Theory Comput Syst (2011) 49:429–459
DOI 10.1007/s00224-011-9327-6

Containment of Conjunctive Queries on Annotated Relations

Todd J. Green

Published online: 30 April 2011

Abstract We study containment and equivalence of (unions of) conjunctive queries on relations annotated with elements of a commutative semiring. Such relations and the semantics of positive relational queries on them were introduced in a recent paper as a generalization of set semantics, bag semantics, incomplete databases, and databases annotated with various kinds of provenance information. We obtain positive decidability results and complexity characterizations for databases with lineage, why-provenance, and provenance polynomial annotations, for both conjunctive queries and unions of conjunctive queries. At least one of these results is surprising given that provenance polynomial annotations seem "more expressive" than bag semantics and under the latter, containment of unions of conjunctive queries is known to be undecidable. The decision procedures rely on interesting variations on the notion of containment mappings. We also show that for any positive semiring (a very large class) and conjunctive queries without self-joins, equivalence is the same as isomorphism.

Keywords Database theory · Provenance · Query optimization

A preliminary version of this paper [17] appeared in the Proceedings of the 12th International Conference on Database Theory (ACM International Conference Proceeding Series, 361 ACM 2009, ISBN 978-1-60558-423-2).

Our work was supported by the National Science Foundation under grants IIS-0447972, 0513778, and 0629846.

Work performed while at the University of Pennsylvania.

T.J. Green (✉)
University of California, Davis, CA 95616, USA
e-mail: green@cs.ucdavis.edu

1 Introduction

We study containment and equivalence of conjunctive queries (CQs) and unions of conjunctive queries (UCQs) under a variety of database semantics, including the classical set semantics and the bag semantics used in practical DBMS implementations, as well as a number of semantics involving *provenance* (aka *lineage*) data annotations. Our goal is to provide a general framework for studying all of these problems and use this framework to draw precise connections among them.

The unifying foundation for our study is that of *K-relations*, which are relations whose tuples are annotated with elements from a commutative semiring K. These were introduced in a recent paper [20] as a generalization of sets, bags, the Boolean c-tables used in incomplete databases [21, 23], probabilistic databases [15, 35], databases with lineage [13] or why-provenance [4] information, and other kinds of annotated relations. The semantics of positive relational algebra queries extends to K-relations via definitions in terms of the abstract "$+$" and "$\cdot$" operations of K. For $K = \mathbb{B}$, the Boolean semiring, this specializes to the usual set semantics, while for $K = \mathbb{N}$, the semiring of natural numbers, it is bag semantics.

The introduction of annotations on relations presents new challenges in query reformulation and optimization, however, as queries that are semantically *equivalent* when posed over ordinary relations may become *inequivalent* when posed over K-relations. Indeed, this phenomenon was already observed for the case of bag semantics [8, 24], where, e.g., adding a "redundant" self-join to a query actually changes the query's meaning. The need to compare query equivalence for different kinds of provenance annotations was also emphasized from early on in [4, 5] and reiterated in [3]. A central theme of this paper is to compare different provenance-annotated semantics among themselves and with the standard set and bag semantics. The comparison is done w.r.t. containment[1] and equivalence of conjunctive queries (CQs) and unions of conjunctive queries (UCQs), leading to four different hierarchies among these semantics. Whether the steps in these hierarchies are strict or not is always informative and sometimes surprising.

We consider in this paper five different kinds of provenance information that can be captured using semiring annotations. These range from the very simple data warehousing *lineage* of [13], in which a tuple in the output is annotated with a set of tuple ids of all "contributing" source tuples, to the *why-provenance* of [4], in which output tuples are annotated with a *set of sets* of contributing source tuples, to the *provenance polynomials* $\mathbb{N}[X]$ of [20], in which the annotations are polynomial expressions over the source tuple ids which fully "document" how an output tuple is produced in the result of a query. Provenance polynomials are as "general" as any other commutative semiring, hence this is the most informative form of provenance annotations. $\mathbb{N}[X]$-relations are not just of theoretical interest, but also have practical applications as the foundation of trust policies and incremental maintenance algorithms in systems for collaborative data sharing [19]. We also consider a new form of provenance, the *Boolean provenance polynomials* $\mathbb{B}[X]$, as well as the form of lineage used in the

[1] We define inclusion of K-relations by the *natural order* present in the semirings of interest to us (see Sect. 3).

Trio project [31], which we show can also be captured using a semiring. These two forms of provenance are intermediate between why-provenance and $\mathbb{N}[X]$.

