewritable in the logic $\mathcal{L}_{\mathbb{K}}$ if and only if the *attack* graph of q is acyclic. This result generalizes an analogous result of Koutris and Wijsen [26] for the Boolean semiring, but also yields new results for a multitude of semirings, including the bag semiring, the tropical semiring, and the fuzzy semiring. The notion of the attack graph was introduced by Wijsen [36, 37] and has played an important role in the study of the consistent answers of self-join free conjunctive queries on ordinary databases. Here, we leverage the insights obtained in this earlier study, but also obtain new insights that entail a further analysis of the properties of the attack graph when the query is evaluated under semiring semantics. As an illustration, it will turn out that the consistent answers $\mathrm{mCA}_{\mathbb{K}}(\mathrm{q}_{\mathrm{path}}, \Sigma, \mathfrak{D})$ of the query $\mathrm{q}_{\mathrm{path}}$ on $\mathfrak{D}$ are definable by the $\mathcal{L}_{\mathbb{K}}$-formula $\exists x \nabla_{R(x,y)} y.\, (R(x,y) \times \nabla_{S(y,z)} z.\, S(y,z))$.

Let q be a fixed self-join free query with one key per relation. Koutris and Wijsen [26] showed that if the attack graph of q contains a *strong* cycle, then computing the consistent answers $\textsc{Cons}(q, \Sigma, \mathfrak{D})$ of q on ordinary databases $\mathfrak{D}$ is a coNP-complete problem (this is a data complexity result, since q is fixed). Note that if $\mathbb{K}$ is a naturally ordered positive semiring, then computing $\mathrm{mCA}_{\mathbb{K}}(q, \mathfrak{D})$ on $\mathbb{K}$-databases $\mathfrak{D}$ is an optimization problem. Here, we focus on the bag semiring $\mathbb{N} = (N, +, \times, 0, 1)$ and show that if the attack graph of q contains a strong cycle, then computing $\mathrm{mCA}_{\mathbb{N}}(q, \Sigma, \mathfrak{D})$ on bag databases $\mathfrak{D}$ not only is a NP-hard problem but also it is NP-hard to even approximate $\mathrm{mCA}_{\mathbb{N}}(q, \Sigma, \mathfrak{D})$ with a constant relative approximation guarantee. This result paves the way to expand the study of the consistent answers of queries on annotated databases with methods and techniques from the rich theory of approximation algorithms for optimization problems.

2 Preliminaries

Semirings. A *commutative semiring* is an algebraic structure $\mathbb{K} = (K, +, \times, 0, 1)$ with $0 \neq 1$ and such that $(K, +, 0)$ and $(K, \times, 1)$ are commutative monoids, $\times$ distributes over $+$, and $0 \times a = a \times 0 = 0$ for every $a \in K$. A semiring has no *zero-divisors* if $a \times b = 0$ implies that $a = 0$ or $b = 0$. We say that $\mathbb{K}$ is *positive* if $a + b = 0$ implies $a = 0$ and $b = 0$, and also $\mathbb{K}$ has no zero-divisors. A semiring is *naturally ordered* if the canonical preorder $\leq_{\mathbb{K}}$, defined by $a \leq_{\mathbb{K}} b \iff \exists c\ (a + c = b)$, is a total order. Every non-empty finite set A of elements from a naturally ordered semiring has a minimum element, denoted by $\min(A)$. Unless specified otherwise, from now on we assume $\mathbb{K}$ to be a naturally ordered positive semiring. Examples of such semirings include the *Boolean* semiring $\mathbb{B} = (\{0, 1\}, \vee, \wedge, 0, 1)$, the *bag* semiring $\mathbb{N} = (N, +, \cdot, 0, 1)$ of the natural numbers, the *tropical* semiring $\mathbb{T} = ([0, \infty], \min, +, \infty, 0)$, the *Viterbi* semiring $\mathbb{V} = ([0, 1], \max, \times, 0, 1)$, and the *fuzzy* semiring $\mathbb{F} = ([0, 1], \max, \min, 0, 1)$. If $\mathbb{K}$ is the Boolean or the bag or the Viterbi or the fuzzy semiring, then $\leq_{\mathbb{K}}$ coincides with the standard order on the universe of $\mathbb{K}$, while if $\mathbb{K}$ is the tropical semiring, then $\leq_{\mathbb{K}}$ is the reverse of the standard order on the universe of $\mathbb{K}$.

$\mathbb{K}$-relations and $\mathbb{K}$-databases. A *(relational) schema τ* is a finite set of relation symbols each with a positive integer as its arity. We fix a countable set A of possible data values. An *n*-ary *$\mathbb{K}$-relation*, where $n > 0$, is a function $R\colon \mathrm{A}^n \to K$ such that $R(t) = 0$ for all but finitely many n-tuples of A^n. The *support* of R is defined as $\mathrm{Supp}(R) \coloneqq \{t \in \mathrm{A}^n \colon R(t) \neq 0\}$. If R and T are $\mathbb{K}$-relations of the same arity, we write $R \leq_{\mathbb{K}} T$ if $R(\vec{a}) \leq_{\mathbb{K}} T(\vec{a})$ for all $\vec{a}$ in the support of R. We write $R \subseteq T$, if $\mathrm{Supp}(R) \subseteq \mathrm{Supp}(T)$ and $R(\vec{a}) = T(\vec{a})$ for all $\vec{a}$ in the support of R. A *$\mathbb{K}$-database* $\mathfrak{D}$ over a schema τ is a collection of $\mathbb{K}$-relations, that is, a collection of functions $R_i : \mathrm{A}^n \to K$, such that the arities of R_i match that of the corresponding relation symbols in τ. We often write $R^{\mathfrak{D}}$ to denote the interpretation of the relation symbol R in $\mathfrak{D}$. The *support* of $\mathfrak{D}$, written $\mathrm{Supp}(\mathfrak{D})$, is the database that consists of the supports $\mathrm{Supp}(R)$ of the $\mathbb{K}$-relations R of $\mathfrak{D}$. For $\mathbb{K}$-databases $\mathfrak{D}$ and $\mathfrak{D}'$ of the same schema τ, we write $\mathfrak{D} \leq_{\mathbb{K}} \mathfrak{D}'$ ($\mathfrak{D} \subseteq \mathfrak{D}'$, resp.) if $R^{\mathfrak{D}} \leq_{\mathbb{K}} R^{\mathfrak{D}'}$ ($R^{\mathfrak{D}} \subseteq R^{\mathfrak{D}'}$, resp.) for every $R \in \tau$. The *active domain* of the $\mathbb{K}$-database $\mathfrak{D}$, denoted by D, is the set of all data values that occur in the support of some $\mathbb{K}$-relation of $\mathfrak{D}$. In this work, we only consider $\mathbb{K}$-databases with a non-empty active domain. This means that there is at least one relation R and a tuple t such that $R(t) \neq 0$.

Conjunctive Queries. We assume familiarity with basic definitions and notions related to first-order logic FO in the context of relational databases (e.g., see [1, 13]). We fix a countably infinite set Var of *first-order variables* $x, y, x_1, \ldots, x_n$ etc. and write $\vec{x}$ to denote a *tuple* of variables. We write $\mathrm{var}(\vec{x})$ to denote the set of variables that occur in $\vec{x}$. An *assignment* on a $\mathbb{K}$-database $\mathfrak{D}$ is a total function $\alpha\colon \mathrm{Var} \to D$. For a variable $x \in \mathrm{Var}$ and a value $a \in D$, we write $\alpha(a/x)$ to denote the assignment that maps x to a, and otherwise agrees with α.

A *conjunctive query* (CQ) is an FO-formula of the form $q(\vec{x}) \coloneqq \exists \vec{y}\big(R_1(\vec{z}_1) \wedge \ldots \wedge R_n(\vec{z}_n)\big)$, where each $R_i(\vec{z}_i)$ is a relational atom and each $\vec{z}_i$ is a tuple of variables in $\vec{x}$ and $\vec{y}$. We assume that all quantified variables of q occur in the quantifier-free part of the query. We say that a CQ q is *closed* if q has no free variables; otherwise, we say that q is *open*. We write $\hat{q}$ to denote the quantifier-free part of q. We call q *self-join-free* (sjfCQ) if no relation symbol occurs more than once in $\hat{q}$. If α is an assignment, we write $\alpha(\hat{q})$ for the set $\{R(\alpha(\vec{x})) \mid R(\vec{x}) \text{ is a conjunct of } \hat{q}\}$ of facts of $\hat{q}$ under α.

Following [19], if $q(\vec{x}) = \exists \vec{y}(R_1(\vec{z}_1) \wedge \ldots \wedge R_k(\vec{z}_k))$ is a CQ, $\mathfrak{D}$ is a $\mathbb{K}$-database, and α is an assignment on $\mathfrak{D}$, then, if $\sum$ and $\times$ are the (iterated) addition and multiplication of the semiring $\mathbb{K}$, the semantics of q on $\mathfrak{D}, \alpha$ is the semiring value

$$q(\mathfrak{D}, \alpha) \coloneqq \sum_{\vec{a} \in D^{|\vec{y}|}} R_1^{\mathfrak{D}}(\beta(\vec{z}_1)) \times \ldots \times R_k^{\mathfrak{D}}(\beta(\vec{z}_k)), \qquad \text{where } \beta \coloneqq \alpha(\vec{a}/\vec{y}).$$

3 Repairs and consistent answers under semiring semantics

Integrity constraints are semantic restrictions that the data of interest must obey. Integrity constraints on ordinary databases are typically expressed as sentences of first-order logic. In particular, this holds true for key constraints, the most widely used integrity constraints. A *key constraint* on a relation symbol R asserts that the values of some attributes of R determine the values of all other attributes of R. In what follows, we adopt the convention that the key attributes occupy the leftmost positions in the relation symbol R. We will write $R(\vec{x};\vec{y})$ to denote that the attributes in $\vec{x}$ form a key of R, and set $\mathrm{key}(R) \coloneqq \mathrm{var}(\vec{x})$. Clearly, every key constraint is expressible in FO. For example, if we have a ternary relation symbol $R(x_1, x_2; y)$, then the first-order sentence $\forall x_1 \forall x_2 \forall y \forall z (R(x_1, x_2, y) \wedge R(x_1, x_2, z) \rightarrow y = z)$ tells that the first two attributes of R form a key.

Let $\mathbb{K}$ be a semiring and let $\mathfrak{D}$ be a $\mathbb{K}$-database. What does it mean to say that $\mathfrak{D}$ satisfies a key constraint or, more generally, an integrity constraint ψ, where ψ is an FO-sentence? To answer this question, we have to give semiring semantics to FO. In the previous section, we already gave such semantics to the fragment of FO that expresses conjunctive queries, where the addition and the multiplication operations of $\mathbb{K}$ were used to interpret, respectively, existential quantification and conjunction. This approach extends naturally to the negation-free fragment of FO, but more care is needed to assign semiring semantics to arbitrary FO-formulas. As part of the study of provenance in databases, Grädel and Tannen [17] gave semiring semantics to FO-formulas in negation normal form NNF (i.e., all negation symbols are "pushed" to the atoms) by using the notion of an *interpretation*, which is a function that assigns semiring values to atomic or negated atomic facts. Here, we give semiring semantics to FO-formulas on a $\mathbb{K}$-database $\mathfrak{D}$ by, in effect, considering a particular *canonical* interpretation on $\mathfrak{D}$; a similar approach was adopted by Barlag et al. in [7] for $\mathbb{K}$-teams, which can be viewed as $\mathbb{K}$-databases over a schema with a single relation symbol. Appendix A contains a more detailed discussion of semiring semantics via interpretations.

Let $\mathbb{K}$ be a semiring, $\mathfrak{D}$ a $\mathbb{K}$-database, and $\alpha\colon \mathrm{Var} \longrightarrow D$ an assignment. If φ is an FO-formula in negation normal form, then we define the semiring value $\varphi(\vec{x})(\mathfrak{D}, \alpha)$ recursively as follows:

$$R(\vec{x})(\mathfrak{D}, \alpha) = R^{\mathfrak{D}}(\alpha(\vec{x}))$$

$$\neg R(\vec{x})(\mathfrak{D}, \alpha) = \begin{cases} 1 & \text{if } R(\vec{x})(\mathfrak{D}, \alpha) = 0 \\ 0 & \text{if } R(\vec{x})(\mathfrak{D}, \alpha) \neq 0 \end{cases} \qquad (x * y)(\mathfrak{D}, \alpha) = \begin{cases} 1 & \text{if } \alpha(\vec{x}) * \alpha(\vec{y}) \\ 0 & \text{otherwise} \end{cases} \text{ where } * \in \{=, \neq\}$$

$$(\varphi \wedge \psi)(\mathfrak{D}, \alpha) = \varphi(\mathfrak{D}, \alpha) \times \psi(\mathfrak{D}, \alpha) \qquad (\varphi \vee \psi)(\mathfrak{D}, \alpha) = \varphi(\mathfrak{D}, \alpha) + \psi(\mathfrak{D}, \alpha)$$

$$\forall x \varphi(\mathfrak{D}, \alpha) = \prod_{a \in D} \varphi(\mathfrak{D}, \alpha(a/x)) \qquad \exists x \varphi(\mathfrak{D}, \alpha) = \sum_{a \in D} \varphi(\mathfrak{D}, \alpha(a/x)).$$

It is straightforward to prove by induction that if α and β are assignments that agree on the free variables of an FO-formula φ, then $\varphi(\mathfrak{D}, \alpha) = \varphi(\mathfrak{D}, \beta)$, for every $\mathbb{K}$ database $\mathfrak{D}$. Thus, from now on, when we consider an FO-formula φ and an assignment α, we will assume that α is a function defined on the free variables on φ, hence α is a finite object. Furthermore, if φ is a sentence (i.e., φ has no free variables), then the semiring value $\varphi(\mathfrak{D}, \alpha)$ does not depend on the assignment α. In what follows, we write $\varphi(\mathfrak{D})$ to denote that value, if φ is an FO-sentence.

The following result is a consequence of Proposition 9 in [17].

Proposition 3.1. *If φ is an FO-formula in NNF, $\mathfrak{D}$ is a $\mathbb{K}$-database, and α is an assignment, then*

$$\varphi(\mathfrak{D}, \alpha) \neq 0 \quad \textit{if and only if} \quad \mathrm{Supp}(\mathfrak{D}), \alpha \models \varphi, \tag{1}$$

where the symbol $\models$ in the right-hand side refers to satisfaction in standard (set-based) FO. *In particular, if φ is an* FO*-sentence in NNF, then $\varphi(\mathfrak{D}) \neq 0$ if and only if* $\mathrm{Supp}(\mathfrak{D}) \models \varphi$.

Proposition 3.1 leads to the following natural definition for satisfaction over $\mathbb{K}$-databases.

Definition 3.2. Let φ be an FO-formula in NNF, $\mathfrak{D}$ a $\mathbb{K}$-database, and α an assignment. We say that $\mathfrak{D}, \alpha$ *satisfies* φ, denoted $\mathfrak{D}, \alpha \models_{\mathbb{K}} \varphi$, if $\varphi(\mathfrak{D}, \alpha) \neq 0$ (equivalently, $\mathrm{Supp}(\mathfrak{D}), \alpha \models \varphi$). In particular, if φ is an FO-sentence in NNF, then we write $\mathfrak{D} \models_{\mathbb{K}} \varphi$ if $\varphi(\mathfrak{D}) \neq 0$ (equivalently, $\mathrm{Supp}(\mathfrak{D}) \models \varphi$).

If Σ is a set of FO-sentences in NNF, we write $\mathfrak{D} \models_{\mathbb{K}} \Sigma$ to denote that $\mathfrak{D} \models_{\mathbb{K}} \varphi$ for every $\varphi \in \Sigma$.

With Definition 3.2 at hand, we are now ready to define the notion of a repair of a $\mathbb{K}$-database $\mathfrak{D}$ with respect to a set Σ of key constraints. Here, we will assume that every key constraint is defined by an FO-sentence in NNF in the standard way. For example, if $R(x_1, x_2; y)$ is a relation symbol in which the first two attributes form a key, then the key constraint expressing this property will be given by the FO-sentence $\forall x_1, x_2, y, z(\neg R(x_1, x_2, y) \vee \neg R(x_1, x_2, z) \vee y = z)$.

Recall also that if $\mathfrak{D}$ and $\mathfrak{D}'$ are two $\mathbb{K}$-databases, then $\mathfrak{D}' \subseteq \mathfrak{D}$ means that for every relation symbol R, the following two properties hold: (i) $\mathrm{Supp}(R^{\mathfrak{D}'}) \subseteq \mathrm{Supp}(R^{\mathfrak{D}})$; (ii) $R^{\mathfrak{D}'}(\vec{a}) = R^{\mathfrak{D}}(\vec{a})$, for every $\vec{a} \in \mathrm{Supp}(R^{\mathfrak{D}'})$. We will write $\mathfrak{D}' \subset \mathfrak{D}$ to denote that $\mathfrak{D}' \subseteq \mathfrak{D}$ and $\mathfrak{D}' \neq \mathfrak{D}$.