To illustrate three of the models, for the annotated source relation R and query Q

$$R \overset{\text{def}}{=} \begin{array}{|ccc|c|}\hline \mathrm{A} & \mathrm{B} & \mathrm{C} & \\ \hline \mathrm{a} & \mathrm{b} & \mathrm{c} & p \\ \mathrm{d} & \mathrm{b} & \mathrm{e} & r \\ \mathrm{f} & \mathrm{g} & \mathrm{e} & s \\ \hline \end{array} \qquad Q \overset{\text{def}}{=} \pi_{\mathrm{AC}}\big(\pi_{\mathrm{AB}} R \bowtie \pi_{\mathrm{BC}} R \ \cup \ \pi_{\mathrm{AC}} R \bowtie \pi_{\mathrm{BC}} R\big)$$

(where p, r, and s are the tuple ids) the data warehousing lineage of (d, e) in the output is $\{r, s\}$, the why-provenance of (d, e) is $\{\{r\}, \{r, s\}\}$, and the $\mathbb{N}[X]$-provenance is $2r^2 + rs$. Intuitively, lineage tells us *which* source tuples were involved in producing a given output tuple—in this case, the source tuples labeled r and s; why-provenance tells us which *sets* of source tuples were involved in producing the output tuple—here either r alone, or r with s, can be used to produce the output tuple; and $\mathbb{N}[X]$-provenance tells us exactly *how* the tuple was produced from the source tuples—here, there are two derivations of the output tuple involving a self-join of r with itself (hence the $2r^2$ term), and one involving a join of r with s (corresponding to the rs term). Additionally, note that by "plugging in" numeric values for the variables (e.g., $p = 2, r = 3, s = 1$) and evaluating the $\mathbb{N}[X]$-provenance of an output tuple, we obtain the multiplicity of the tuple under bag semantics (e.g., $2 \cdot 3^2 + 1 = 19$ for (d, e)).

Another central theme of this paper is to establish the complexity of containment and equivalence of CQs/UCQs for various semirings. For the semirings $\mathbb{B}$ (resp., $\mathbb{N}$) this corresponds to set (resp., bag) semantics and the questions were studied in the past [6, 8, 30] as was the case of bag-set semantics [11, 29] (In Sect. 7.5 we discuss the relationship between the latter and our results.) Results for an entire class of semirings (the distributive lattices) have already been established in [16, 20]. This paper focuses primarily on the provenance semirings.

A priori, it is not clear that containment and equivalence for queries on relations with provenance annotations should even be decidable, as bag containment is known to be undecidable for UCQs [24], and $\mathbb{N}[X]$ seems related to bags. Nevertheless, we are able to show that containment is decidable for all the forms of provenance annotations we consider, for both CQs and UCQs. We also establish interesting connections with the same problems for bag semantics. In particular our contributions are:

- We show that the various forms of provenance annotations we consider are related by *surjective semiring homomorphisms*, which yields easy bounds on their relative behavior with respect to query containment.
- We show that for UCQs, $\mathbb{N}[X]$- containment implies K-containment for *any* semiring K, and for any *positive* K (a very large class that includes all the semirings we consider in this paper, see Sect. 3), K-containment implies containment under the usual set semantics.

		$\mathbb{B}$	$\mathsf{PosBool}(X)$	$\mathsf{Lin}(X)$	$\mathsf{Why}(X)$	$\mathsf{Trio}(X)$	$\mathbb{B}[X]$	$\mathbb{N}[X]$	$\mathbb{N}$
CQs	cont	NP	NP	NP	NP	NP	NP	NP	? (Π_2^p-hard)
	equiv	NP	NP	NP	GI	GI	GI	GI	GI
UCQs	cont	NP	NP	NP	NP	in PSPACE	NP	in PSPACE	undec
	equiv	NP	NP	NP	NP	GI	NP	GI	GI

Fig. 1 Complexity of containment and equivalence. *Non-shaded boxes* indicate contributions of this paper. NP is short for NP-complete. GI is short for GI-complete (i.e., complete for the class of problems polynomial time reducible to graph isomorphism)

- For the case of CQs without self-joins, we show that for any positive K, K-equivalence is the same as *isomorphism*, and thus its complexity is complete for the class GI of problems polynomial time reducible to *graph isomorphism*.[2]
- We show that containment of CQs and UCQs is decidable for lineage, why-provenance, $\mathbb{B}[X]$, and $\mathbb{N}[X]$ annotations. The decision procedures involve interesting variations on the concept of *containment mappings*, or (in the case of $\mathbb{N}[X]$-containment and $\mathsf{Trio}(X)$-containment of UCQs) establishing a *small counterexample property* (see Sects. 7.4 and 7.6). We also identify the complexity in each case as NP-complete (with the exception of $\mathbb{N}[X]$-containment and $\mathsf{Trio}(X)$-containment of UCQs, where we give PSPACE upper bounds).
- We show that for why-provenance, $\mathbb{B}[X]$, and $\mathbb{N}[X]$, equivalence of CQs implies isomorphism, and the complexity is therefore somewhat lower than for containment (GI-complete). $\mathbb{N}[X]$-equivalence of UCQs is also shown to be the same as isomorphism and GI-complete. Lineage-equivalence of CQs and why-prov. and $\mathbb{B}[X]$-equivalence of UCQs are shown to remain NP-complete.
- We show that for CQs, why-prov. containment implies bag-containment, and bag containment implies lineage-containment. We also show that for UCQs $\mathbb{N}[X]$-equivalence is the same as bag equivalence hence providing a proof that the latter is the same as isomorphism and therefore GI-complete.

Figure 1 summarizes the complexity results mentioned above (for completeness we include previously known results in the shaded boxes). Figure 2 summarizes the logical relationships for containment/equivalence among the various semirings we consider.

The rest of this paper is organized as follows. We define K-relations and the semantics of queries on them in Sect. 2. We define the various semirings for provenance in Sect. 4; we also establish there the existence of semiring homomorphisms relating the various models. We define containment of queries on K-relations in terms of the *natural order* in Sect. 3 and discuss the connections with semiring homomorphisms. We review the background concepts of containment mappings and canonical databases in Sect. 5. We derive the bounds on containment based on surjective semiring homomorphisms in Sect. 6. We present the main results on containment and equivalence in Sect. 7. We discuss related work in Sect. 8. Finally, we conclude with some ideas for future work in Sect. 9.

[2]Graph isomorphism is known to be in NP, but is not known or believed to be either NP-complete or in PTIME, see [26].

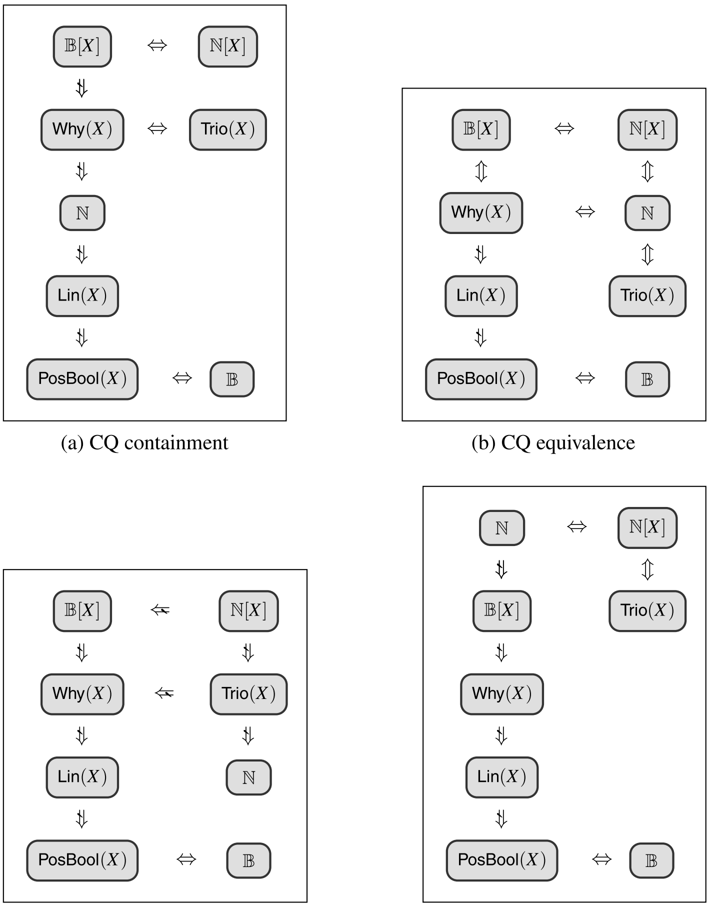

Fig. 2 Logical implications of containment and equivalence. $K_1 \Rightarrow K_2$ indicates that K_1-containment (equivalence) implies K_2-containment (equivalence). A ticked arrow "$\not\Rightarrow$" indicates that the implication is strict

2 Queries on K-Relations

Fix a countable domain $\mathbb{D}$ of constants, denoted by $a, b, c, \ldots$. Let $(K, +, \cdot, 0, 1)$ be a *commutative semiring*, i.e., $(K, +, 0)$ and $(K, \cdot, 1)$ are commutative monoids, $\cdot$ is distributive over $+$ and $\forall a$, $0 \cdot a = a \cdot 0 = 0$. An n-ary K*-relation* is a function $R : \mathbb{D}^n \to K$ such that its *support* defined by $\mathsf{supp}(R) \overset{\text{def}}{=} \{t : t \in \mathbb{D}^n, R(t) \neq 0\}$ is finite. This is a generalization of the usual notion of a relation to include semiring annotations. We call a value $t \in \mathbb{D}^n$ an n*-tuple* (or simply a *tuple*). If R is an n-ary K-relation and t is an n-tuple, we call the value $R(t) \in K$ the *annotation* of t in R. A K*-instance* is a mapping from predicate symbols to K-relations. This generalizes the usual notion of database instances to work with K-relations. If $\mathfrak{A}$ is a K-instance and S is a predicate symbol, we denote by $S^{\mathfrak{A}}$ the value of S in $\mathfrak{A}$. We will sometimes abuse notation by using $R, S, \ldots$ both for predicate symbols and for the K-relations they represent.