Definition 3.3. Let Σ be a set of key constraints, $\mathbb{K}$ a naturally ordered positive semiring, and $\mathfrak{D}$ a $\mathbb{K}$-database. A $\mathbb{K}$-database $\mathfrak{D}'$ is a *repair* of $\mathfrak{D}$ w.r.t. Σ if:

(1) $\mathfrak{D}' \models_{\mathbb{K}} \Sigma$ (which amounts to $\mathrm{Supp}(\mathfrak{D}') \models \varphi$, for every $\varphi \in \Sigma$);
(2) $\mathfrak{D}' \subseteq \mathfrak{D}$ and there is no $\mathbb{K}$-database $\mathfrak{D}''$ such that $\mathfrak{D}' \subset \mathfrak{D}'' \subseteq \mathfrak{D}$ and $\mathfrak{D}'' \models_{\mathbb{K}} \Sigma$.

We write $\mathrm{Rep}(\mathfrak{D}, \Sigma)$ for the collection of repairs of $\mathfrak{D}$ with respect to Σ.

Several remarks are in order now concerning the notion of repair in Definition 3.3. First, in the case of ordinary databases (i.e., $\mathbb{K}$ is the Boolean semiring $\mathbb{B}$), this notion coincides with the standard notion of a (subset) repair of an ordinary database with respect to a set of key constraints. Second, this notion is quite robust in the sense that we get the same notion if, instead of $\subseteq$, we had used the more relaxed notion $\leq_{\mathbb{K}}$ for comparing $\mathbb{K}$-databases. Finally, the notion can be extended by considering arbitrary FO-sentences as integrity constraints. Here, we gave the definition of a repair with respect to key constraints, since, in this paper, our entire focus is on such constraints.

Example 3.4. Let $\mathbb{N} = (N, +, \cdot, 0, 1)$ be the bag semiring and consider a schema consisting of a single ternary relation $R(x_1, x_2; y)$ in which the first two attributes form a key. Let $\mathfrak{D}$ be the $\mathbb{N}$-database with $R^{\mathfrak{D}}(a, b, c) = 2, R^{\mathfrak{D}}(a, b, d) = 3, R^{\mathfrak{D}}(a, a, a) = 4$, and $R^{\mathfrak{D}}(a', b', c') = 0$ for all other values (the values a, b, c, d are assumed to be distinct). Then $\mathfrak{D}$ has exactly two repairs $\mathfrak{D}_1$ and $\mathfrak{D}_2$ with respect the key constraint of R, where the non-zero values of $R^{\mathfrak{D}_1}$ are $R^{\mathfrak{D}_1}(a, b, c) = 2, R^{\mathfrak{D}_1}(a, a, a) = 4$, while the non-zero values of $R^{\mathfrak{D}_2}$ are $R^{\mathfrak{D}_2}(a, b, d) = 3, R^{\mathfrak{D}_2}(a, a, a) = 4$.

We are now ready to define the notion of consistent answers in the semiring setting.

Definition 3.5. Let $\mathbb{K}$ be a naturally ordered positive semiring, Σ a set of key constraints, φ an FO-formula, $\mathfrak{D}$ a $\mathbb{K}$-database, and α a variable assignment. The *consistent answers* $\mathrm{mCA}_{\mathbb{K}}(\varphi, \Sigma, \mathfrak{D}, \alpha)$ *of* φ *on* $\mathfrak{D}, \alpha$ *with respect to* Σ is defined as $\min_{\mathfrak{D}' \in \mathrm{Rep}(\mathfrak{D}, \Sigma)} \varphi(\mathfrak{D}', \alpha)$. If φ is an FO-sentence, then the *consistent answers of* φ *on* $\mathfrak{D}$ *with respect to* Σ is the value $\mathrm{mCA}_{\mathbb{K}}(\varphi, \Sigma, \mathfrak{D}) = \min_{\mathfrak{D}' \in \mathrm{Rep}(\mathfrak{D}, \Sigma)} \varphi(\mathfrak{D}')$.

The consistent answers provide the tightest lower bound on the values $\varphi(\mathfrak{D}', \alpha)$ as $\mathfrak{D}'$ ranges over all repairs of $\mathfrak{D}$. On the Boolean semiring $\mathbb{B}$, they coincide with the consistent answers.

Example 3.6. Let $\mathbb{V} = ([0, 1], \max, \times, 0, 1)$ be the Viterbi semiring. For $n \geq 2$, let $\mathfrak{D}$ be a $\mathbb{V}$-database with $\mathbb{V}$-relations $E_i(x; y)$, for $1 \leq i \leq n$, each encoding a simple directed edge-weighted graph in which the weight of every edge is a real number in $(0, 1]$. The weight of an edge can be thought of as the *confidence* of the edge. We define the *confidence of a path along* $E_1, E_2, \ldots, E_n$ to be the product of the confidences of its edges. Let us now consider the closed conjunctive query $q_n \coloneqq \exists x_0 \cdots \exists x_n \big(E_1(x_0; x_1) \wedge \cdots \wedge E_n(x_{n-1}; x_n)\big)$ and the set Σ of the key constraints of E_i, $1 \leq i \leq n$. Then, $\mathrm{mCA}_{\mathbb{V}}(q_n, \Sigma, \mathfrak{D}, \alpha)$ is a number $c \in [0, 1]$ that has the following two properties: (i) in every repair of $\mathfrak{D}$, there is a path along $E_1, E_2, \ldots, E_n$ of confidence at least c; (ii) there is a repair of $\mathfrak{D}$ in which the maximum confidence of a path along $E_1, \ldots, E_n$ is c.

Henceforth, we will focus on consistent answers for self-join-free CQs and key constraints.

4 Consistent answers and query rewriting

Data complexity is one of the most useful measures in studying query answering [34]. This stems from the observation that in practice queries are typically of small size (e.g., they are written by a user), while databases are of large size. In data complexity, the query is fixed and only the database is the input. Hence, each query q gives rise to a separate computational problem.

Definition 4.1. Let $\mathbb{K}$ be a naturally ordered positive semiring, q a CQ, and Σ a set of key constraints. We define the following function problem.

PROBLEM: $\mathrm{mCA}_{\mathbb{K}}(q, \Sigma)$
INPUT: A $\mathbb{K}$-database $\mathfrak{D}$ and an assignment α.
OUTPUT: The value of $\mathrm{mCA}_{\mathbb{K}}(q, \Sigma, \mathfrak{D}, \alpha)$.

If $\mathbb{K} = \mathbb{B}$ and q is a closed CQ, then $\mathrm{mCA}_{\mathbb{K}}(q, \Sigma)$ is a decision problem, which in the introduction we denoted by $\textsc{Cons}(q, \Sigma)$. We say that $\textsc{Cons}(q, \Sigma)$ is FO-*rewritable* if there is a $q' \in \mathsf{FO}$ such that $\textsc{Cons}(q, \Sigma, \mathfrak{D}) = q'(\mathfrak{D})$, for every $\mathfrak{D}$. The main benefit of an FO-rewriting of $\mathrm{mCA}_{\mathbb{B}}(q, \Sigma)$ is that such a rewriting reduces consistent query answering to query evaluation. Thus, a database engine developed for query evaluation can be directly used to compute the consistent answers. Note that having a query rewriting in some logic is trivial; for instance, one may internalize the process of checking all repairs using second-order logic. The real benefit of having an FO-rewriting is that the evaluation of FO-formulas, with respect to data complexity, is fast and highly parallelizable. In fact, FO-definable properties can be recognized by circuits of constant-depth and polynomial-size; more precisely, they lie in the circuit complexity class $\mathsf{DLOGTIME}$-uniform AC^0 [9]. Our goal is to identify a suitable notion of FO-rewritability for $\mathrm{mCA}_{\mathbb{K}}(q, \Sigma)$, where $\mathbb{K}$ is a naturally ordered positive semiring.

As a motivation, we first present an example of the rewriting of $\mathrm{mCA}_{\mathbb{K}}(q, \Sigma)$ in the semiring context for a particular query q. In Section 4.2, we introduce the logic $\mathcal{L}_{\mathbb{K}}$ that will be used to express rewritings. In Section 4.3, we show that the data complexity of $\mathcal{L}_{\mathbb{K}}$ lies in a semiring variant of uniform AC^0. Finally, in Section 4.4, we present our main result concerning the rewriting of $\mathrm{mCA}_{\mathbb{K}}(q, \Sigma)$ for sjfCQs and key constraints, one key for each relation in q. In what follows, we consider only sjfCQs and key constraints that can be read from the query; thus, we drop Σ from the notation and write simply $\mathrm{mCA}_{\mathbb{K}}(q)$, $\mathrm{mCA}_{\mathbb{K}}(q, \mathfrak{D}, \alpha)$, and $\mathrm{Rep}(\mathfrak{D})$. Recall that we also drop α, when we consider closed queries and when the proofs do no technically require an assignment.

4.1 Rewriting of the consistent answers of the path query

Recall that $\mathrm{q}_{\mathrm{path}} = \exists x \exists y \exists z (R(x; y) \land S(y; z))$, for which $\mathrm{mCA}_{\mathbb{B}}(\mathrm{q}_{\mathrm{path}})$ has the following FO-rewriting: $\exists x \exists z' (R(x, z') \land \forall z (R(x, z) \to \exists y S(z, y)))$ [15]. We want to obtain a similar expression in our setting for $\mathrm{q}_{\mathrm{path}}$, that is, an expression using semiring operations that provides the answer $\mathrm{mCA}_{\mathbb{K}}(\mathrm{q}_{\mathrm{path}}, \mathfrak{D})$ for $\mathbb{K}$-databases without having to evaluate $\mathrm{q}_{\mathrm{path}}$ on every repair of $\mathfrak{D}$. We define the expression $e_{path}(\mathfrak{D})$ as follows:

$$e_{path}(\mathfrak{D}) \coloneqq \sum_{a \in D} \min_{b \in D : R^{\mathfrak{D}}(a,b) \neq 0} (R^{\mathfrak{D}}(a, b) \times \min_{c \in D : S^{\mathfrak{D}}(b,c) \neq 0} S^{\mathfrak{D}}(b, c)), \tag{2}$$

where $\sum$, $\times$, and min refer to operations of a naturally ordered positive semiring $\mathbb{K}$. We call $R^{\mathfrak{D}}(a, b) \neq 0$ and $S^{\mathfrak{D}}(b, c) \neq 0$ *guards* of the minimisation operators. The min-operator in (2) may take an empty set as its range. This occurs specifically when there is no $b \in D$ such that $R^{\mathfrak{D}}(a, b) \neq 0$. In expressions of the above form, we adopt the convention that $\min(\emptyset) = 0$. We claim that

$$\mathrm{mCA}_{\mathbb{K}}(\mathrm{q}_{\mathrm{path}}, \mathfrak{D}) = e_{path}(\mathfrak{D}), \tag{3}$$

for every $\mathbb{K}$-database $\mathfrak{D}$. Next we show that the equality holds. Note that

$$\mathrm{mCA}_{\mathbb{K}}(\mathrm{q}_{\mathrm{path}}, \mathfrak{D}) = \min_{\mathfrak{D}' \in \mathrm{Rep}(\mathfrak{D})} \mathrm{q}_{\mathrm{path}}(\mathfrak{D}') = \min_{\mathfrak{D}' \in \mathrm{Rep}(\mathfrak{D})} \sum_{a,b,c \in D'} R^{\mathfrak{D}'}(a,b) \times S^{\mathfrak{D}'}(b,c),$$

where the former equality is the definition of $\mathrm{mCA}_{\mathbb{K}}$ and the latter follows from the semiring semantics for CQs. To show (3), it suffices to establish that the following two statements hold:

(i) For every repair $\mathfrak{D}'$ of $\mathfrak{D}$, we have that $e_{path}(\mathfrak{D}) \leq_{\mathbb{K}} \mathrm{q}_{\mathrm{path}}(\mathfrak{D}')$;
(ii) There is a repair $\mathfrak{D}^*$ of $\mathfrak{D}$ such that $e_{path}(\mathfrak{D}) = \mathrm{q}_{\mathrm{path}}(\mathfrak{D}^*)$.

The first statement is proved by inspecting the expression case-by-case, while the second is proved by constructing a suitable repair. The full proof is in Appendix B.1.

In order to construct a rewriting of $\mathrm{mCA}_{\mathbb{K}}(\mathrm{q}_{\mathrm{path}})$, we need a logical formula that evaluates to $e_{path}(\mathfrak{D})$, for every $\mathbb{K}$-database $\mathfrak{D}$. We first argue why existing rewritings for the Boolean semiring [15, 26] do not straightforwardly extend to arbitrary naturally ordered positive semirings. Intuitively, this is because expressions like (2) are built from sum ($\sum$), product ($\prod$), and min. While sum and product naturally correspond to existential and universal quantification, respectively, standard rewritings do not provide a natural equivalent for min. In fact, min is not needed in the Boolean semiring, since in expressions like (2), the guarded min-operator would always evaluate to 1 provided that the guards are supported, and to 0 otherwise. Unguarded min-operators, on the other hand, directly correspond to product on the Boolean semiring.

To formalize the preceding discussion, the first-order rewriting of Fuxman and Miller [15] and Koutris and Wijsen [26] for $\mathrm{mCA}_{\mathbb{B}}(\mathrm{q}_{\mathrm{path}})$, when expressed in negation normal form, is

$$\exists x \exists z' (R(x,z') \wedge \forall z (\neg R(x,z) \vee \exists y S(z,y))). \tag{4}$$

Under the semiring semantics of first-order logic, this rewriting would evaluate to the expression

$$\sum_{a \in D, b \in D} \Big(R^{\mathfrak{D}}(a,b) \times \prod_{c \in D} \big(\overline{\mathrm{Supp}}(R^{\mathfrak{D}}(a,c)) + \sum_{d \in D} S^{\mathfrak{D}}(c,d) \big) \Big), \tag{5}$$

where $\overline{\mathrm{Supp}}(k)$ is a function introduced in Section 4.2 that maps any non-zero semiring value k to 0, and 0 to 1. While this expression is correct for the Boolean semiring, as expected, it is incorrect for other semirings, such as the bag semiring, since it uses a product of semiring values instead of minimization. This raises the question of whether a correct rewriting is obtained by interpreting the universal quantifier as a minimization operator. Under this interpretation, the rewriting (4) evaluates to the expression

$$\sum_{a \in D, b \in D} \Big(R^{\mathfrak{D}}(a,b) \times \min_{c \in D} \big(\overline{\mathrm{Supp}}(R^{\mathfrak{D}}(a,c)) + \sum_{d \in D} S^{\mathfrak{D}}(c,d) \big) \Big).$$

To see why the latter expression is incorrect for the bag semiring, consider an $\mathbb{N}$-database $\mathfrak{D}$ with $R^{\mathfrak{D}}(a,b) = R^{\mathfrak{D}}(a,c) = 1$ and $S^{\mathfrak{D}}(b,d) = S^{\mathfrak{D}}(c,d) = 1$. Then $\mathrm{mCA}_{\mathbb{N}}(\mathrm{q}_{\mathrm{path}}, \mathfrak{D}) = 1$, but the expression evaluates to 2.

To obtain a rewriting whose semantics matches $e_{path}(\mathfrak{D})$, we need a logic capable of expressing the *guarded* minimization operators used in (2). A rewriting in such a logic could be expressed as follows:

$$\exists x \nabla_{R(x,y)} y.\ (R(x,y) \times \nabla_{S(y,z)} z.\ S(y,z)), \tag{6}$$

where $\nabla_{R(x,y)} y$ is a sort of minimization operator that utilizes a *guard*. In the next section, we introduce a logic $\mathcal{L}_{\mathbb{K}}$ that supports such formulas.

4.2 A logic for query rewriting over $\mathbb{K}$-databases

We have just exhibited an expression based on semiring operations for rewriting the consistent answers of the path query on $\mathbb{K}$-databases. Next we make this more formal, and introduce a general notion of query rewritability for $\mathbb{K}$-databases. For this purpose, we need to define a suitable logic.

As before, τ is a relational schema and $\mathbb{K}$ a naturally ordered positive semiring. The syntax of $\mathcal{L}_{\mathbb{K}}$ is given by the following grammar:

$$\varphi \coloneqq R(\vec{x}) \mid x = y \mid \varphi \land \varphi \mid \varphi \lor \varphi \mid \exists x\, \varphi \mid \nabla x \varphi(x) \mid \overline{\mathsf{Supp}}(\varphi),$$

where $R \in \tau$ of arity r, and $\vec{x} = (x_1, \ldots, x_r)$, for $x_1, \ldots, x_r, x, y \in \mathsf{Var}$.