Example 2.1 We show below a simple example of an $\mathbb{N}$-instance $\mathfrak{A}$, where $\mathbb{N}$ is the semiring of natural numbers:

$$R^{\mathfrak{A}} \stackrel{\text{def}}{=} \begin{array}{|cc|c|}\hline \text{a} & \text{b} & 2 \\ \text{d} & \text{b} & 1 \\ \text{b} & \text{c} & 7 \\ \hline\end{array} \qquad S^{\mathfrak{A}} \stackrel{\text{def}}{=} \begin{array}{|ccc|c|}\hline \text{b} & \text{g} & \text{f} & 3 \\ \text{d} & \text{a} & \text{b} & 1 \\ \hline\end{array}$$

In the example, $R^{\mathfrak{A}}$ and $S^{\mathfrak{A}}$ are $\mathbb{N}$-relations, and the annotation of (a, b) in $R^{\mathfrak{A}}$ is $R^{\mathfrak{A}}(\text{a}, \text{b}) = 2$. (The annotation of any tuple not shown in a table is understood to be 0.)

We use Datalog-style syntax for conjunctive queries and unions of conjunctive queries. A *conjunctive query* (CQ) is an expression of the form

$$Q(\bar{u}) \text{ :- } R_1(\bar{u}_1), \ldots, R_n(\bar{u}_n)$$

where $Q(\bar{u})$ is the *head* of the query, denoted $\mathsf{head}(Q)$, the *multiset* (bag) of *atoms* $R_1(\bar{u}_1), \ldots, R_n(\bar{u}_n)$ is the *body* of the query, denoted $\mathsf{body}(Q)$, $\bar{u}$ is the tuple of *distinguished variables* and constants, $\bar{u}_1, \ldots, \bar{u}_n$ are tuples of variables and constants whose arities are consistent with their associated predicate symbols, and each variable appearing in the head also appears somewhere in the body. We denote the set of variables appearing in Q by $\mathsf{vars}(Q)$ and the set of constants by $\mathsf{consts}(Q)$. When $\bar{u}$ is empty we say that Q is a *Boolean conjunctive query*; for these we will sometimes drop the parentheses in the head and write Q :- $R_1(\bar{u}_1), \ldots, R_n(\bar{u}_n)$. We say that a CQ has a *self-join* if some predicate symbol appears more than once in the body of a CQ. The *degree* of a CQ Q is the maximum number of occurrences of any single predicate symbol in $\mathsf{body}(Q)$.

A *union of conjunctive queries* (UCQ) is a bag $\bar{Q} = (Q_1, \ldots, Q_n)$ of CQs. The arities of the heads of the CQs in a UCQ must all agree. The *degree* of a UCQ $\bar{Q}$ is the maximum degree of a CQ in $\bar{Q}$.

The semantics of CQs on K-relations is based on the notion of *valuations*. A valuation is a function $\nu : \mathsf{vars}(Q) \to \mathbb{D}$ extended to be the identity on constants. Valuations operate component-wise on tuples in the expected way. Let Q be a CQ

$$Q(\bar{u}) \text{ :- } R_1(\bar{u}_1), \ldots, R_n(\bar{u}_n)$$

and let $\mathfrak{A}$ be a K-instance of the same schema. The *result of evaluating Q on* $\mathfrak{A}$ is the K-relation defined

$$[\![Q]\!]^{\mathfrak{A}}(t) \stackrel{\text{def}}{=} \sum_{\nu \text{ s.t.} \nu(\bar{u})=t} \prod_{i=1}^{n} R_i{}^{\mathfrak{A}}(\nu(\bar{u}_i)) \tag{2.1}$$

and the sums and products are in K. A valuation ν which maps $\bar{u}$ to t such that the product in (2.1) is non-zero is called a *derivation* of t, and we say that it *justifies* the associated product. The meaning of (2.1) is unchanged if we assume the sum ranges only over derivations of t.

We extend the semantics to UCQs as follows. If $\bar{Q} = (Q_1, \dots, Q_n)$ is a UCQ, then the *result of evaluating* $\bar{Q}$ on a K-instance $\mathfrak{A}$ is the K-relation defined

$$[\![\bar{Q}]\!]^{\mathfrak{A}}(t) \overset{\text{def}}{=} \sum_{i=1}^{n} [\![Q_i]\!]^{\mathfrak{A}}(t) \tag{2.2}$$

For the commutative semiring $(\mathbb{B}, \vee, \wedge, \mathsf{false}, \mathsf{true})$ this specializes to the set semantics for UCQs. For $(\mathbb{N}, +, \cdot, 0, 1)$ it is bag semantics. For $(\mathsf{PosBool}(X), \vee, \wedge, \mathsf{false}, \mathsf{true})$ (see Sect. 4) it is the positive Boolean c-tables used in incomplete databases [23].

A subtlety in the preceding definitions is that we allow the same atom to appear multiple times in the body of a CQ (and similarly, we allow the same CQ to appear multiple times in a UCQ). With set semantics the distinction is immaterial, but for other K, where idempotence of multiplication (and addition) may not hold, the distinction does matter. The classic example is adding a "redundant" self-join to a query in the case of $K = \mathbb{N}$.

In contrast to repetitions, the order of atoms in the body of a CQ (and order of CQs in a UCQ) is not important, since we are considering only K-relations where K is *commutative* (cf. Proposition 3.4 in [20]). Thus the body of a CQ can be viewed a *bag* of atoms. When comparing the bodies of CQs, we will use the notation $\mathsf{body}(P) \leq_{\mathbb{N}} \mathsf{body}(Q)$ to mean bag containment of the query bodies. We will also identify queries which are the same up to reordering of atoms in the body, i.e., $P = Q$ means $\mathsf{head}(P) = \mathsf{head}(Q)$, $\mathsf{body}(P) \leq_{\mathbb{N}} \mathsf{body}(Q)$, and $\mathsf{body}(Q) \leq_{\mathbb{N}} \mathsf{body}(P)$.

We use the notation $P \cong Q$ ($\bar{P} \cong \bar{Q}$) to denote that P and Q ($\bar{P}$ and $\bar{Q}$) are *isomorphic*, i.e., syntactically identical up to renaming of variables and reordering of terms (and, for UCQs, reordering of CQs).

3 The Natural Order

We define containment of K-relations and queries over K-instances in terms of the *natural order*. Let $(K, +, \cdot, 0, 1)$ be a semiring and define $a \leq b \overset{\text{def}}{\iff} \exists c\ a + c = b$. When $\leq$ is a partial order we say that K is *naturally-ordered*. The semirings $\mathbb{B}, \mathbb{N}$, $\mathsf{PosBool}(X)$ are naturally ordered, as are all of the semirings for provenance from Sect. 4. For $\mathsf{PosBool}(X)$ the natural order corresponds to logical entailment: $\varphi \leq \psi$ iff $\varphi \models \psi$. For $\mathbb{B}[X]$ we have $a \leq b$ iff every monomial in a also appears in b. For $\mathbb{N}[X]$ we have $a \leq b$ iff every monomial in a also appears in b with an equal or greater coefficient. Thus, $2x^2y \leq 5x^2y + 2z$, but $x + 2y \not\leq 5x + 3y^2$. For lineage and why-provenance the natural order corresponds to set inclusion (n.b. for why-provenance, this is only set inclusion "at the outer level"—e.g., $\{\{x\}\} \leq \{\{x\}, \{y, z\}\}$ but $\{\{x\}, \{y, z\}\} \not\leq \{\{x, y\}, \{y, z\}\}$).

Definition 3.1 Let K be a naturally-ordered semiring and let R_1, R_2 be two K-relations. We define containment of R_1 in R_2 by

$$R_1 \leq_K R_2 \overset{\text{def}}{\iff} \forall t \; R_1(t) \leq R_2(t)$$

We define containment of queries P, Q with respect to K-relation semantics by

$$P \sqsubseteq_K Q \overset{\text{def}}{\iff} \text{for every } K\text{-instance } \mathfrak{A},\ [\![P]\!]^{\mathfrak{A}} \leq_K [\![Q]\!]^{\mathfrak{A}}$$

When $K = \mathbb{B}$ (resp., $K = \mathbb{N}$) we get the usual notion of query containment with respect to set (resp., bag) semantics. For $\mathsf{PosBool}(X)$, we get the *structural containment* and *structural equivalence* of [32].[3]

Definition 3.2 (Semiring homomorphism) Let K_1, K_2 be semirings. A mapping $h : K_1 \to K_2$ is called a *semiring homomorphism* if $h(0) = 0$, $h(1) = 1$, and for all $a, b \in K_1$, we have $h(a + b) = h(a) + h(b)$ and $h(a \cdot b) = h(a) \cdot h(b)$.

Proposition 3.3 *Let K_1, K_2 be naturally-ordered commutative semirings. If $h : K_1 \to K_2$ is a semiring homomorphism then for all $a, b \in K_1$, $a \leq_{K_1} b \implies h(a) \leq_{K_2} h(b)$. If h is also surjective, then for all $a, b \in K_1$, $a \leq_{K_1} b \iff h(a) \leq_{K_2} h(b)$.*

Proof Straightforward calculation. □

4 Semirings for Provenance

In this section we define several kinds of provenance annotations that can be captured in the semiring framework. We will also observe that the various models are related by *surjective semiring homomorphisms*, as summarized in Fig. 3. In Sect. 6, we will use the existence of surjective semiring homomorphisms to establish some basic relationships among the provenance models with respect to query containment.

We fix a countable set X of *variables*, which can be thought of as *tuple identifiers*, and parametrize all of the provenance models by this set X.

The most informative form of provenance annotations in the framework of K-relations is the semiring of *provenance polynomials* [20]:

Definition 4.1 (Provenance Polynomials) The *provenance polynomials semiring* for X is the semiring of polynomials with variables from X and coefficients from $\mathbb{N}$, with the operations defined as usual: $(\mathbb{N}[X], +, \cdot, 0, 1)$.

The provenance polynomials are the "most informative" among semiring annotation by dint of their *universality*: any function $\nu : X \to K$ (call it a "valuation") can

[3]There are reasonable alternatives to the natural order for incomplete databases, such as considering various orders on the sets of *possible worlds* they represent.