Let $\mathfrak{D}$ be a $\mathbb{K}$-database and $\alpha\colon \mathsf{Var} \to D$ an assignment. The value $\varphi(\mathfrak{D}, \alpha)$ of a formula $\varphi \in \mathcal{L}_{\mathbb{K}}$ is defined recursively as follows. The cases for literals, Boolean connectives, and the existential quantifier are as the semiring semantics for FO (see page 5). The semantics of the new constructs is as follows:

$$\nabla x \varphi(x)(\mathfrak{D}, \alpha) = \min_{a \in D} \varphi(\mathfrak{D}, \alpha(a/x)) \qquad \overline{\mathsf{Supp}}(\varphi)(\mathfrak{D}, \alpha) = \begin{cases} 1 & \text{if } \varphi(\mathfrak{D}, \alpha) = 0 \\ 0 & \text{otherwise.} \end{cases}$$

Over the Boolean semiring, the semantics of the minimization operator ∇x coincides with the universal quantifier and $\overline{\mathsf{Supp}}(\varphi)$ corresponds to negation. In general, the $\overline{\mathsf{Supp}}$ operator serves as a (weak) negation operator since it flattens every non-zero annotation to return 0, thus in a sense losing all the shades of information provided by non-zero annotations. We use the shorthand $\mathsf{Supp}(\varphi)$ for $\overline{\mathsf{Supp}}(\overline{\mathsf{Supp}}(\varphi))$, which flattens all non-zero weights to 1.

Remark 4.1. $\mathcal{L}_{\mathbb{B}}$ is essentially FO; furthermore, in $\mathcal{L}_{\mathbb{B}}$, ∇x is $\forall x$ and $\overline{\mathsf{Supp}}(\varphi)$ is $\neg\varphi$.

This observation can be extended to the result that FO embeds into $\mathcal{L}_{\mathbb{K}}$, for every naturally ordered positive semiring $\mathbb{K}$, in the following sense. The translation between the logics is obtained by identifying ∇x with $\forall x$ and $\overline{\mathsf{Supp}}(\varphi)$ with $\neg\varphi$.

Proposition 4.2. *Let $\mathbb{K}$ be a naturally ordered positive semiring. For every $\varphi \in \mathcal{L}_{\mathbb{K}}$, there exists $\psi \in$ FO such that $\varphi(\mathfrak{D}, \alpha) \neq 0$ if and only if $\mathsf{Supp}(\mathfrak{D}), \alpha \models \psi$, for every $\mathbb{K}$-database $\mathfrak{D}$ and every assignment α. Conversely, for every $\psi \in$ FO, there exists $\varphi \in \mathcal{L}_{\mathbb{K}}$ such that the "if and only if" holds. If the annotations of $\mathfrak{D}$ are from the set $\{0, 1\}$, then $\varphi(\mathfrak{D}, \alpha) \neq 0$ if and only if $\mathfrak{D}, \alpha \models \psi$.*

The formula $\nabla x.\, \varphi(x)$ computes the minimum value of $\varphi(a/x)$, where a ranges over the active domain of the database. However, sometimes we want a to range over the support of some definable predicate, that we call *guard* and write G. In general, the guard is an $\mathcal{L}_{\mathbb{K}}$-formula $G(\vec{y}, z)$, and typically it has free variables from φ. For this purpose, we define the following shorthand

$$\nabla_G z.\, \varphi(\vec{y}, z) \coloneqq \nabla z. \theta(\vec{y}, z), \text{ where } \theta(\vec{y}, z) \coloneqq \Big(\big(\overline{\mathsf{Supp}}(G(\vec{y}, z)) \land \exists z' \varphi(\vec{y}, z') \land \chi \big) \lor \big(\varphi(\vec{y}, z) \land \chi \big) \Big), \quad (7)$$

where $G, \varphi \in \mathcal{L}_{\mathbb{K}}$ and $\chi \coloneqq \mathsf{Supp}(\exists z' G(\vec{y}, z'))$. The idea of (7) is as follows. We first explain $\theta(\vec{y}, z)$. Informally, if the subformula $\exists z' G(\vec{y}, z')$ of χ is not supported (i.e., $G(\vec{y}, z')$ evaluates to 0 for every z'), then $\theta(\vec{y}, z)$ evaluates to 0. The more interesting case is where $\exists z' G(\vec{y}, z')$ is supported. In that case, $\theta(\vec{y}, z)$ returns the value of $\varphi(\vec{y}, z)$ if $G(\vec{y}, z)$ is supported; otherwise it returns the sum of all values $\varphi(\vec{y}, z')$, where z' ranges over the active domain. It is then evident that $\varphi(\vec{y}, z)$ is minimized for a z such that $G(\vec{y}, z)$ is supported (if such a z exists).

It is now straightforward to check that the following holds:

Proposition 4.3. *If G and φ are $\mathcal{L}_{\mathbb{K}}$-formulas, $\mathfrak{D}$ is a $\mathbb{K}$-database, and α is an assignment, we have that $\nabla_G x. \varphi(x)(\mathfrak{D}, \alpha) = \min_{a \in D : G(\mathfrak{D}, \alpha(a/x)) \neq 0} \varphi(\mathfrak{D}, \alpha(a/x))$.*

Next, we define what rewritable means in our setting:

Definition 4.4. A *semiring rewriting* of $\mathrm{mCA}_{\mathbb{K}}(q)$ is an $\mathcal{L}_{\mathbb{K}}$-formula φ_q such that for every $\mathbb{K}$-database $\mathfrak{D}$ and assignment α, we have that $\mathrm{mCA}_{\mathbb{K}}(q, \mathfrak{D}, \alpha) = \varphi_q(\mathfrak{D}, \alpha)$. We say that q is $\mathcal{L}_{\mathbb{K}}$-*rewritable* if there exists a semiring rewriting of q.

Example 4.5. Consider $\mathrm{q}_{\mathrm{path}}$ and the expression $e_{path}(\mathfrak{D})$ defined in Section 4.1. Observe that $\mathrm{mCA}_{\mathbb{K}}(\mathrm{q}_{\mathrm{path}})$ is $\mathcal{L}_{\mathbb{K}}$-rewritable, since the $\mathcal{L}_{\mathbb{K}}$-formula $\exists x \nabla_{R(x,y)} y.\ (R(x,y) \times \nabla_{S(y,z)} z.\ S(y,z))$ evaluates to $e_{path}(\mathfrak{D})$, over any $\mathbb{K}$-database $\mathfrak{D}$.

When $\mathbb{K} = \mathbb{B}$ (i.e., over the Boolean semiring $\mathbb{B}$), the guarded minimization operators in the latter formula can be eliminated using (7), leading to a formula whose interpretation matches (5) and thus coincides with existing rewritings [15, 26]. However, this elimination is not possible for semirings in general.

4.3 $\mathbb{K}$-circuits and complexity theory

Our goal is to use $\mathcal{L}_{\mathbb{K}}$ as a logic for rewritings of $\mathrm{mCA}_{\mathbb{K}}(q, \Sigma)$. Note that, from Proposition 4.2, it follows directly that any general rewriting result for our logic embeds the rewriting of Koutris and Wijsen for ordinary databases. We need to establish that $\mathcal{L}_{\mathbb{K}}$ maintains some of the benefits of FO mentioned in the beginning of Section 4. In particular, the data complexity of FO is in DLOGTIME-uniform AC^0 (and hence in P). Arguably, a rewriting of $\mathrm{mCA}_{\mathbb{K}}(q)$ in $\mathcal{L}_{\mathbb{K}}$ retains the conceptual benefit of query rewriting: instead of computing answers of q for each repair, one may compute the answers of the rewriting directly on the inconsistent database. In order to argue about the complexity of $\mathcal{L}_{\mathbb{K}}$, and hence about the suitability of the logic for rewriting, we need to introduce a generalization of AC^0 to semirings. For this purpose, we define a variant of semi-unbounded fan-in arithmetic AC^0. We emphasize that, in general, an input to $\mathrm{mCA}_{\mathbb{K}}(q, \Sigma)$ is a $\mathbb{K}$-database, where K could be infinite (such as the real numbers), and hence a suitable model of computation has to be able to deal with arithmetics over an arbitrary semiring.

Next, we give the definitions concerning circuits that are needed for our purpose. We refer the reader to the book [35] for a thorough introduction to circuit complexity and to the book [21] for an exposition of tropical circuits as a tool for studying discrete optimization problems via dynamic programming. $\mathbb{K}$-circuits are a model of computation for computing semiring-valued functions.

Definition 4.6. Let $\mathbb{K}$ be a naturally ordered positive semiring. A $\mathbb{K}$-*circuit with min* is a finite simple directed acyclic graph of labeled nodes, also called *gates*, such that

- there are gates labeled *input*, each of which has indegree 0,
- there are gates labeled *constant*, with indegree 0 and labeled with a $c \in K$,
- there are gates labeled *addition*, *multiplication*, *min*, and $\overline{\mathrm{Supp}}$,
- exactly one gate of outdegree 0 is additionally labeled *output*.

Additionally, the input gates are ordered. Note that addition, multiplication, and min gates can have arbitrary in-degree, and the $\overline{\mathrm{Supp}}$-gates have in-degree 1. The *depth* of a circuit C is the length of the longest path from an input gate to an output gate in C, while the *size* of C is the number of gates in C.

A circuit C of this kind, with n input gates, computes the function $f_C \colon K^n \longrightarrow K$ as follows: First, the input to the circuit is placed in the input gates. Then, in each step, each gate whose predecessor gates all have a value, computes the respective function it is labeled with, using the values of its predecessors as inputs. The output of f_C is then the value of the output gate after the computation.

Each $\mathbb{K}$-circuit computes a function with a fixed number of arguments; for this reason, we consider families $(C_n)_{n\in\mathbb{N}}$ of $\mathbb{K}$-circuits, where each circuit C_n has exactly n input gates. A family $C = (C_n)_{n\in\mathbb{N}}$ of $\mathbb{K}$-circuits computes the function $f_C \colon K^* \longrightarrow K$ defined as $f_C(\vec{x}) \coloneqq f_{C_{|\vec{x}|}}(\vec{x})$.

Remark 4.2. A frequent requirement for considering a circuit family $(C_n)_{n\in\mathbb{N}}$ as an algorithm is the existence of an algorithm that given n outputs the circuit C_n. Circuit families for which such an algorithm exists are called *uniform.* The data complexity of FO is DLOGTIME-uniform AC^0 [9], and hence there is a DLOGTIME algorithm that describes C_n, given n. More formally, the algorithm takes two numbers (i, j) as an input and outputs the type of the ith gate of C_n, the index of its jth predecessor, and, if the gate is an input gate, the index of the input string it corresponds to. See [35] for details on circuit uniformity and [10] for uniformity in the context of real computation.

Definition 4.7. For a naturally ordered positive semiring $\mathbb{K}$, let $\mathsf{AC}^0_{\mathbb{K}}(+,\times,\min,\overline{\mathrm{Supp}})$ consist of all families $(C_n)_{n\in\mathbb{N}}$ of $\mathbb{K}$-circuits as defined in Definition 4.6 that are of constant depth and of polynomial size. This means that there is some constant $k \in \mathbb{N}$ and a univariate polynomial function $f\colon \mathbb{N} \to \mathbb{N}$ such that, for each $i \in \mathbb{N}$, the circuit C_i is of depth k and has at most $f(n)$ many gates. We write $\mathsf{AC}^0_{\mathbb{K}}(+,\times_2,\min,\overline{\mathrm{Supp}})$ for the restriction of $\mathsf{AC}^0_{\mathbb{K}}(+,\times,\min,\overline{\mathrm{Supp}})$ in which all multiplication gates have in-degree 2. We write $\mathsf{FnAC}^0_{\mathbb{K}}(+,\times,\min,\overline{\mathrm{Supp}})$ and $\mathsf{FnAC}^0_{\mathbb{K}}(+,\times_2,\min,\overline{\mathrm{Supp}})$, for the class of functions computed by $\mathsf{AC}^0_{\mathbb{K}}(+,\times,\max,\overline{\mathrm{Supp}})$ and $\mathsf{AC}^0_{\mathbb{K}}(+,\times_2,\min,\overline{\mathrm{Supp}})$ circuits, respectively.

Note that, for the Boolean semiring, the classes $\mathsf{AC}^0_{\mathbb{B}}(+,\times,\min,\overline{\mathrm{Supp}})$, $\mathsf{AC}^0_{\mathbb{B}}(+,\times_2,\min,\overline{\mathrm{Supp}})$, and AC^0 coincide, for in this case × and min correspond to $\wedge$-gates, + corresponds to $\vee$-gates, and $\overline{\mathrm{Supp}}$ corresponds to not-gates. Hence, by a classical result by Immerman relating FO and AC^0, the functions in DLOGTIME-uniform $\mathsf{FnAC}^0_{\mathbb{B}}(+,\times_2,\min,\overline{\mathrm{Supp}})$ are precisely those that can be defined in $\mathcal{L}_{\mathbb{B}}$.

The proof of the next result is in Appendix B.2.

Proposition 4.8. *For every naturally ordered positive semiring $\mathbb{K}$, the data complexity of $\mathcal{L}_{\mathbb{K}}$ is in* DLOGTIME*-uniform* $\mathsf{FnAC}^0_{\mathbb{K}}(+,\times_2,\min,\overline{\mathrm{Supp}})$.

If we can argue that functions in $\mathsf{FnAC}^0_{\mathbb{K}}(+,\times_2,\min,\overline{\mathrm{Supp}})$ are in some sense simple, we have grounds for $\mathcal{L}_{\mathbb{K}}$ being a “good” logic for rewritings. Consider an $\mathsf{AC}^0_{\mathbb{K}}(+,\times_2,\min,\overline{\mathrm{Supp}})$ circuit family $(C_n)_{n\in\mathbb{N}}$ with depth k and $p(n)$ many gates, for some polynomial function p. The computation of an $\mathsf{AC}^0_{\mathbb{K}}(+,\times_2,\min,\overline{\mathrm{Supp}})$-circuit differs from a computation of an $\mathsf{AC}^0_{\mathbb{K}}(+,\times_2)$ circuit only in the additional minimization and $\overline{\mathrm{Supp}}$ gates. In its computation, the circuit C_n needs to evaluate at most $p(n)$ minimization and $\overline{\mathrm{Supp}}$ gates. To evaluate a single minimization gate, it suffices to make at most $p(n)$-many comparisons between semiring values to find the smallest input to the gate. Similarly, to evaluate a $\overline{\mathrm{Supp}}$-gate, it suffices to make one $a = 0$-comparison for a semiring value a. Hence, in comparison to $\mathsf{AC}^0_{\mathbb{K}}(+,\times_2)$ circuits, the evaluation of a $\mathsf{AC}^0_{\mathbb{K}}(+,\times_2,\min,\overline{\mathrm{Supp}})$ circuit needs to make additionally polynomially many comparisons between two semiring values. Therefore, if $\mathsf{AC}^0_{\mathbb{K}}(+,\times_2)$ is computationally favorable and size comparisons between semiring values can be made efficiently, then we have an argument that $\mathsf{AC}^0_{\mathbb{K}}(+,\times_2,\min,\overline{\mathrm{Supp}})$ is computationally favorable as well. It is easy to see that $\mathsf{AC}^0_{\mathbb{K}}(+,\times_2)$ circuit families compute polynomial functions of constant degree, and hence its computation is in a strong sense polynomial. Thus, functions in $\mathsf{FnAC}^0_{\mathbb{K}}(+,\times_2,\min,\overline{\mathrm{Supp}})$ are in a strong sense polynomial.

4.4 Acyclicity of the attack graph and semiring rewriting of consistent answers

We will next establish a necessary and sufficient condition for $\mathrm{mCA}_{\mathbb{K}}(q,\Sigma)$ to be rewritable in $\mathcal{L}_{\mathbb{K}}$, when q is a self-join free conjunctive query and Σ is a set of key constraints, one for each relation in q. To this effect, we consider the concepts of an attack and of an attack graph, two concepts that were introduced by Wijsen [36, 37] and also used in [2, 26]. Our main result asserts that the attack graph of q (see Definition 4.11) is acyclic if and only if $\mathrm{mCA}_{\mathbb{K}}(q,\Sigma)$ is $\mathcal{L}_{\mathbb{K}}$-rewritable.

Theorem 4.9. *Let $\mathbb{K}$ be a naturally ordered positive semiring, q be a self-join free conjunctive query, and Σ a set of key constraints, one for each relation in q. The attack graph of q is acyclic if and only if* $\mathrm{mCA}_{\mathbb{K}}(q,\Sigma)$ *is $\mathcal{L}_{\mathbb{K}}$-rewritable.*

The proof of the theorem is divided into two parts. We first establish the easier right-to-left direction (Proposition 4.12) before building up to prove the more involved left-to-right direction.