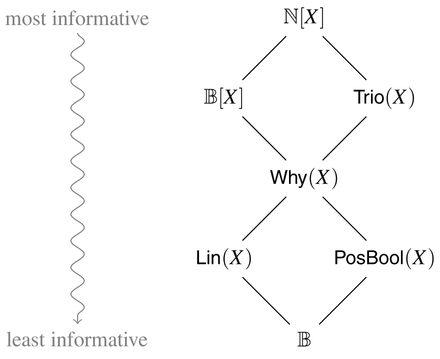

Fig. 3 Provenance hierarchy. A path downward from K_1 to K_2 indicates that there exists a surjective semiring homomorphism $h : K_1 \to K_2$

be extended uniquely to a semiring homomorphism $\mathsf{Eval}_\nu : \mathbb{N}[X] \to K$. Formally, we define Eval_ν recursively as follows:

- $\mathsf{Eval}_\nu(x) \overset{\text{def}}{=} \nu(x)$
- $\mathsf{Eval}_\nu(F + G) \overset{\text{def}}{=} \mathsf{Eval}_\nu(F) + \mathsf{Eval}_\nu(G)$
- $\mathsf{Eval}_\nu(F \cdot G) \overset{\text{def}}{=} \mathsf{Eval}_\nu(F) \cdot \mathsf{Eval}_\nu(G)$

where $+$ and $\cdot$ in the right-hand side of the last two equations are the operations in K. Intuitively, Eval_ν operates by assigning the value $\nu(x)$ to each variable x in a polynomial expression, then evaluating the resulting expression in K. For instance, taking $K = \mathbb{N}$, if ν maps p to 2 and r to 3 (and all other values to 0), then $\mathsf{Eval}_\nu(2r^2 + rs) = 2 \cdot 3 \cdot 3 + 1 = 19$, as in the example from the introduction.

Combined with the commutation with homomorphisms property (cf. Proposition 6.1), this allows the computations for any commutative semiring K to *factor* through the computations for the provenance polynomials (see [20]). We shall make use of this special ability of $\mathbb{N}[X]$ to "simulate" computations later in Sect. 7.4 (cf. Lemma 7.15). In Sect. 7.6, we shall note the failure of this property for $\mathsf{Trio}(X)$, and how a weaker version of it can nevertheless be recovered for that semiring.

To illustrate, consider the $\mathbb{N}[X]$-relation R in Fig. 4(a) and consider the UCQ $\bar{Q}$ defined by

$$\bar{Q}(x,z) \text{ :- } R(x,y,u),\, R(v,y,z)$$
$$\bar{Q}(x,z) \text{ :- } R(x,u,z),\, R(v,y,z)$$

Figure 4(b) shows the result of $\bar{Q}$ applied to R.

The second provenance model we consider is obtained from the provenance polynomials by replacing natural number coefficients with Boolean coefficients:

Definition 4.2 (Boolean Provenance Polynomials) The *Boolean provenance polynomials semiring* for X is the semiring of polynomials over variables X with Boolean coefficients: $(\mathbb{B}[X], +, \cdot, 0, 1)$.

Considering the same UCQ $\bar{Q}$ as before, Fig. 4(c) shows the result of applying $\bar{Q}$ to R, where R is interpreted as a $\mathbb{B}[X]$-relation. Note that the annotations in Fig. 4(c)

a	b	c	p
d	b	e	r
f	g	e	s

(a) Source R

a	c	$2p^2$
a	e	pr
d	c	pr
d	e	$2r^2+rs$
f	e	$2s^2+rs$

(b) $\mathbb{N}[X]$

a	c	p^2
a	e	pr
d	c	pr
d	e	r^2+rs
f	e	s^2+rs

(c) $\mathbb{B}[X]$

a	c	$2p$
a	e	pr
d	c	pr
d	e	$2r+rs$
f	e	$2s+rs$

(d) $\mathsf{Trio}(X)$

a	c	$\{\{p\}\}$
a	e	$\{\{p,r\}\}$
d	c	$\{\{p,r\}\}$
d	e	$\{\{r\},\{r,s\}\}$
f	e	$\{\{s\},\{r,s\}\}$

(e) $\mathsf{Why}(X)$

a	c	p
a	e	$p\wedge r$
d	c	$p\wedge r$
d	e	r
f	e	s

(f) $\mathsf{PosBool}(X)$

a	c	$\{p\}$
a	e	$\{p,r\}$
d	c	$\{p,r\}$
d	e	$\{r,s\}$
f	e	$\{r,s\}$

(g) $\mathsf{Lin}(X)$

Fig. 4 Provenance annotations

can be obtained from those in Fig. 4(b) by simply dropping the numeric coefficients. In fact, one can check that the operation $f:\mathbb{N}[X]\to\mathbb{B}[X]$ which "drops coefficients" (i.e., by replacing non-zero coefficients with true) is a *surjective semiring homomorphism*.

The third provenance model we consider, $\mathsf{Trio}(X)$, is inspired by the form of lineage used in the Trio project [31]. Like $\mathbb{B}[X]$, this semiring can be viewed as being obtained from $\mathbb{N}[X]$, but instead of "dropping coefficients," this time we "drop exponents." We formalize this using the notion of *quotient semirings*:

Definition 4.3 (Congruence relation) If K is a semiring and $\approx$ is an equivalence relation on K, then we say that $\approx$ is a *congruence relation* on K if $a\approx a'$ and $b\approx b'$ implies $a+b\approx a'+b'$ and $a\cdot b\approx a'\cdot b'$.