We write $\mathsf{var}(R_i)$ and $\mathsf{var}(q)$ to denote the sets of variables that occur in R_i and q, respectively. We write $q[x]$ to denote the CQ obtained from q by removing the quantifier $\exists x$, if there is one. If $R(\vec{y};\vec{z})$ is an atom of q, we write $q \setminus R$ for the query resulting by removing that atom from q, as well as the existential quantifiers binding the variables that appear only in the atom $R(\vec{y};\vec{z})$. We say that an atom $R(\vec{y};\vec{z})$ *induces* the functional dependency $\mathsf{key}(R) \longrightarrow \mathsf{var}(R)$ (or $\mathsf{var}(\vec{y}) \longrightarrow \mathsf{var}(R)$). If q is a sjfCQ, we define $\Sigma(q)$ to be the set of functional dependencies *induced* by the atoms in q. Formally,

$$\Sigma(q) \coloneqq \{\mathsf{key}(R) \longrightarrow \mathsf{var}(R) \mid R \in q\}.$$

The *closure* of a set X of variables with respect to a set Λ of functional dependencies contains all the variables $y \in \mathsf{var}(q)$ such that $\Lambda \models X \longrightarrow y$. Here, $\Lambda \models X \longrightarrow y$ means that the set of functional dependencies in Λ semantically entails the functional dependency $X \longrightarrow y$. Syntactically, this entailment can be derived using Armstrong's axioms for functional dependencies [5]. For $R(\vec{y};\vec{z}) \in q$, we write $(\mathsf{var}(\vec{y}))^+_{\Sigma(q\setminus R)}$ for the *closure of* $\mathsf{var}(\vec{y})$ *with respect to* $\Sigma(q \setminus R)$, that is

$$(\mathsf{var}(\vec{y}))^+_{\Sigma(q\setminus R)} \coloneqq \{x \in \mathsf{var}(q) \mid \Sigma(q \setminus R\}) \models \mathsf{var}(\vec{y}) \longrightarrow x\}.$$

Definition 4.10. Let q be a sjfCQ. An atom $R(\vec{y};\vec{z})$ of q *attacks a variable* x bound in q if there exists a non-empty sequence of variables $x_1,\dots,x_n$ bound in q, such that:

(1) $x_1 \in \mathsf{var}(\vec{z})$ (i.e., is a non-key variable of R), and $x_n = x$;
(2) for every $i < n$, we have that x_i, x_{i+1} occur together in some atom of q; and
(3) for every $i \leq n$, we have that $x_i \notin (\mathsf{var}(\vec{y}))^+_{\Sigma(q\setminus R)}$.

A variable x in q is *unattacked in* q if no atom of q attacks it.

Definition 4.11 (Attack graph). The *attack graph* of q is the graph defined as follows: (i) the vertices are the atoms of q; (ii) there is a directed edge from $R(\vec{y};\vec{z})$ to a different atom $R'(\vec{y'};\vec{z'})$ if $R(\vec{y};\vec{z})$ attacks a variable in $\mathsf{var}(\vec{y'})$, i.e., if $R(\vec{y};\vec{z})$ attacks a key variable of R' that is bound in q.

Our definition of an attack graph is phrased slightly differently but remains equivalent to the definitions found in [2, 26]. Specifically, note that if an atom attacks a bound variable in some other atom R, then it necessarily also attacks a bound variable that occurs in $\mathsf{key}(R)$.

The next proposition establishes the right-to-left direction of Theorem 4.9.

Proposition 4.12. *Let $\mathbb{K}$ be a naturally ordered positive semiring, q be a self-join free conjunctive query, and Σ a set of key constraints, one for each relation in q. If* $\mathrm{mCA}_{\mathbb{K}}(q,\Sigma)$ *is $\mathcal{L}_{\mathbb{K}}$-rewritable, then the attack graph of q is acyclic.*

Proof. Assume that φ is $\mathcal{L}_{\mathbb{K}}$-rewriting of $\mathrm{mCA}_{\mathbb{K}}(q,\Sigma)$. Let ψ be the corresponding FO-formula obtained from Proposition 4.2. It is easy to see that ψ is an FO-rewriting of $\textsc{Cons}(q,\Sigma)$ on ordinary databases. From the results in [26], it follows that the attack graph of q is acyclic. □

Towards proving the converse direction of Theorem 4.9, we start with the next lemma, whose proof can be found in Appendix B.3:

Lemma 4.13. *If q is a* sjfCQ *with an acyclic attack graph and x is an unattacked variable, then $q[x]$ has an acyclic attack graph. Moreover, the attack graph of $q[x]$ is a subgraph of the attack graph of q.*

Lemma 4.14. *Let q be a* sjfCQ *and Σ be a set of key constraints, one key per relation. Let $\mathfrak{D}$ be a $\mathbb{K}$-database and γ an assignment such that $\mathfrak{D}', \gamma \models_{\mathbb{K}} q$ for every repair $\mathfrak{D}'$ of $\mathfrak{D}$ (with respect to Σ), and let x be an unattacked variable that is bound in q. Then, there is an element $c \in D$ such that $\mathfrak{D}', \gamma(c/x) \models_{\mathbb{K}} q[x]$, for every repair $\mathfrak{D}'$ of $\mathfrak{D}$.*

Proof. Let $\mathfrak{D}$ and γ be as described in the statement of the lemma. For every repair $\mathfrak{D}'$ of $\mathfrak{D}$, let $\rho(\mathfrak{D}') = \{c \in D \mid \mathfrak{D}', \gamma(c/x) \models_{\mathbb{K}} q[x]\}$. This set is always nonempty, since by assumption, $\mathfrak{D}', \gamma \models_{\mathbb{K}} q$ for every repair $\mathfrak{D}'$ of $\mathfrak{D}$. We will prove the following stronger formulation of the lemma:

***Claim* 1.** *There is a repair $\mathfrak{R}^*$ of $\mathfrak{D}$ such that $\rho(\mathfrak{R}^*) = \bigcap_{\mathfrak{D}' \text{ repair of } \mathfrak{D}} \rho(\mathfrak{D}') \neq \emptyset$.*

Let $\mathfrak{R}^*$ be a repair of $\mathfrak{D}$ for which $\rho(\mathfrak{R}^*)$ is minimal according to subset inclusion. We will show that $\rho(\mathfrak{R}^*) \subseteq \rho(\mathfrak{S})$, for every repair $\mathfrak{S}$ of $\mathfrak{D}$. To this end, fix an arbitrary repair $\mathfrak{S}$ of $\mathfrak{D}$, and let $\mathfrak{R}$ be a repair such that $\rho(\mathfrak{R}) = \rho(\mathfrak{R}^*)$ and, in addition, there is no other repair $\mathfrak{T}$ of $\mathfrak{D}$ such that $\rho(\mathfrak{R}^*) = \rho(\mathfrak{T})$ and $\mathfrak{T} \cap \mathfrak{S} \supsetneq \mathfrak{R} \cap \mathfrak{S}$. We will show that $\rho(\mathfrak{R}) \subseteq \rho(\mathfrak{S})$. Take $a \in \rho(\mathfrak{R})$. We want to show that $a \in \rho(\mathfrak{S})$. Towards a contradiction, assume $a \notin \rho(\mathfrak{S})$. Hence, $\mathfrak{R}, \gamma(a/x) \models_{\mathbb{K}} q[x]$ and $\mathfrak{S}, \gamma(a/x) \not\models_{\mathbb{K}} q[x]$. Let α be a valuation such that $\mathfrak{R}, \alpha \models_{\mathbb{K}} \hat{q}$ and agrees with $\gamma(a/x)$ with respect to the free variables of $q[x]$. Thus, there is a fact $A \in \alpha(\hat{q})$ such that $A \in \mathfrak{R}$ and $A \notin \mathfrak{S}$. Since $\mathfrak{S}$ is a repair of $\mathfrak{D}$ it contains a fact A' that is key-equal to A. Now, consider $\mathfrak{R}' = (\mathfrak{R} \setminus \{A\}) \cup \{A'\}$. Note that, $\mathfrak{R}'$ is a repair of $\mathfrak{D}$, for we are considering only key constraints with one key per relation and sjfCQs. Since $\mathfrak{R}' \cap \mathfrak{S} \supsetneq \mathfrak{R} \cap \mathfrak{S}$, it follows from the choice of $\mathfrak{R}$ that $\rho(\mathfrak{R}') \neq \rho(\mathfrak{R})$. We will show the following claim, which will then lead to a contradiction.

***Claim* 2.** $\rho(\mathfrak{R}') \subsetneq \rho(\mathfrak{R})$

Proof. Since $\rho(\mathfrak{R}') \neq \rho(\mathfrak{R})$, we only need to show containment. Pick an arbitrary $b \in \rho(\mathfrak{R}')$, and notice that $\mathfrak{R}', \gamma(b/x) \models_{\mathbb{K}} q[x]$. Let β be a valuation such that $\mathfrak{R}', \beta \models_{\mathbb{K}} \hat{q}$ and agrees with $\gamma(b/x)$ with respect to the free variables of $q[x]$. The claim follows if $\mathfrak{R}, \beta \models_{\mathbb{K}} q[x]$, and thus assume $\mathfrak{R}', \beta \not\models_{\mathbb{K}} q[x]$. Since $\mathfrak{R}$ and $\mathfrak{R}'$ contain the same facts with the exception of A and A', then necessarily $A' \in \beta(\hat{q})$. Let $R(\vec{y}; \vec{z})$ be the atom in q with the same relation name as A and A', and let δ be the valuation for the variables in q defined as follows:

$$\delta(v) = \begin{cases} \alpha(v) & \text{if } R(\vec{y}; \vec{z}) \text{ attacks } v \text{ in } q \\ \beta(v) & \text{otherwise} \end{cases}$$

Since x is unattacked, it is not attacked by R, and thus $\delta(x) = \beta(x) = b$. Hence, to prove that $b \in \rho(\mathfrak{R})$ it suffices to show that $\mathfrak{R}, \delta \models_{\mathbb{K}} \hat{q}$. Recall that A and A' are key-equal facts of relation name R such that $A \in \alpha(\hat{q})$ and $A' \in \beta(\hat{q})$. Hence $\alpha(\vec{y}) = \beta(\vec{y})$ for the key variables $\vec{y}$ of R.

Let $\Sigma(q \setminus R)$, and let $(\mathsf{var}(\vec{y}))^+_{\Sigma(q \setminus R)}$. Notice that, if we define

$$\begin{aligned} Y_0 &:= \mathsf{var}(\vec{y}) \\ Y_{n+1} &:= \{w \in \mathsf{Var} \mid \text{ there is } (V \rightarrow W) \in \Sigma(q \setminus R) \text{ such that } V \subseteq Y_n \text{ and } w \in W\} \end{aligned}$$

then $(\mathsf{var}(\vec{y}))^+_{\Sigma(q \setminus R)} = \bigcup_n Y_n$.

Clearly, $\mathsf{var}(\vec{y}) \subseteq (\mathsf{var}(\vec{y}))^+_{\Sigma(q \setminus R)}$. We now show that α and β agree on all variables in $(\mathsf{var}(\vec{y}))^+_{\Sigma(q \setminus R)}$. More precisely, we show that $\alpha(w) = \beta(w)$ for every $w \in (\mathsf{var}(\vec{y}))^+_{\Sigma(q \setminus R)} = \bigcup_n Y_n$ by induction on n. For Y_0 this follows, for we already established that $\alpha(\vec{y}) = \beta(\vec{y})$ for the key variables of R. Now assume $\alpha(Y_n) = \beta(Y_n)$, we want to show that $\alpha(Y_{n+1}) = \beta(Y_{n+1})$. Let $w \in Y_{n+1} \setminus Y_n$, then there is a functional dependency $V \rightarrow W \in \Sigma(q \setminus R)$ such that $V \subseteq Y_n$ and $w \in W$. This dependency is of the form $\vec{y'} \rightarrow \vec{z'}$ for some atom $S(\vec{y'}; \vec{z'}) \neq R(\vec{y}; \vec{z})$, where $\vec{y'} \subseteq V \subseteq Y_n$ and $\vec{z'} \subseteq W$. By inductive hypothesis, $\alpha(\vec{y'}) = \beta(\vec{y'})$. Furthermore, $\mathfrak{R}$ and $\mathfrak{R}'$ agree on every atom with the exception of A and A' which are the facts corresponding to the relation R, thus $\mathfrak{R}$ and $\mathfrak{R}'$ agree on $S(\vec{y'};\vec{z'})$ and this shows that α and β agree on all variables in Y_{n+1}. It follows by the induction principle that α and β agree on all variables in $(\mathsf{var}(\vec{y}))^+_{\Sigma(q\setminus R)}$.

Notice now that for every non-key variable z of R, if $z \notin (\mathsf{var}(\vec{y}))^+_{\Sigma(q\setminus R)}$ then $R(\vec{y};\vec{z})$ attacks z, since in that case the sequence of length one consisting only of z witnesses the attack, and consequently $\delta(z) = \alpha(z)$. Furthermore, since α and β agree on variables in $(\mathsf{var}(\vec{y}))^+_{\Sigma(q\setminus R)}$, δ maps the atom $R(\vec{y};\vec{z})$ to the fact $R(\alpha(\vec{y});\alpha(\vec{z})) = A$, which belongs to $\mathfrak{R}$.

Let $S(\vec{y'};\vec{z'})$ be an arbitrary atom of $q \setminus R$. Since $\mathfrak{R}, \alpha \models_{\mathbb{K}} \hat{q}$ and $\mathfrak{R}', \beta \models_{\mathbb{K}} \hat{q}$ there are facts $B \in \mathfrak{R}$ and $B' \in \mathfrak{R}'$ such that α and β map $S(\vec{y'};\vec{z'})$ to B and B', respectively. Since $\mathfrak{R}$ and $\mathfrak{R}'$ contain the same facts with the exception of A and A', we have that $B, B' \in \mathfrak{R}$. Hence, it suffices to show that δ maps $S(\vec{y'};\vec{z'})$ to either B or B'.

If all variables in $S(\vec{y'};\vec{z'})$ are attacked (not attacked, resp.) by $R(\vec{y};\vec{z})$ then, by definition, δ coincides with α (β, resp.) with respect to the variables in S and hence maps $S(\vec{y'};\vec{z'})$ to B (B', resp.).

If we are not in the above case there are both variables that are attacked and not attacked by $R(\vec{y};\vec{z})$. Let x_1 and x_2 be arbitrary variables that occur in $S(\vec{y'};\vec{z'})$ such that $R(\vec{y};\vec{z})$ attacks x_1 but does not attack x_2. Towards a contradiction suppose $x_2 \notin (\mathsf{var}(\vec{y}))^+_{\Sigma(q\setminus R)}$. Since R attacks x_1, there is a sequence $v_1, \dots, v_k$ of variables starting in a non-key variable of R and ending in x_1 witnessing the attack (see Definition 4.10). In particular is $v_k = x_1$. Since x_1 and x_2 belong to the same atom S in q and $x_2 \notin (\mathsf{var}(\vec{y}))^+_{\Sigma(q\setminus R)}$, we can extend the sequence by setting $v_{k+1} = x_2$ thus obtaining a witness for the attack from R to x_2, which results in a contradiction since x_2 is not attacked by our assumption. Hence, $x_2 \in (\mathsf{var}(\vec{y}))^+_{\Sigma(q\setminus R)}$ and thus $\alpha(x_2) = \beta(x_2) = \delta(x_2)$. Also, $\delta(x_1) = \alpha(x_1)$ by construction of δ. Since, x_1 and x_2 were arbitrary, δ maps $S(\vec{y'};\vec{z'})$ to B, which is in $\mathfrak{R}$.

We have shown that δ maps $S(\vec{y'};\vec{z'})$ to either B or B', and hence to a fact in $\mathfrak{R}$. Therefore, δ maps every atom in q to a fact in $\mathfrak{R}$ and thus $\mathfrak{R}, \delta \models_{\mathbb{K}} \hat{q}$, which then finishes the proof of Claim 2. □

Since we assumed that $\mathfrak{R}$ is a repair of $\mathfrak{D}$ such that $\rho(\mathfrak{R})$ is subset-minimal, Claim 1 now follows by a contradiction given by Claim 2, which concludes the proof of the lemma. □

The proof of the following lemma can be found in Appendix B.3.

Lemma 4.15. *Let q be a* sjfCQ *and Σ a set of key constraints, one per relation, with an acyclic attack graph. Let $\alpha : \mathsf{Var} \longrightarrow A$ be an assignment and x be an unattacked variable in q. Then, for every $\mathbb{K}$-database $\mathfrak{D}$,*

$$\mathrm{mCA}_{\mathbb{K}}(q, \mathfrak{D}, \alpha) = \sum_{c \in D} \mathrm{mCA}_{\mathbb{K}}(q[x], \mathfrak{D}, \alpha(c/x)).$$

We are now ready to give the proof of our main result.

Theorem 4.9. *Let $\mathbb{K}$ be a naturally ordered positive semiring, q be a self-join free conjunctive query, and Σ a set of key constraints, one for each relation in q. The attack graph of q is acyclic if and only if $\mathrm{mCA}_{\mathbb{K}}(q, \Sigma)$ is $\mathcal{L}_{\mathbb{K}}$-rewritable.*

Proof. The right-to-left direction follows from Proposition 4.12. We prove the left-to-right direction by induction on $|q|$. We will simultaneously define the rewriting and prove its correctness. We first prove the correctness of the rewriting for $\mathbb{K}$-databases $\mathfrak{D}$ and assignments α such that $\mathfrak{D}', \alpha \models_{\mathbb{K}} q$, for every repair $\mathfrak{D}'$ of $\mathfrak{D}$, then at the end of the proof argue that the rewriting is correct also for the remaining case. Let q be a self-join-free conjunctive query whose attack graph is acyclic. For the base case, assume $|q| = 1$, that is, $q = \exists\vec{x}R(\vec{y};\vec{z})$ where $\vec{x} \subseteq \vec{y} \cup \vec{z}$. Since q is acyclic, there exists an unattacked atom in q, which in this case is $R(\vec{y};\vec{z})$. Set $\vec{y}_{\vec{x}}$ (resp. $\vec{z}_{\vec{x}}$) to be the list of variables that are in $\vec{y}$ (resp. in $\vec{z}$) and $\vec{x}$. By Lemma 4.15, we get

$$\mathrm{mCA}_{\mathbb{K}}(q,\mathfrak{D},\alpha) = \sum_{\vec{a}\in D^{|\vec{y}_{\vec{x}}|}} \mathrm{mCA}_{\mathbb{K}}(q[\vec{y}_{\vec{x}}],\mathfrak{D},\alpha'),\ \text{where } \alpha' \coloneqq \alpha(\vec{a}/\vec{y}_{\vec{x}})\ .$$

By definition of mCA and since there is exactly one key-equal fact to $R(\alpha'(\vec{y});\vec{z})$ in every repair $\mathfrak{D}'$ of $\mathfrak{D}$,

$$\mathrm{mCA}_{\mathbb{K}}(q[\vec{y}_{\vec{x}}],\mathfrak{D},\alpha') = \min_{\mathfrak{D}'\in\mathrm{Rep}(\mathfrak{D},\Sigma)} q[\vec{y}_{\vec{x}}](\mathfrak{D}',\alpha') = \min_{\vec{b}\in D^{|\vec{z}_{\vec{x}}|}:R^{\mathfrak{D}}(\beta(\vec{y});\beta(\vec{z}))\neq 0} R^{\mathfrak{D}}(\beta(\vec{y});\beta(\vec{z})), \tag{7}$$

where $\beta \coloneqq \alpha'(\vec{b}/\vec{z}_{\vec{x}})$. This yields the following $\mathcal{L}_{\mathbb{K}}$-rewriting φ_q for the base case:

$$\varphi_q \coloneqq \exists \vec{y}_x \nabla_{R(\vec{y};\vec{z})} \vec{z}_{\vec{x}}.R(\vec{y};\vec{z}).$$

If there is a repair $\mathfrak{D}'$ of $\mathfrak{D}$ for which $\mathfrak{D}',\alpha \not\models_{\mathbb{K}} q$, since the semiring is positive, the last minimization in (7) is over the empty set for every $\vec{a}\in D^{|\vec{y}_{\vec{x}}|}$, thus $\varphi_q(\mathfrak{D},\alpha) = 0 = \mathrm{mCA}_k(q,\mathfrak{D},\alpha)$. For the inductive step, suppose that $|q| = n > 1$, and that the claim holds for every acyclic sjfCQ of size at most $n-1$. Let $R(\vec{y};\vec{z})$ be an unattacked atom in q, which exists since q is acyclic. As done in the base case, we define $\vec{y}_{\vec{x}}$ and $\vec{z}_{\vec{x}}$, we use Lemma 4.15, and focus on rewriting the expression $\mathrm{mCA}_{\mathbb{K}}(q[\vec{y}_{\vec{x}}],\mathfrak{D},\alpha')$, where $\alpha' \coloneqq \alpha(\vec{a}/\vec{y}_{\vec{x}})$. We want to show that:

$$\begin{aligned}\min_{\mathfrak{D}'\in\mathrm{Rep}(\mathfrak{D},\Sigma)} q[\vec{y}_{\vec{x}}](\mathfrak{D}',\alpha') &= \min_{\vec{b}\in D^{|\vec{z}_{\vec{x}}|}:R^{\mathfrak{D}}(\beta(\vec{y});\beta(\vec{z}))\neq 0} \big(R^{\mathfrak{D}}(\beta(\vec{y});\beta(\vec{z})) \times \min_{\mathfrak{D}'\in\mathrm{Rep}(\mathfrak{D},\Sigma)} \widetilde{q}(\mathfrak{D}',\beta)\big)\\ &= \min_{\vec{b}\in D^{|\vec{z}_{\vec{x}}|}:R^{\mathfrak{D}}(\beta(\vec{y});\beta(\vec{z}))\neq 0} \big(R^{\mathfrak{D}}(\beta(\vec{y});\beta(\vec{z})) \times \mathrm{mCA}_{\mathbb{K}}(\widetilde{q},\mathfrak{D},\beta)\big)\end{aligned} \tag{8}$$

where $\widetilde{q} = q[\vec{y}_{\vec{x}}][\vec{z}_{\vec{x}}] \setminus R(\vec{y};\vec{z})$, and $\beta \coloneqq \alpha'(\vec{b}/\vec{z}_{\vec{x}})$. It is easy to see that:

$$\min_{\mathfrak{D}'\in\mathrm{Rep}(\mathfrak{D},\Sigma)} q[\vec{y}_{\vec{x}}](\mathfrak{D}',\alpha') \geq_{\mathbb{K}} \min_{\vec{b}\in D^{|\vec{z}_{\vec{x}}|}:R^{\mathfrak{D}}(\beta(\vec{y});\beta(\vec{z}))\neq 0} \big(R^{\mathfrak{D}}(\beta(\vec{y});\beta(\vec{z})) \times \min_{\mathfrak{D}'\in\mathrm{Rep}(\mathfrak{D},\Sigma)} \widetilde{q}(\mathfrak{D}',\beta)\big).$$

To show the other direction, assume that $\mathrm{mCA}_{\mathbb{K}}(q[\vec{y}_{\vec{x}}],\mathfrak{D},\alpha') > 0$, and let $\mathfrak{D}_{\min}$ be a repair such that $q[\vec{y}_{\vec{x}}](\mathfrak{D}_{\min},\alpha')$ is minimum over all repairs of $\mathfrak{D}$. We will prove that, for every repair $\mathfrak{D}^*$ of $\mathfrak{D}$ and for every $\vec{b}\in D^{|\vec{z}_{\vec{x}}|}$ such that $R^{\mathfrak{D}}(\beta(\vec{y});\beta(\vec{z}))\neq 0$:

$$q[\vec{y}_{\vec{x}}](\mathfrak{D}_{\min},\alpha') \leq_{\mathbb{K}} R^{\mathfrak{D}}(\beta(\vec{y});\beta(\vec{z})) \times \widetilde{q}(\mathfrak{D}^*,\beta). \tag{9}$$

It follows, as in the proof of Lemma 4.15, that $q[\vec{y}_{\vec{x}}](\mathfrak{D}_{\min},\alpha') \leq_{\mathbb{K}} q[\vec{y}_{\vec{x}}](\mathfrak{D}^*,\alpha')$. Since there is exactly one key-equal fact (with a non-zero annotation) with relation name R per repair, in particular in $\mathfrak{D}^*$, the latter can be rewritten as $R^{\mathfrak{D}}(\beta(\vec{y});\beta(\vec{z})) \times \widetilde{q}(\mathfrak{D}^*,\beta)$. This proves (9), which then gives us the desired expression for $\mathrm{mCA}_{\mathbb{K}}(q[\vec{y}_{\vec{x}}],\mathfrak{D},\alpha')$.

On the other hand, since $|\widetilde{q}| < n$, it follows from the induction hypothesis that $\mathrm{mCA}_{\mathbb{K}}(\widetilde{q},\Sigma)$ admits an $\mathcal{L}_{\mathbb{K}}$-rewriting $\varphi_{\widetilde{q}}$, hence we obtain the following $\mathcal{L}_{\mathbb{K}}$-rewriting for $\mathrm{mCA}_{\mathbb{K}}(q,\Sigma)$:

$$\varphi_q \coloneqq \exists \vec{y}_{\vec{x}} \nabla_{R(\vec{y};\vec{z})} \vec{z}_{\vec{x}}.\varphi_{\widetilde{q}}.$$

If there exists a repair $\mathfrak{D}'$ such that $\mathfrak{D}',\alpha \not\models_{\mathbb{K}} q$, then $\mathrm{mCA}(q,\mathfrak{D},\alpha) = 0$. From this and the hypothesis that the semiring $\mathbb{K}$ is positive, it follows that when we inspect (8) for every $\vec{a}\in D^{|\vec{y}_{\vec{x}}|}$, either the set $\{\vec{b}\in D^{|\vec{z}_{\vec{x}}|} : R^{\mathfrak{D}}(\beta(\vec{z}),\beta(\vec{z}))\neq 0\}$ is empty and thus the minimization is done over an empty set, or the minimization of $\widetilde{q}(\mathfrak{R},\alpha)$ over all repairs $\mathfrak{R}$ is 0, hence $\varphi_q(\mathfrak{D},\alpha) = 0$. This completes the proof of the theorem. □

The preceding theorem generalizes the Koutris-Wijsen rewritability result in [26]. Furthermore, it gives new rewritability results for the bag, the tropical, the Viterbi, and the fuzzy semiring.

5 From rewritability to non-approximability

Since the early days of computational complexity, there has been an extensive study of the *approximation* properties of optimization problems, i.e., whether or not there are "good" algorithms that approximate the optimum (maximum or minimum) value of some objective function defined on a space of feasible solutions. Most of the work on the approximation properties of optimization problems has focused on optimization problems whose underlying decision problem is NP-complete. This study has shown that such optimization problems may have very different approximation properties, ranging from polynomial-time approximation schemes (e.g., Knapsack) to polynomial-time constant-approximation algorithms (e.g., Min Vertex Cover) to polynomial-time logarithmic-approximation algorithms (e.g., Min Set Cover), or even worse approximation properties (e.g., Max Clique) - see [16, 32]. There has also been work on the approximation properties of optimization problems whose underlying decision problem is in some lower complexity class, such as L and NL, where L and NL denote, respectively, deterministic log-space and non-deterministic log-space [33].

Let $\mathbb{K} = (K, +, \times, 0, 1)$ be a naturally ordered positive semiring. Every closed query q and every set Σ of constraints give rise to the optimization problem $\mathrm{mCA}_{\mathbb{K}}(q, \Sigma)$: given a $\mathbb{K}$-database $\mathfrak{D}$, compute $\mathrm{mCA}_{\mathbb{K}}(q, \Sigma, \mathfrak{D})$. The decision problem underlying $\mathrm{mCA}_{\mathbb{K}}(q, \Sigma)$ asks: given a $\mathbb{K}$-database $\mathfrak{D}$ and an element $k \in \mathbb{K}$, is $\mathrm{mCA}_{\mathbb{K}}(q, \Sigma, \mathfrak{D}) \leq_{\mathbb{K}} k$? In the case of the Boolean semiring $\mathbb{B}$, both the optimization problem $\mathrm{mCA}_{\mathbb{B}}(q, \Sigma)$ and its underlying decision problem coincide with the decision problem $\textsc{Cons}(q, \Sigma)$. Furthermore, as mentioned earlier, it has been shown in [26, 29] that if q is a self-join free conjunctive query with one key constraint per relation, then $\textsc{Cons}(q, \Sigma)$ exhibits a trichotomy, which we now spell out in detail: (i) if the attack graph of q is acyclic, then $\textsc{Cons}(q, \Sigma)$ is FO-rewritable; (ii) if the attack graph of q contains a *weak* cycle but no *strong* cycle, then $\textsc{Cons}(q, \Sigma)$ is L-complete, hence it is in P but it is not FO-rewritable; (iii) if the attack graph of q contains a strong cycle, then $\textsc{Cons}(q, \Sigma)$ is coNP-complete (the precise definitions of a weak cycle and of a strong cycle in the attack graph can be found in [26]).

Let q be a closed self-join free conjunctive query and let Σ be a set of key constraints, one for each relation of q. In what follows, we will leverage the above trichotomy result for $\textsc{Cons}(q, \Sigma)$ to study the approximation properties of computing $\mathrm{mCA}_{\mathbb{N}}(q, \Sigma)$, where $\mathbb{N} = (N, +, \times, 0, 1)$ is the bag semiring. Let $\varepsilon \geq 1$ be a fixed constant and consider a minimization problem Q in which the objective function takes positive integers as values. An ε-approximation algorithm for Q is an algorithm that, given an input to Q, returns the value A of the objective function on some feasible solution so that $A/\mathrm{opt} \leq \varepsilon$, where opt is the value of the objective function on the given input (note that $A/\mathrm{opt} \geq 1$ because Q is a minimization problem). Since $\mathrm{mCA}_{\mathbb{N}}(q, \Sigma)$ may take the value 0 on an input $\mathbb{N}$-database $\mathfrak{D}$, we will consider the minimization problem: given an $\mathbb{N}$-database $\mathfrak{D}$, compute $\mathrm{mCA}_{\mathbb{N}}(q, \Sigma, \mathfrak{D}) + 1$.

Assume that q is a closed self-join free conjunctive query and Σ is a set of key constraints, one for each relation of q. For every $\varepsilon \geq 1$, let $\mathsf{APPROX}(q, \Sigma, \varepsilon)$ be the following function problem:

PROBLEM $\mathsf{APPROX}(q, \Sigma, \varepsilon)$
INPUT: An arbitrary $\mathbb{N}$-database $\mathfrak{D}$.
OUTPUT: The value $q(\mathfrak{D}')$ of q on some repair $\mathfrak{D}'$ of $\mathfrak{D}$ such that $q(\mathfrak{D}')/\mathrm{opt} \leq \varepsilon$, where $\mathrm{opt} = \mathrm{mCA}_{\mathbb{N}}(q, \Sigma, \mathfrak{D}) + 1$.

Clearly, the inequality $q(\mathfrak{D}')/\mathrm{opt} \leq \varepsilon$ encapsulates a relative approximation guarantee. For example, if we take $\varepsilon = 1.5$, then $\mathsf{APPROX}(q, \Sigma, 1.5)$ asks for the value $q(\mathfrak{D}')$ of q on some repair $\mathfrak{D}'$ of $\mathfrak{D}$ such that $q(\mathfrak{D}') \leq 1.5 \cdot (\mathrm{mCA}_{\mathbb{N}}(q, \Sigma, \mathfrak{D}) + 1)$.

Proposition 5.1. *Let q be a closed self-join-free conjunctive query and let Σ be a set of key constraints, one for each relation of q. For every $\varepsilon \geq 1$, there is a first-order reduction from the complement of* $\textsc{Cons}(q, \Sigma)$ *to* $\mathsf{APPROX}(q, \Sigma, \varepsilon)$.

Proof. Fix q, Σ, and $\varepsilon \geq 1$, as in the hypothesis of this proposition. Let M be a fixed natural number such that $M > \varepsilon$ and pick a relation symbol R occurring in q.

Given a standard database $\mathfrak{D}_0$ (i.e., a $\mathbb{B}$-database) that is an input to $\textsc{Cons}(q, \Sigma)$, we construct an $\mathbb{N}$-database $\mathfrak{D}$ as follows:

- Annotate every R-fact of $\mathfrak{D}_0$ with M, that is, if $t \in \mathrm{Supp}(R^{\mathfrak{D}_0})$, then $R^{\mathfrak{D}}(t) = M$; otherwise $R^{\mathfrak{D}}(t) = 0$.
- Annotate all other facts in $\mathfrak{D}_0$ with 1, that is, for every relation symbol S in q that is different from R, if $s \in \mathrm{Supp}(S^{\mathfrak{D}_0})$, then $S^{\mathfrak{D}}(s) = 1$; otherwise $S^{\mathfrak{D}}(s) = 0$.

Since M is fixed, the $\mathbb{N}$-database $\mathfrak{D}$ can be constructed from $\mathfrak{D}_0$ in first-order logic. Furthermore, it is clear that there is a one-to-one correspondence between the repairs of $\mathfrak{D}_0$ and the repairs of $\mathfrak{D}$.

We now claim that $\textsc{Cons}(q, \Sigma, \mathfrak{D}_0)$ is false if and only if $\mathsf{APPROX}(q, \Sigma, \varepsilon)$ returns $0 = q(\mathfrak{D}')$ for some repair $\mathfrak{D}'$ of $\mathfrak{D}$, where, of course, the evaluation of $q(\mathfrak{D}')$ is over the bag semiring $\mathbb{N}$.

Assume first that $\textsc{Cons}(q, \Sigma, \mathfrak{D}_0)$ is false, which means that there is a repair $\mathfrak{D}_0^*$ of $\mathfrak{D}_0$ that falsifies q. It is easily verified that there is a (unique) repair $\mathfrak{D}^*$ of $\mathfrak{D}$ such that $\mathrm{Supp}(\mathfrak{D}^*) = \mathfrak{D}_0^*$. Furthermore, $q(\mathfrak{D}^*) = 0$, hence $\mathrm{mCA}_{\mathbb{N}}(q, \Sigma, \mathfrak{D}) = 0$ and opt $= 1$. Now take any repair $\mathfrak{D}'$ of $\mathfrak{D}$ such that $q(\mathfrak{D}') \neq 0$. By our construction, $q(\mathfrak{D}') \geq M > \varepsilon$, hence $q(\mathfrak{D}')/\mathrm{opt} \geq M > \varepsilon$, which implies that $q(\mathfrak{D}') \neq 0$ is not a valid output for $\mathsf{APPROX}(q, \Sigma, \varepsilon)$. It follows that $\mathsf{APPROX}(q, \Sigma, \varepsilon)$ must return $0 = q(\mathfrak{D}')$ for some repair $\mathfrak{D}'$ of $\mathfrak{D}$.