Definition 4.4 (Quotient semiring) Let K be a semiring and let $\approx$ be a congruence relation on K. If $a\in K$ then denote the equivalence class of a in $\approx$ by $a/{\approx}$. Then the *quotient of K by* $\approx$ is the semiring whose domain is the set $K/{\approx}$ of equivalence classes of $\approx$, $0\stackrel{\text{def}}{=}0_K/{\approx}$, $1\stackrel{\text{def}}{=}1_K/{\approx}$, $(a/{\approx})+(b/{\approx})=(a+b)/{\approx}$, and $(a/{\approx})\cdot(b/{\approx})\stackrel{\text{def}}{=}(a\cdot b)/{\approx}$.

Now, let $f:\mathbb{N}[X]\to\mathbb{N}[X]$ be the mapping that "drops exponents", e.g., f maps $2x^2y+3xy+2z^3+1$ to $5xy+2z+1$. Denote by $\approx_f$ the equivalence relation on $\mathbb{N}[X]$ defined by $a\approx_f b \stackrel{\text{def}}{\iff} f(a)=f(b)$. One can check that $\approx_f$ is a congruence relation. This justifies the following:

Definition 4.5 (Trio Semiring) The *Trio semiring* for X is the quotient semiring of $\mathbb{N}[X]$ by $\approx_f$, denoted $\mathsf{Trio}(X)$.

As an example, considering again the same UCQ $\bar{Q}$, Fig. 4(d) shows the result of applying $\bar{Q}$ to R, where R is interpreted as a $\mathsf{Trio}(X)$-relation, and an annotation A is understood to represent its equivalence class $A/{\approx_f}$ in $\approx_f$. Note that the mapping $h : \mathbb{N}[X] \to \mathsf{Trio}(X)$ defined by $h(A) \mapsto A/{\approx_f}$ is a surjective semiring homomorphism.

The fourth provenance model we consider is the *why-provenance* of [4]. The why-provenance of a tuple is the *set of sets* of "contributing" source tuples, which is called the *proof witness basis* in [4]. This can be captured using a semiring [3] (called the *proof why-provenance* semiring in [3]):

Definition 4.6 (Why-Provenance) The *why-provenance semiring* for X is $(\mathsf{Why}(X), \cup, \Cup, \emptyset, \{\emptyset\})$ where $\mathsf{Why}(X) \overset{\text{def}}{=} \mathcal{P}(\mathcal{P}(X))$ and $\Cup$ denotes *pairwise union*:

$$A \Cup B \overset{\text{def}}{=} \{a \cup b : a \in A, b \in B\}$$

Considering again the same query $\bar{Q}$, we can interpret the source relation in Fig. 4(a) as a why-provenance relation by doubly-nesting the variables (e.g., p becomes $\{\{p\}\}$). Figure 4(e) shows the query output and the resulting why-provenance annotations. Note that these annotations can be obtained from the $\mathbb{B}[X]$-annotations by dropping exponents (and writing the result as a set of sets rather than sum of monomials). One can check that the corresponding operation $g : \mathbb{B}[X] \to \mathsf{Why}(X)$ which "drops exponents" is in fact a surjective semiring homomorphism. Note also that the annotations can be obtained from the $\mathsf{Trio}(X)$-annotations by dropping coefficients, and it is easy to verify that the corresponding operation $h : \mathsf{Trio}(X) \to \mathsf{Why}(X)$ which does this is also a surjective semiring homomorphism.

An interesting variation on the why-provenance semiring is obtained by requiring that the witness basis for an output tuple be *minimal*. Here the domain is $\mathsf{irr}(\mathcal{P}(X))$ the set of *irredundant* subsets of $\mathcal{P}(X)$, i.e., W is in $\mathsf{irr}(\mathcal{P}(X))$ if for any A, B in W neither is a subset of the other. We can associate with any $W \subseteq \mathcal{P}(X)$ a unique irredundant subset $\mathsf{irr}(W)$ by repeatedly looking for elements A, B such that $A \subseteq B$ and deleting B from W. Then we define a semiring $(\mathsf{irr}(\mathcal{P}(X)), +, \cdot, 0, 1)$ as follows:

$$\begin{aligned} I + J &\overset{\text{def}}{=} \mathsf{irr}(I \cup J) & \qquad I \cdot J &\overset{\text{def}}{=} \mathsf{irr}(I \Cup J) \\ 0 &\overset{\text{def}}{=} \emptyset & 1 &\overset{\text{def}}{=} \{\emptyset\} \end{aligned}$$

This is the semiring in which we compute the *minimal witness basis* [4]. It is a well-known semiring: the construction above is the construction for the free distributive lattice generated by the set X. Moreover, it is isomorphic to the semiring of *positive Boolean expressions* $(\mathsf{PosBool}(X), \vee, \wedge, \mathsf{false}, \mathsf{true})$ used in *incomplete databases* [23].[4] The domain of this semiring is the set of all Boolean expressions over variables X which are *positive*, i.e., they involve only disjunction, conjunction, and constants for true and false.[5]

Containment of UCQs for $\mathsf{PosBool}(X)$ is known to coincide with containment under the usual set semantics:[6]

[4]This characterization of minimal witness basis and its relationship to $\mathsf{PosBool}(X)$ are due to Val Tannen.