Conversely, assume that $\mathsf{APPROX}(q, \Sigma, \varepsilon)$ returns $0 = q(\mathfrak{D}')$ for some repair $\mathfrak{D}'$ of $\mathfrak{D}$. By our construction, it must be the case that $\mathrm{Supp}(\mathfrak{D}')$ is a repair of $\mathfrak{D}_0$ that falsifies q, hence $\textsc{Cons}(q, \Sigma, \mathfrak{D}_0)$ is false. This concludes the proof of Proposition 5.1. □

Corollary 5.2. *Let q be a closed self-join-free conjunctive query and let Σ be a set of key constraints, one for each relation of q. Then the following statements are true.*

- *If the attack graph of q contains a (weak or strong) cycle, then for every $\varepsilon \geq 1$, the problem* $\mathsf{APPROX}(q, \Sigma, \varepsilon)$ *is* L*-hard under first-order reductions; and*
- *if the attack graph of q contains a strong cycle, then for every $\varepsilon \geq 1$, the problem* $\mathsf{APPROX}(q, \Sigma, \varepsilon)$ *is* NP*-hard under first-order reductions.*

Proof. As discussed earlier, it is shown in [26] that the complement of $\textsc{Cons}(q, \Sigma)$ is L-hard if the attack graph of q contains a cycle, and NP-hard if the attack graph of q contains a strong cycle. The desired conclusions then follow from Proposition 5.1. □

We illustrate Corollary 5.2 with two examples.

- Let $\mathrm{q}_{\mathrm{cycle}}$ be the query $\exists x \exists y (R(x; y) \wedge S(y; x))$. It is known that the attack graph of $\mathrm{q}_{\mathrm{cycle}}$ contains a weak cycle but not a strong cycle. Thus, for every $\varepsilon \geq 1$, we have that $\mathsf{APPROX}(\mathrm{q}_{\mathrm{cycle}}, \Sigma, \varepsilon)$ is L-hard under first-order reductions.
- Let $\mathrm{q}_{\mathrm{sink}}$ be the query $\exists x \exists y \exists z (R(x; z) \wedge S(y; z))$. It is known that the attack graph of $\mathrm{q}_{\mathrm{sink}}$ contains a strong cycle. Thus, for every $\rho \geq 1$ and every $\alpha \geq 0$, we have that $\mathsf{APPROX}(\mathrm{q}_{\mathrm{sink}}, \Sigma, \varepsilon)$ is NP-hard under first-order reductions.

6 Directions for Future Research

In this paper, we initiated a study of consistent query answering for databases over a naturally ordered positive semiring. The results obtained suggest several research directions in this area.

First, an interesting open question is to extend our complexity study of $\mathrm{mCA}_{\mathbb{K}}(q, \Sigma)$ from rewritability in $\mathcal{L}_{\mathbb{K}}$ to computability in polynomial time. Consider a self-join free conjunctive query whose attack graph contains a weak cycle, but not a *strong* cycle. In the case of standard databases, it was shown in [26] that the consistent answers of such queries are polynomial-time computable, but not FO-rewritable. Does this result extend and how does it extend to the consistent answers of such queries for $\mathbb{K}$-databases, where $\mathbb{K}$ is a naturally ordered positive semiring? Here, among others, we have the problem of how to formulate the "right" notion of polynomial-time computability over semirings. For standard databases, two polynomial-time approaches are known for computing consistent answers to self-join-free conjunctive queries with a cyclic attack graph that has no strong cycles: through a rewriting in a variant of Datalog [29], or via a more general algorithm developed by Figueira et al. [14], which can be formulated in some fixpoint logic. It is an open question whether these approaches can be adapted beyond the Boolean semiring to arbitrary naturally ordered positive semirings. In this respect, it is also significant that the algorithm of Figueira et al. is not restricted to the self-join-free case but can also handle some, though not all, conjunctive queries with self-joins. Their algorithm could provide a pathway for extending the results in the current paper to conjunctive queries with self-joins. Nevertheless, one should be aware that consistent query answering to conjunctive queries with self-joins is a notorious open problem. For example, for conjunctive queries q with self-joins and primary keys Σ, the complexity of $\mathrm{mCA}_{\mathbb{B}}(q, \Sigma)$ is understood for queries with at most two atoms [31], but is largely open for queries with three or more atoms.

Second, we initiated an investigation into approximating consistent query answers when the computation of exact results is intractable. In particular, we showed that if q is a self-join free conjunctive query whose attack graph contains a strong cycle, then the consistent answers on bag databases (i.e., $\mathrm{mCA}_{\mathbb{N}}(q, \Sigma)$) are not approximable in polynomial time, unless $\mathsf{P} = \mathsf{NP}$. How does this result extend to naturally ordered positive semirings other than the bag semiring?

Third, in a recent paper [2], consistent query answering is studied for primary keys and numerical queries that return a single number obtained by aggregating (e.g., by means of SUM or AVG) the results returned by a self-join-free conjunctive query $q(r)$, where the free variable r is numerical and ranges over $\mathbb{N}$. The range semantics established in [4] requires computing the greatest lower bound (glb) and the least upper bound (lub) on the answers to the numerical query over all repairs. The problem of finding the glb for SUM queries can be restated as a special case of $\mathrm{mCA}_{\mathbb{N}}(q, \Sigma)$. The authors of [2] study rewritings in aggregate logic, which is different from the logic $\mathcal{L}_{\mathbb{N}}$ in the current paper, and obtain a dichotomy similar to our Theorem 4.9. While their rewriting adresses a special case of $\mathrm{mCA}_{\mathbb{N}}(q, \Sigma)$, the two approaches differ significantly in their formalism and underlying syntax. A deeper theoretical exploration is required to precisely pinpoint the differences and commonalities between the two approaches.

Finally, it is also natural to introduce least-upper-bound semantics in the context of semirings. Specifically, the *possible answers* $\mathrm{MCA}_{\mathbb{K}}(\varphi, \Sigma, \mathfrak{D}, \alpha)$ *of* φ *on* $\mathfrak{D}, \alpha$ *with respect to* Σ is defined as $\max_{\mathfrak{D}' \in \mathrm{Rep}(\mathfrak{D},\Sigma)} \varphi(\mathfrak{D}', \alpha)$. Thus, the possible answers provide the tightest upper bound on the values $\varphi(\mathfrak{D}', \alpha)$ as $\mathfrak{D}'$ ranges over all repairs of $\mathfrak{D}$.

Acknowledgments

We would like to thank the anonymous reviewers for their constructive feedback. The second and the third author were partially supported by the DFG grant VI 1045-1/1.

References

[1] Serge Abiteboul, Richard Hull, and Victor Vianu. 1995. *Foundations of Databases.* Addison-Wesley.

[2] Aziz Amezian El Khalfioui and Jef Wijsen. 2024. Computing Range Consistent Answers to Aggregation Queries via Rewriting. *Proc. ACM Manag. Data* 2, 5, Article 218 (Nov. 2024), 19 pages. https://doi.org/10.1145/3695836

[3] Marcelo Arenas, Leopoldo E. Bertossi, and Jan Chomicki. 1999. Consistent Query Answers in Inconsistent Databases. In *Proceedings of the Eighteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, May 31 - June 2, 1999, Philadelphia, Pennsylvania, USA*, Victor Vianu and Christos H. Papadimitriou (Eds.). ACM Press, 68–79. https://doi.org/10.1145/303976.303983

[4] Marcelo Arenas, Leopoldo E. Bertossi, Jan Chomicki, Xin He, Vijay Raghavan, and Jeremy P. Spinrad. 2003. Scalar aggregation in inconsistent databases. *Theor. Comput. Sci.* 296, 3 (2003), 405–434. https://doi.org/10.1016/S0304-3975(02)00737-5

[5] William W. Armstrong. 1974. Dependency Structures of Data Base Relationships.. In *Proc. of IFIP World Computer Congress*. 580–583.

[6] Albert Atserias and Phokion G. Kolaitis. 2024. Consistency of Relations over Monoids. *Proc. ACM Manag. Data* 2, 2 (2024), 107. https://doi.org/10.1145/3651608

[7] Timon Barlag, Miika Hannula, Juha Kontinen, Nina Pardal, and Jonni Virtema. 2023. Unified Foundations of Team Semantics via Semirings. In *KR*. 75–85.

[8] Timon Barlag and Heribert Vollmer. 2021. A Logical Characterization of Constant-Depth Circuits over the Reals. In *Logic, Language, Information, and Computation - 27th International Workshop, WoLLIC 2021, Virtual Event, October 5-8, 2021, Proceedings (Lecture Notes in Computer Science, Vol. 13038)*, Alexandra Silva, Renata Wassermann, and Ruy J. G. B. de Queiroz (Eds.). Springer, 16–30. https://doi.org/10.1007/978-3-030-88853-4_2

[9] David A. Mix Barrington, Neil Immerman, and Howard Straubing. 1990. On Uniformity within NC^1. *J. Comput. Syst. Sci.* 41, 3 (1990), 274–306. https://doi.org/10.1016/0022-0000(90)90022-D

[10] Lenore Blum, Felipe Cucker, Michael Shub, and Steve Smale. 1997. *Complexity and Real Computation*. Springer-Verlag, Berlin, Heidelberg.

[11] Katrin M. Dannert, Erich Grädel, Matthias Naaf, and Val Tannen. 2021. Semiring Provenance for Fixed-Point Logic. In *29th EACSL Annual Conference on Computer Science Logic, CSL 2021, January 25-28, 2021, Ljubljana, Slovenia (Virtual Conference) (LIPIcs, Vol. 183)*, Christel Baier and Jean Goubault-Larrecq (Eds.). Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 17:1–17:22. https://doi.org/10.4230/LIPIcs.CSL.2021.17

[12] Daniel Deutch, Tova Milo, Sudeepa Roy, and Val Tannen. 2014. Circuits for Datalog Provenance. In *Proc. 17th International Conference on Database Theory (ICDT), Athens, Greece, March 24-28, 2014*, Nicole Schweikardt, Vassilis Christophides, and Vincent Leroy (Eds.). OpenProceedings.org, 201–212. https://doi.org/10.5441/002/ICDT.2014.22

[13] Heinz-Dieter Ebbinghaus, Jörg Flum, and Wolfgang Thomas. 1984. *Mathematical logic*. Springer.

[14] Diego Figueira, Anantha Padmanabha, Luc Segoufin, and Cristina Sirangelo. 2023. A Simple Algorithm for Consistent Query Answering Under Primary Keys. In *26th International Conference on Database Theory, ICDT 2023, March 28-31, 2023, Ioannina, Greece (LIPIcs, Vol. 255)*, Floris Geerts and Brecht Vandevoort (Eds.). Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 24:1–24:18. https://doi.org/10.4230/LIPICS.ICDT.2023.24

[15] Ariel Fuxman and Renée J. Miller. 2007. First-order query rewriting for inconsistent databases. *J. Comput. Syst. Sci.* 73, 4 (2007), 610–635. https://doi.org/10.1016/J.JCSS.2006.10.013

[16] M. R. Garey and David S. Johnson. 1979. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman.

[17] Erich Grädel and Val Tannen. 2017. Semiring Provenance for First-Order Model Checking. *CoRR* abs/1712.01980 (2017).

[18] Todd J. Green. 2011. Containment of Conjunctive Queries on Annotated Relations. *Theory Comput. Syst.* 49, 2 (2011), 429–459. https://doi.org/10.1007/S00224-011-9327-6

[19] Todd J. Green, Gregory Karvounarakis, and Val Tannen. 2007. Provenance semirings. In *Proceedings of the Twenty-Sixth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, June 11-13, 2007, Beijing, China*, Leonid Libkin (Ed.). ACM, 31–40. https://doi.org/10.1145/1265530.1265535

[20] Erich Grädel and Val Tannen. 2024. Provenance analysis and semiring semantics for first-order logic. (2024). To appear in a volume in honor of János Makowsky, published by Birkhäuser.

[21] S. Jukna. 2023. *Tropical Circuit Complexity: Limits of Pure Dynamic Programming*. Springer International Publishing. https://books.google.co.uk/books?id=sLUO0AEACAAJ

[22] Mahmoud Abo Khamis, Hung Q. Ngo, Reinhard Pichler, Dan Suciu, and Yisu Remy Wang. 2022. Convergence of Datalog over (Pre-) Semirings. In *PODS '22: International Conference on Management of Data, Philadelphia, PA, USA, June 12 - 17, 2022*, Leonid Libkin and Pablo Barceló (Eds.). ACM, 105–117. https://doi.org/10.1145/3517804.3524140

[23] Egor V. Kostylev, Juan L. Reutter, and András Z. Salamon. 2012. Classification of annotation semirings over query containment. In *Proceedings of the 31st ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS 2012, Scottsdale, AZ, USA, May 20-24, 2012*, Michael Benedikt, Markus Krötzsch, and Maurizio Lenzerini (Eds.). ACM, 237–248. https://doi.org/10.1145/2213556.2213590

[24] Paraschos Koutris, Xiating Ouyang, and Jef Wijsen. 2021. Consistent Query Answering for Primary Keys on Path Queries. In *PODS'21: Proceedings of the 40th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, Virtual Event, China, June 20-25, 2021*, Leonid Libkin, Reinhard Pichler, and Paolo Guagliardo (Eds.). ACM, 215–232. https://doi.org/10.1145/3452021.3458334

[25] Paraschos Koutris, Xiating Ouyang, and Jef Wijsen. 2024. Consistent Query Answering for Primary Keys on Rooted Tree Queries. *Proc. ACM Manag. Data* 2, 2 (2024), 76. https://doi.org/10.1145/3651139

[26] Paraschos Koutris and Jef Wijsen. 2017. Consistent Query Answering for Self-Join-Free Conjunctive Queries Under Primary Key Constraints. *ACM Trans. Database Syst.* 42, 2 (2017), 9:1–9:45. https://doi.org/10.1145/3068334

[27] Paraschos Koutris and Jef Wijsen. 2018. Consistent Query Answering for Primary Keys and Conjunctive Queries with Negated Atoms. In *Proceedings of the 37th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, Houston, TX, USA, June 10-15, 2018*, Jan Van den Bussche and Marcelo Arenas (Eds.). ACM, 209–224. https://doi.org/10.1145/3196959.3196982

[28] Paraschos Koutris and Jef Wijsen. 2020. First-Order Rewritability in Consistent Query Answering with Respect to Multiple Keys. In *Proceedings of the 39th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS 2020, Portland, OR, USA, June 14-19, 2020*, Dan Suciu, Yufei Tao, and Zhewei Wei (Eds.). ACM, 113–129. https://doi.org/10.1145/3375395.3387654

[29] Paraschos Koutris and Jef Wijsen. 2021. Consistent Query Answering for Primary Keys in Datalog. *Theory Comput. Syst.* 65, 1 (2021), 122–178. https://doi.org/10.1007/S00224-020-09985-6

[30] Leonid Libkin. 2004. *Elements of Finite Model Theory.* Springer. https://doi.org/10.1007/978-3-662-07003-1

[31] Anantha Padmanabha, Luc Segoufin, and Cristina Sirangelo. 2024. A Dichotomy in the Complexity of Consistent Query Answering for Two Atom Queries With Self-Join. *Proc. ACM Manag. Data* 2, 2 (2024), 74. https://doi.org/10.1145/3651137

[32] Christos H. Papadimitriou. 1994. *Computational complexity.* Addison-Wesley.

[33] Till Tantau. 2007. Logspace Optimization Problems and Their Approximability Properties. *Theory Comput. Syst.* 41, 2 (2007), 327–350. https://doi.org/10.1007/S00224-007-2011-1

[34] Moshe Y. Vardi. 1982. The complexity of relational query languages (Extended Abstract). In *Proc. ACM Symposium on Theory of Computing (STOC 82.* 137–146.

[35] Heribert Vollmer. 1999. *Introduction to Circuit Complexity - A Uniform Approach.* Springer. https://doi.org/10.1007/978-3-662-03927-4

[36] Jef Wijsen. 2010. On the first-order expressibility of computing certain answers to conjunctive queries over uncertain databases. In *Proceedings of the Twenty-Ninth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS 2010, June 6-11, 2010, Indianapolis, Indiana, USA*, Jan Paredaens and Dirk Van Gucht (Eds.). ACM, 179–190. https://doi.org/10.1145/1807085.1807111

[37] Jef Wijsen. 2012. Certain conjunctive query answering in first-order logic. *ACM Trans. Database Syst.* 37, 2 (2012), 9:1–9:35. https://doi.org/10.1145/2188349.2188351

[38] Hangdong Zhao, Shaleen Deep, Paraschos Koutris, Sudeepa Roy, and Val Tannen. 2024. Evaluating Datalog over Semirings: A Grounding-based Approach. *Proc. ACM Manag. Data* 2, 2 (2024), 90. https://doi.org/10.1145/3651591

A Semiring semantics via interpretations

In this appendix, we give some background on semiring semantics via interpretations and provide further justification for the definition of repairs given in our paper, i.e, why it is reasonable to define repairs using flattening, as done in Section 3. The main references are two papers by Grädel and Tannen [17] and [20] in which semiring semantics to first-order logic FO is given using the notion of an *interpretation*. We first recall the basic definitions from these two papers.

In what follows, we assume that $\mathbb{K} = (K, +, \times, 0, 1)$ is a positive semiring and $\tau = (R_1, \dots, R_m)$ is a relational schema.

Let D be a finite set and let Lit_D be the set of all atomic and negated atomic facts involving elements of D, i.e., all expressions of the form $R_i(\mathbf{a})$ and $\neg R_i(\mathbf{a})$, where $\mathbf{a}$ is a tuple of elements from D. We will refer to such expressions as *literals* from D.

- An *interpretation* on D is a function $\pi : \mathsf{Lit}_D \to K$.
- An interpretation π is *model defining* if for every atomic fact $R_i(\mathbf{a})$, exactly one of the values $\pi(R_i(\mathbf{a}))$ and $\pi(\neg R_i(\mathbf{a}))$ is different from 0.
- Every model-defining interpretation π on D determines a unique finite structure $\mathbf{D}_\pi = (D, R_1^{\mathbf{D}_\pi}, \dots, R_m^{\mathbf{D}_\pi})$ with universe D, where for every i and for every tuple $\mathbf{a}$ from D, we have that $\mathbf{a} \in R_i^{\mathbf{D}_\pi}$ if and only if $\pi(R_i(\mathbf{a})) \neq 0$.

Let π be an interpretation on D. To every FO-formula $\varphi(x_1, \dots, x_n)$ in negation normal form (NNF) and every tuple $(a_1, \dots, a_n)$ from D, Grädel and Tannen [17, 20] assign a value $\pi(\varphi(x_1/a_1, \dots, x_n/a_n))$ in K. The definition extends the values of the interpretation π by induction on the construction of FO-formulas; in this definition, the addition + operation of $\mathbb{K}$ is used to define the semantics of $\vee$ and $\exists$, while the multiplication × operation of $\mathbb{K}$ is used to define the semantics of $\wedge$ and $\forall$. In particular, for every FO-sentence ψ in NNF, the interpretation assigns a semiring value $\pi(\psi)$ to ψ. The following proposition (Proposition 5 in [20]) will be useful in the sequel.

Proposition A.1. *Let π be a model-defining interpretation on D. Then for every FO-sentence ψ in NNF, we have that*

$$\pi(\psi) \neq 0 \quad \textit{if and only if} \quad \mathbf{D}_\pi \models \psi.$$

Note that $\mathbf{D}_\pi$ is an ordinary finite structure. Grädel and Tannen [17, 20] do *not* give semantics of FO on $\mathbb{K}$ structures, i.e., on finite structures of the form $\mathbf{D} = (D, R_1^{\mathbf{A}}, \dots, R_m^{\mathbf{A}})$, where each $R_i^{\mathbf{D}}$ is a function from D^{r_i} to K and r_i is the arity of the relation symbol R_i. In particular, they never define what it means for a $\mathbb{K}$-structure $\mathbf{D}$ or for a $\mathbb{K}$-database $\mathfrak{D}$ to *satisfy* a FO-sentence ψ. Yet, we need such a definition in order to define the notion of a *repair* of a $\mathbb{K}$-database with respect to a set of integrity constraints.

Definition A.2. Let $\mathbf{D} = (D, R_1^{\mathbf{D}}, \dots, R_m^{\mathbf{D}})$ be a finite $\mathbb{K}$-structure.

- We say that an interpretation π on D is *compatible with* $\mathbf{D}$ if π is model-defining and for every i and every tuple $\mathbf{a}$ from D, we have that $\pi(R_i(\mathbf{a})) = R_i^{\mathbf{D}}(\mathbf{a})$.
- The *canonical compatible interpretation with* $\mathbf{D}$ is the interpretation $\pi_{\mathbf{D}}$ such that for every i and every tuple $\mathbf{D}$ from D, we have that $\pi(R_i(\mathbf{a})) = R_i^{\mathbf{D}}$ and

$$\pi(\neg R_i(\mathbf{a})) = \begin{cases} 0, & \text{if } R_i^{\mathbf{D}}(\mathbf{a}) \neq 0 \\ 1, & \text{if } R_i^{\mathbf{D}}(\mathbf{a}) = 0. \end{cases}$$

Note that if π is an interpretation that is compatible with a $\mathbb{K}$-structure $\mathbf{D}$, then, by definition, π has to agree with each relation $R_i^{\mathbf{D}}$ on the atomic facts $R_i(\mathbf{a})$. For the negated atomic facts $\neg R_i(\mathbf{a})$, we have that $\pi(\neg R_i(\mathbf{a})) = 0$ if $R_i^{\mathbf{D}}(\mathbf{a}) \neq 0$, because π is model defining. If, however, $R_i^{\mathbf{D}}(\mathbf{a}) = 0$, then

$\pi(\neg R_i(\mathbf{a}))$ can be any non-zero value from K; in the case of the canonical compatible interpretation $\pi_{\mathbf{D}}$, we have that this non-zero value is 1.

Let $\mathbf{D}$ be a $\mathbb{K}$-structure and consider the canonical compatible interpretation $\pi_{\mathbf{D}}$. As discussed above this gives rise to an ordinary structure $\mathbf{D}_{\pi_{\mathbf{D}}}$ with universe D. From the definitions, it follows that $\mathbf{D}_{\pi_{\mathbf{D}}} = \mathrm{Supp}(\mathbf{D})$, where $\mathrm{Supp}(\mathbf{D}) = (D, \mathrm{Supp}(R_1^{\mathbf{D}}), \ldots, \mathrm{Supp}(R_m^{\mathbf{D}}))$ is the ordinary structure with universe D and relations the supports of the relations of $\mathbb{D}$.

Proposition A.3. *Let* $\mathbf{D}$ *be a* $\mathbb{K}$*-structure and let* ψ *be a* FO*-sentence. Then the following hold:*

- *If* π_1 *and* π_2 *are two interpretations on* D *that are compatible with* $\mathbf{D}$*, then* $\pi_1(\psi) \neq 0$ *if and only if* $\pi_2(\psi) \neq 0$.
- *For every interpretation* π *on* D *that is compatible with* $\mathbf{D}$*, the following statements are equivalent:*

(1) $\pi(\psi) \neq 0$*;*
(2) $\pi_{\mathbf{D}}(\psi) \neq 0$*;*
(3) $\mathrm{Supp}(\mathbf{D}) \models \psi$.

Proof. The first part of the proposition is proved using a straightforward induction and the definition of compatibility. The second part of the proposition follows from the first part of the proposition together with Proposition A.1, and the earlier fact that $\mathbf{D}_{\pi_{\mathbf{D}}} = \mathrm{Supp}(\mathbf{D})$. □

The preceding proposition motivates the following definition of what it means for a $\mathbb{K}$-structure to "satisfy" a FO-sentence.

Definition A.4. We say that a $\mathbf{K}$-structure $\mathbf{D}$ *satisfies* a FO-sentence ψ in NNF, denoted $\mathbf{D} \models_{\mathbb{K}} \psi$, if for some (equivalently, for all) interpretation π that is compatible with $\mathbf{D}$, we have that $\pi(\psi) \neq 0$.

By Proposition A.3, we have that $\mathbf{D} \models_{\mathbb{K}} \psi$ if and only if $\mathrm{Supp}(\mathbf{A}) \models \psi$.

Let us now turn to $\mathbb{K}$-databases. Every $\mathbb{K}$-database $\mathfrak{D}$ with $\mathbb{K}$-relations $R_1^{\mathfrak{D}}, \ldots, R_m^{\mathfrak{D}}$ can be viewed as a finite $\mathbb{K}$-structure $\overline{\mathfrak{D}}$ with universe the active domain D of $\mathfrak{D}$ and with the same relations as those of $\mathfrak{D}$, i.e., $\overline{\mathfrak{D}} = (D, R_1^{\mathfrak{D}}, \ldots, R_m^{\mathfrak{D}})$. It is now clear that the semiring semantics of FO on a $\mathbb{K}$-database $\mathfrak{D}$ that we gave in Section 3 coincides with the definition of the semantics of FO derived by using the canonical compatible interpretation $\pi_{\overline{\mathfrak{D}}}$. In particular, Proposition 3.1 is a special case of the above Proposition A.3. Furthermore, in defining what it means for a $\mathbb{K}$-database $\mathfrak{D}$ to satisfy a FO-sentence ψ, we could have used any interpretation π on D that is compatible with $\mathfrak{D}$, instead of the canonical compatible one. And this amounts to $\mathrm{Supp}(\mathfrak{D})$ satisfying ψ in the standard sense.

In summary, the preceding considerations justify using "flattening" to define the notion of a repair of a $\mathbb{K}$-database with respect to a set of key constraints and, more broadly, with respect to a set of FO-sentences.

B Proofs omitted from the main body

B.1 Rewritability of the consistent answers of the path query

Recall that $\mathrm{q_{path}} = \exists x \exists y \exists z (R(x;y) \wedge S(y;z))$, for which $\mathrm{mCA}_{\mathbb{B}}(\mathrm{q_{path}})$ is in P [15], and has the following first-order rewriting: $\exists x \exists z' (R(x,z') \wedge \forall z (R(x,z) \rightarrow \exists y (S(z,y))))$. We would like to obtain a similar expression in our setting for $\mathrm{q_{path}}$, that is, an expression utilising semiring operations that provides the answer $\mathrm{mCA}_K(\mathrm{q_{path}}, \mathfrak{D})$ for $\mathbb{K}$-databases without having to evaluate $\mathrm{q_{path}}$ on every repair of $\mathfrak{D}$. The semiring semantics of FO on a $\mathbb{K}$-database $\mathfrak{D}$ we gave in Section 3 are precisely the semiring semantics of FO using the canonical interpretation We define the expression $e_{path}(\mathfrak{D})$ as follows:

$$e_{path}(\mathfrak{D}) := \sum_{a \in D} \min_{b \in D : R^{\mathfrak{D}}(a,b) \neq 0} (R^{\mathfrak{D}}(a,b) \times \min_{c \in D : S^{\mathfrak{D}}(b,c) \neq 0} S^{\mathfrak{D}}(b,c)),$$

where $\sum$, $\times$, and min refer to operations of a naturally ordered positive semiring $\mathbb{K}$. We claim that

$$\mathrm{mCA}_{\mathbb{K}}(\mathrm{q}_{\mathrm{path}}, \mathfrak{D}) = e_{path}(\mathfrak{D}) \tag{10}$$

for every $\mathbb{K}$-database $\mathfrak{D}$.

Proof. We show that the equality holds. Note that

$$\mathrm{mCA}_{\mathbb{K}}(\mathrm{q}_{\mathrm{path}}, \mathfrak{D}) = \min_{\mathfrak{D}' \in \mathrm{Rep}(\mathfrak{D})} \mathrm{q}_{\mathrm{path}}(\mathfrak{D}') = \min_{\mathfrak{D}' \in \mathrm{Rep}(\mathfrak{D})} \sum_{a,b,c \in D'} R^{\mathfrak{D}'}(a,b) \times S^{\mathfrak{D}'}(b,c),$$

where the former equality is the definition of $\mathrm{mCA}_{\mathbb{K}}$ and the latter follows from the semiring semantics for CQs. To show (10), it suffices to establish that the following two statements hold:

(i) For every repair $\mathfrak{D}'$ of $\mathfrak{D}$, we have that $e_{path}(\mathfrak{D}) \leq_{\mathbb{K}} \mathrm{q}_{\mathrm{path}}(\mathfrak{D}')$;
(ii) There is a repair $\mathfrak{D}^*$ of $\mathfrak{D}$ such that $e_{path}(\mathfrak{D}) = q(\mathfrak{D}^*)$.

For (i), let $\mathfrak{D}' \in \mathrm{Rep}(\mathfrak{D})$ and $a, b, c \in D'$ be arbitrary, and consider the expression $R^{\mathfrak{D}'}(a,b) \times S^{\mathfrak{D}'}(b,c)$. We distinguish the following three cases:
Case 1: a is not a key value for $R^{\mathfrak{D}}$.
Case 2: a is a key value for $R^{\mathfrak{D}}$, but b is not a key value for $S^{\mathfrak{D}}$.
Case 3. a is a key value for $R^{\mathfrak{D}}$ and b is a key value for $S^{\mathfrak{D}}$.

Note that, in the sum defining $e_{path}(\mathfrak{D})$, the element a contributes only one summand to that sum. In Case 1, in which a is not a key value for $R^{\mathfrak{D}}$, the summand to which a contributes has value 0, since we set $\min(\emptyset) = 0$. Similarly, in Case 2, in which a is a key value for $R^{\mathfrak{D}}$ but b is not a key value for $S^{\mathfrak{D}}$, the factor $\min_{z \in D: S^{\mathfrak{D}}(b,z) \neq 0} S^{\mathfrak{D}}(b,z)$ to which b contributes takes value 0, and hence the summand to which a contributes takes value 0. In Case 3, in which a is a key value for $R^{\mathfrak{D}}$ and b is a key value for $S^{\mathfrak{D}}$, by monotonicity of multiplication (if $j' \leq_{\mathbb{K}} j$ then $i \times j' \leq_{\mathbb{K}} i \times j$) we have

$$\min_{y \in D: R^{\mathfrak{D}}(a,y) \neq 0} (R^{\mathfrak{D}}(a,y) \times \min_{z \in D: S^{\mathfrak{D}}(y,z) \neq 0} S^{\mathfrak{D}}(y,z)) \leq_{\mathbb{K}} R^{\mathfrak{D}'}(a,b) \times S^{\mathfrak{D}'}(b,c).$$

Therefore, the summand of $e_{path}(\mathfrak{D})$ to which a contributes is dominated by the summand of $\mathrm{q}_{\mathrm{path}}(\mathfrak{D}')$ to which a contributes. Hence $e_{path}(\mathfrak{D}) \leq_{\mathbb{K}} \mathrm{q}_{\mathrm{path}}(\mathfrak{D}')$, which concludes the proof of (i).

For (ii), we distinguish two cases: $\mathfrak{D} \not\models_{\mathbb{K}} \mathrm{q}_{\mathrm{path}}$ or $\mathfrak{D} \models_{\mathbb{K}} \mathrm{q}_{\mathrm{path}}$. If $\mathfrak{D} \not\models_{\mathbb{K}} \mathrm{q}_{\mathrm{path}}$, then for every repair $\mathfrak{D}'$ of $\mathfrak{D}$, we have that $\mathfrak{D}' \not\models_{\mathbb{K}} \mathrm{q}_{\mathrm{path}}$, Hence, $\mathrm{q}_{\mathrm{path}}(\mathfrak{D}') = 0$, for every repair $\mathfrak{D}'$ of $\mathfrak{D}$. Now, an inspection for $e_{path}(\mathfrak{D})$ shows that $e_{path}(\mathfrak{D}) = 0$. Therefore, we can pick any repair $\mathfrak{D}^*$ of $\mathfrak{D}$ (at least one repair exists) and conclude that $e_{path}(\mathfrak{D}) = 0 = \mathrm{q}_{\mathrm{path}}(\mathfrak{D}^*)$.

Finally, suppose that $\mathfrak{D} \models_{\mathbb{K}} \mathrm{q}_{\mathrm{path}}$. Our goal is to show that there exists a repair $\mathfrak{D}^*$ of $\mathfrak{D}$ such that $e_{path}(\mathfrak{D}) = q(\mathfrak{D}^*)$. We build a repair $\mathfrak{D}^*$ as follows. For every element b that is a key value for $S^{\mathfrak{D}}$, we choose a value c^* such that $S^{\mathfrak{D}}(b,c^*) = \min\{S^{\mathfrak{D}}(b,c) \mid S^{\mathfrak{D}}(b,c) \neq 0, c \in D\}$. Now, for every element a that is a key value for $R^{\mathfrak{D}}$, we have two possibilities:

(i) There exists a b such that $R^{\mathfrak{D}}(a,b) \neq 0$, and $S^{\mathfrak{D}}(b,c) = 0$ for every $c \in D$ (i.e., b is not a key value for $S^{\mathfrak{D}}$). In this case, we pick one such b and put (a,b) in $R^{\mathfrak{D}^*}$.
(ii) For every b such that $R^{\mathfrak{D}}(a,b) \neq 0$, there exists a value c for which $S^{\mathfrak{D}}(b,c) \neq 0$. In this case, we choose an element b^* with $R^{\mathfrak{D}}(a,b^*) \neq 0$ and such that the value $R^{\mathfrak{D}}(a,b^*) \times S^{\mathfrak{D}}(b^*,c^*)$ is minimised, and we put (a,b^*) in $R^{\mathfrak{D}^*}$.

Let a be a key value for $R^{\mathfrak{D}}$ for which the first possibility holds. If b is the element for which $R^{\mathfrak{D}^*}(a,b)$ is a fact of $\mathfrak{D}^*$, then $\min\{S^{\mathfrak{D}}(b,c) : S^{\mathfrak{D}}(b,c) \neq 0, c \in D\} = 0$. Consequently, we have that

$$\min_{y \in D: R^{\mathfrak{D}}(a,y) \neq 0} (R^{\mathfrak{D}}(a,y) \times \min_{z \in D: S^{\mathfrak{D}}(y,z) \neq 0} S^{\mathfrak{D}}(y,z)) = 0.$$

Moreover, $R^{\mathfrak{D}^*}(a,b) \times S^{\mathfrak{D}^*}(b,c) = 0$, for any $c \in D^*$. Thus, the value of the summand in $e_{path}(\mathfrak{D})$ contributed by a equals the value of the summand in $\mathrm{q}_{\mathrm{path}}(\mathfrak{D}^*)$ contributed by a.

Let a be a key value such that the second possibility holds. From the definition of $\mathfrak{D}^*$, it follows that the expression $\min_{y \in D : R^{\mathfrak{D}}(a,y) \neq 0}(R^{\mathfrak{D}}(a,y) \times \min_{z \in D : S^{\mathfrak{D}}(y,z) \neq 0} S^{\mathfrak{D}}(y,z))$ in $e_{path}(\mathfrak{D})$ is equal to the value in $\mathrm{q}_{\mathrm{path}}(\mathfrak{D}^*)$ contributed by a.

By combining the findings in these two possibilities, we conclude that $e_{path}(\mathfrak{D}) = \mathrm{q}_{\mathrm{path}}(\mathfrak{D}^*)$. □

B.2 $\mathbb{K}$-circuits and complexity theory

Proposition 4.8. *For every naturally ordered positive semiring $\mathbb{K}$, the data complexity of $\mathcal{L}_{\mathbb{K}}$ is in* DLOGTIME*-uniform* $\mathsf{FnAC}^0_{\mathbb{K}}(+, \times_2, \min, \overline{\mathrm{Supp}})$.

Proof. The proof proceeds analogously to the classical case proving that FO is in DLOGTIME-uniform AC^0. When considered as inputs to circuits, $\mathbb{K}$-databases are encoded as strings of semiring values in the same fashion as Boolean databases are encoded as strings of Booleans (see, e.g., [30]). The inputs that we consider also include an assignment α giving values for some fixed finite set X of variables. If $R^{\mathfrak{D}}$ is a $\mathbb{K}$-relation of arity k in $\mathfrak{D}$, then its encoding $\mathrm{enc}(R^{\mathfrak{D}}, \alpha)$ is simply the concatenation of the semiring values $R^{\mathfrak{D}}(\alpha(\vec{a}))$, for $\vec{a} \in (D \cup X)^k$, written in some predefined order (that is we stipulate some ordering on $D \cup X$ and use that to define an ordering of $(D \cup X)^k$). The encoding $\mathrm{enc}(\mathfrak{D}, \alpha)$ of the $\mathbb{K}$-database and an assignment is then the concatenation of the encodings of its $\mathbb{K}$-relations $\mathrm{enc}(R^{\mathfrak{D}}, \alpha)$ in some predefined order.

Given $n \in \mathbb{N}$ and a formula $\varphi \in \mathcal{L}_{\mathbb{K}}$, one can recursively define the $\mathrm{AC}^0_{\mathbb{K}}(+, \times_2, \min)$-circuit that computes the value of φ in a $\mathfrak{D}, \alpha$, such that $|D| = n$, on the input $\mathrm{enc}(\mathfrak{D}, \alpha)$. In the transformation of $\varphi \in \mathcal{L}_{\mathbb{K}}$ to a circuit, it suffices to transform subformulas of the forms $\exists x.\varphi(\vec{y}, x)$ and $\nabla x.\varphi(\vec{y}, x)$ to expressions $\sum_{a \in D} \varphi(\vec{y}, a/x)$ and $\min_{a \in D} \varphi(\vec{y}, a/x)$, respectively. After this, the remaining construction of the circuit is to treat each subformula as a gate of the circuit labelled with its top-most connective. Gates corresponding to atomic formulas are input gates and are labelled with an appropriate part of the input $\mathrm{enc}(\mathfrak{D}, \alpha)$ determined by the ordering used in it. The argument that there is a DLOGTIME algorithm that describes C_n, given n, is the same as in the classical case for FO (see [8] for a similar proof for $\mathrm{FO}_{\mathbb{R}}$ and DLOGTIME-uniform $\mathrm{AC}^0_{\mathbb{R}}(+, \times)$). □

B.3 Acyclicity of the attack graph and semiring rewriting of the consistent answers

Lemma 4.13. *If q is a* sjfCQ *with an acyclic attack graph and x is an unattacked variable, then $q[x]$ has an acyclic attack graph. Moreover, the attack graph of $q[x]$ is a subgraph of the attack graph of q.*

Proof. Notice that the attack graphs of $q[x]$ and q have the same atoms, hence they have the same nodes. Consider an edge in the attack graph of $q[x]$, that is, an edge between the nodes corresponding to atoms $R(\vec{y}; \vec{z})$ and $R'(\vec{y'}; \vec{z'})$. Thus, $R(\vec{y}; \vec{z})$ attacks a variable y'_i in $\mathrm{var}(\vec{y'})$ that is bounded in $q[x]$. In particular, y'_i is bounded in q, and both atoms $R(\vec{y}; \vec{z})$ and $R'(\vec{y'}; \vec{z'})$ are in q. Since $R(\vec{y}; \vec{z})$ attacks y'_i, there are witnesses $x_1, \ldots, x_m = y'_i$ such that:

(1) $x_1 \in \mathrm{var}(\vec{z})$ (i.e., is a non-key variable of R), and $x_m = y'_i$;
(2) for all $i < n$ we have that x_i, x_{i+1} occur together in some atom of $q[x]$ (thus, they occur together in some atom of q); and
(3) For every $i \leq n$ we have that $x_i \notin \{\mathrm{var}(\vec{y})\}^+_{\Sigma(q[x] \setminus R)}$. Notice that the set of key-constraints in $q[x] \setminus \{R(\vec{y}; \vec{z})\}$ and the set of key constraints in $q \setminus \{R(\vec{y}; \vec{z})\}$ coincide .

We can conclude that $R(\vec{y}; \vec{z})$ attacks y'_i in the attack graph of q as well, in other words, the edge between $R(\vec{y}; \vec{z})$ and $R'(\vec{y'}; \vec{z'})$ is also in the attack graph of q, therefore the attack graph of $q[x]$ is a subgraph of the attack graph of q. □

Lemma 4.15. *Let q be a* sjfCQ *and Σ a set of key constraints, one per relation, with an acyclic attack graph. Let α :* Var $\rightarrow$ *A be an assignment and x be an unattacked variable in q. Then, for every $\mathbb{K}$-database $\mathfrak{D}$,*

$$\mathrm{mCA}_{\mathbb{K}}(q, \mathfrak{D}, \alpha) = \sum_{c \in D} \mathrm{mCA}_{\mathbb{K}}(q[x], \mathfrak{D}, \alpha(c/x)).$$

Proof. Note that, by definition

$$\mathrm{mCA}_{\mathbb{K}}(q, \mathfrak{D}, \alpha) = \min_{\mathfrak{D}' \in \mathrm{Rep}(\mathfrak{D}, \Sigma)} q(\mathfrak{D}', \alpha) = \min_{\mathfrak{D}' \in \mathrm{Rep}(\mathfrak{D}, \Sigma)} \sum_{c \in D'} q[x](\mathfrak{D}', \alpha(c/x)).$$

Hence, it suffices to establish that

$$\min_{\mathfrak{D}' \in \mathrm{Rep}(\mathfrak{D}, \Sigma)} \sum_{c \in D'} q[x](\mathfrak{D}', \alpha(c/x)) = \sum_{c \in D} \min_{\mathfrak{D}' \in \mathrm{Rep}(\mathfrak{D}, \Sigma)} q[x](\mathfrak{D}', \alpha(c/x)).$$

It is easy to see that

$$\min_{\mathfrak{D}' \in \mathrm{Rep}(\mathfrak{D}, \Sigma)} \sum_{c \in D'} q[x](\mathfrak{D}', \alpha(c/x)) \geq_{\mathbb{K}} \sum_{c \in D} \min_{\mathfrak{D}' \in \mathrm{Rep}(\mathfrak{D}, \Sigma)} q[x](\mathfrak{D}', \alpha(c/x)).$$

For the other direction, if $\mathrm{mCA}_{\mathbb{K}}(q, \mathfrak{D}, \alpha) = 0$, then the lemma holds. Suppose $\mathrm{mCA}_{\mathbb{K}}(q, \mathfrak{D}, \alpha) > 0$. We need to show that

$$\min_{\mathfrak{D}' \in \mathrm{Rep}(\mathfrak{D}, \Sigma)} \sum_{c \in D'} q[x](\mathfrak{D}', \alpha(c/x)) \leq_{\mathbb{K}} \sum_{c \in D} \min_{\mathfrak{D}' \in \mathrm{Rep}(\mathfrak{D}, \Sigma)} q[x](\mathfrak{D}', \alpha(c/x)). \tag{11}$$

For every repair $\mathfrak{R}$ of $\mathfrak{D}$, set $\rho(\mathfrak{R}) \coloneqq \{c \in D \mid \mathfrak{R}, \alpha(c/x) \models_{\mathbb{K}} q[x]\}$ and $v(\mathfrak{R}) \coloneqq \sum_{c \in R} q[x](\mathfrak{R}, \alpha(c/x))$. Since $\mathrm{mCA}_{\mathbb{K}}(q, \mathfrak{D}, \alpha) > 0$, which is the minimum $v(\mathfrak{R})$ over the repairs $\mathfrak{R}$ of $\mathfrak{D}$, we have that $v(\mathfrak{R}) > 0$ for every $\mathfrak{R}$. Let $m = \min_{\mathfrak{R} \in \mathrm{Rep}(\mathfrak{D}, \Sigma)} v(\mathfrak{R}) = \mathrm{mCA}_{\mathbb{K}}(q, \mathfrak{D}, \alpha)$ and fix $\mathfrak{R}_{\min} \in \mathrm{Rep}(\mathfrak{D}, \Sigma)$ such that $v(\mathfrak{R}_{\min}) = m$. We will show the following claim:

***Claim* 3.** *If $a \in \rho(\mathfrak{R}_{\min})$ and $\mathfrak{R} \in \mathrm{Rep}(\mathfrak{D}, \Sigma)$, then $q[x](\mathfrak{R}_{\min}, \alpha(a/x)) \leq_{\mathbb{K}} q[x](\mathfrak{R}, \alpha(a/x))$.*

Proof of Claim 3. Let $(R_1, R_2, \ldots, R_n)$ be a topological ordering of the attack graph of q. For simplicity, when we use a relation name R_i in contexts where an atom is expected, we mean the unique R_i-atom of q. If $\mathfrak{R}$ is a repair, and $i \in \{0, 1, 2, \ldots, n\}$, then we write $\mathrm{PreCopy}(\mathfrak{R}, i)$ for the smallest subset of $\mathrm{Rep}(\mathfrak{D}, \Sigma)$ that contains $\mathfrak{R}'$ whenever $\mathfrak{R}$ and $\mathfrak{R}'$ have exactly the same R_j-facts for all $j \in \{1, 2, \ldots, i\}$. Thus, $\mathrm{PreCopy}(\mathfrak{R}, 0) = \mathrm{Rep}(\mathfrak{D}, \Sigma)$, and $\mathrm{PreCopy}(\mathfrak{R}, n) = \{\mathfrak{R}\}$. We introduce some convenient terminology. An *R_ℓ-block* is a maximal set of R_ℓ-facts of $\mathrm{Supp}(\mathfrak{D})$ that agree on all key attributes. Clearly, for each $\ell \in \{1, 2, \ldots, n\}$, every repair selects exactly one fact from each R_ℓ-block. An *embedding of q into $\mathfrak{R}$* is an assignment that maps every atom of q to a fact in $\mathrm{Supp}(\mathfrak{R})$. If θ is an embedding, then $\theta(R_\ell)$ denotes the fact to which the atom R_ℓ is mapped (where $\ell \in \{1, 2, \ldots, n\}$). A repair $\mathfrak{R}$ is called *superfrugal* [2] if no repair $\mathfrak{R}'$ has a set of embeddings that is a strict subset of that of $\mathfrak{R}$. An embedding in a superfrugal repair is also called a $\forall$*embedding*. If β is a $\forall$embedding, then, for all $i \in \{1, 2, \ldots, n\}$, the R_i-block that contains $\beta(R_i)$ is called a *$\forall R_i$-block*. From [2, Lemma 4.5], it follows that for all $i \in \{1, 2, \ldots, n\}$, we can select a fact from each block that is not a $\forall R_i$-block such that the selected facts do not belong to any embedding into the resulting repair. Therefore, in what follows, it suffices to consider only $\forall R_i$-blocks.

A repair $\mathfrak{R}$ such that $v(\mathfrak{R}) = m$ can be constructed as follows, for decreasing values of $i = n, n-1, \ldots, 1$:

- for $i = n$, select in each $\forall R_n$-block the fact with the smallest annotated semiring value; and
- for $i < n$, in each $\forall R_i$-block, select the fact that yields the smallest annotated semiring value, given the already fixed R_j-facts for $j \in \{i+1, i+2, \ldots, n\}$. This selection is illustrated next and, as we will argue shortly, is independent of the R_ℓ-facts for $\ell < i$.

Example B.1. Consider the bag semiring. Assume that the last two facts of q, in the topological order of the attack graph, are $R_{n-1}(v; y)$ and $R_n(y, z)$, where all attributes of R_n are key attributes. Assume that the only R_n-facts with positive support are $R_n^{\mathfrak{D}}(b, d_1) = 1, R_n^{\mathfrak{D}}(b, d_2) = 1$, and $R_n^{\mathfrak{D}}(c, d_1) = 8$. Furthermore, assume that $R_{n-1}^{\mathfrak{D}}(a, b) = 2$ and $R_{n-1}^{\mathfrak{D}}(a, c) = 1$, two facts belonging to the same $\forall R_{n-1}$-block. The former R_{n-1}-fact contributes $(2 \times 1) + (2 \times 1) = 4$, while the latter contributes $1 \times 8 = 8$. So our procedure will select $R_{n-1}(a, b)$, which, notably, is not the fact with the smallest annotated semiring value in its block.

It can now be argued by induction on decreasing $i = n, n-1, \ldots, 1$ that every superfrugal repair $\mathfrak{R}$ containing the selected R_j-facts for every $j \in \{i, i+1, \ldots, n\}$ minimizes the semiring value of q across all repairs in $\text{PreCopy}(\mathfrak{R}, i-1)$. The reasoning is similar to that in Claim 2. Assume that in some superfrugal repair $\mathfrak{R}$, we replace a fact B with C in some $\forall R_i$-block $\mathscr{B}$, such that the semiring value of q decreases. Let $\mathfrak{R}_C$ be the repair satisfying $\text{Supp}(\mathfrak{R}_C) = (\text{Supp}(\mathfrak{R}) \setminus \{B\}) \cup \{C\}$. There will be an embedding β into $\mathfrak{R}$ such that $\beta(R_i) = B$, as well as an embedding γ into $\mathfrak{R}_C$ such that $\gamma(R_i) = C$. Since B and C agree on their key, it follows that β and γ agree on every variable in $V := \{v \in \text{var}(q) \mid \Sigma(q \setminus \{R_i\}) \models \text{key}(R_i) \rightarrow v\}$. Define θ_β^γ as the assignment such that for every variable v in $\text{var}(q)$,

$$\theta_\gamma^\beta(v) = \begin{cases} \gamma(v) & \text{if } R_i \text{ attacks } v; \\ \beta(v) & \text{otherwise.} \end{cases}$$

Since x is unattacked in q, we have $\theta_\gamma^\beta(x) = \beta(x)$. Notably, θ_γ^β is an embedding into $\mathfrak{R}_C$ because if an atom R_ℓ of q (where $\ell \in \{1, 2, \ldots, n\}$) contains both variables attacked and unattacked by R_i, then β and γ agree on the unattacked variables, as they belong to V; consequently, $\theta_\gamma^\beta(R_\ell) = \gamma(R_\ell)$. Furthermore, if $\ell \neq i$ and R_ℓ is not attacked by R_i, then $\theta_\gamma^\beta(R_\ell) = \beta(R_\ell)$, which, in particular, holds for $\ell \in \{1, 2, \ldots, i-1\}$. Note also that x occurs in at least one atom that is not attacked by R_i. Informally, we conclude that if B is replaced by C, then *every* embedding into $\mathfrak{R}$ that used B can be transformed into one that uses C instead, while remaining unchanged over all atoms not attacked by R_i. Consequently, the optimal (minimizing) choice from $\mathscr{B}$ is independent of atoms not attacked by R_i. In particular, regarding x, if β_1 and β_2 are distinct embeddings into $\mathfrak{R}$ such that $\beta_1(R_i) = \beta_2(R_i) = B$, then the optimal choice in $\mathscr{B}$ is the same for both β_1 and β_2, even if $\beta_1(x) \neq \beta_2(x)$. For example, the selection from the R_{n-1}-block in Example B.1 does not require knowledge of the R_ℓ-facts for $\ell < n-1$.

When i reaches 1, the superfrugal repair with the selected facts minimizes the semiring value of q across all repairs of $\text{PreCopy}(\mathfrak{R}, 0) = \text{Rep}(\mathfrak{D}, \Sigma)$. Moreover, our minimization procedure ensures that for every $a \in D$, the semiring value of $q[x]$ with respect to $\alpha(a/x)$ is also minimized. This concludes the proof of Claim 3. □

Equation (11) now follows from Claim 3, and this finishes the proof of the lemma. □

Received December 2024; revised February 2025; accepted March 2025