[5]Also, we identify those expressions that are equivalent modulo the axioms of Boolean algebra.

[6]This result was claimed in [20], but Gösta Grahne recently pointed out to the author that [16] had already proved this in a more general form, for queries on relations annotated with elements of a *distributive*

Theorem 4.7 ([16]) *If K is a distributive lattice then for any UCQs $\bar{P}$, $\bar{Q}$*

$$\bar{P} \sqsubseteq_K \bar{Q} \quad \textit{iff} \quad \bar{P} \sqsubseteq_{\mathbb{B}} \bar{Q}$$

$\mathsf{PosBool}(X)$ is a distributive lattice, so Theorem 4.7 justifies the "$\Leftrightarrow$" between $\mathbb{B}$ and $\mathsf{PosBool}(X)$ in the diagrams in Fig. 2. Other interesting examples of annotations from distributive lattices include the semiring of full Boolean expressions (including negation), the *fuzzy semiring* [20], and finite total orders such as the semiring of *security clearances* proposed in [14].

Taking again the same query $\bar{Q}$ and applying it to the source table in Fig. 4(a) viewed as a $\mathsf{PosBool}(X)$-relation, we obtain the $\mathsf{PosBool}(X)$-relation shown in Fig. 4(f).

The last and simplest form of provenance information we consider is the data warehousing *lineage* of [13]. In this scheme, a tuple t in a query output is annotated with the *set* of all contributing source tuples (its *lineage*). This can be captured using the following semiring [3]:

Definition 4.8 (Lineage Semiring) The *lineage semiring* for X is $(\mathcal{P}(X) \cup \{\bot\}, +, \cdot, \bot, \emptyset)$ where X is a set of variables, $\bot + S = S + \bot = S$, $\bot \cdot S = S \cdot \bot = \bot$, and $S + T = S \cdot T = S \cup T$ if $S, T \neq \bot$.

We can interpret the source relation in Fig. 4(a) as a lineage annotated relation by nesting the annotations, e.g., p becomes $\{p\}$. Applying the same query $\bar{Q}$ as before to this relation, we obtain the lineage annotated relation shown in Fig. 4(g). Note that the lineage for an output tuple can be obtained from the why-provenance of the tuple by *flattening* the set of sets, i.e., applying the function $h : \mathsf{Why}(X) \to \mathsf{Lin}(X)$ defined by $h(I) = \bigcup_{S \in I} S$. Once again, we can show that h is a surjective semiring homomorphism.

5 Containment Mappings

In characterizing K-containment of CQs we will use variations on the notion of *containment mappings*. Let P, Q be conjunctive queries, and let h be a mapping $h : \mathsf{vars}(Q) \to \mathsf{vars}(P) \cup \mathsf{consts}(P)$ extended to be the identity on constants (we will typically use the shorthand $h : Q \to P$). We define h to operate component-wise on tuples, atoms, and CQs by replacing each occurrence of a variable x with $h(x)$. We say that $h : Q \to P$ is a *containment mapping* if $h(\mathsf{head}(Q)) = \mathsf{head}(P)$ and for every atom $R_i(\bar{u})$ in the body of Q the atom $R_i(h(\bar{u}))$ occurs in the body of P.

We will also make use of the notion of the *canonical database* (or *tableau*) for a query. This is the instance $\mathsf{can}(Q)$ obtained by viewing the body of a CQ Q as a database. In doing this we blur the distinction between variables and domain values. To make the notion precise, we work with an extended domain $\mathbb{D} \cup \mathcal{X}$ where $\mathcal{X}$

bilattice. Related results have also been established in the contexts of *parametric databases* [27] and *deterministic XML* [4].

contains new constants to use for "freezing" variables into domain values. We also assume the existence of a function $c_{[\cdot]}$ that maps a variable x to a unique constant c_x from $\mathcal{X}$. We extend $c_{[\cdot]}$ to be the identity on ordinary constants from $\mathbb{D}$. Then we define $\mathsf{can}(Q)$ to be the database instance containing a tuple $R_i(c_{u_1}, \ldots, c_{u_k})$ for each atom $R_i(u_1, \ldots, u_k)$ in the body of Q. Note that when a query has duplicate atoms in the body, this does not result in duplicate tuples in the canonical database.

Example 5.1 To illustrate, consider the CQ Q defined

$$Q(x, y) \text{ :- } R(x, a), S(a, y, u), S(y, u, u)$$

The canonical database for Q is shown below:

$$R^{\mathsf{can}(Q)} = \boxed{c_x \ a} \qquad S^{\mathsf{can}(Q)} = \begin{array}{|ccc|}\hline a & c_y & c_u \\ c_y & c_u & c_u \\ \hline\end{array}$$

The classical result of [6] relates containment mappings, canonical databases, and containment of CQs under set semantics:

Theorem 5.2 ([6]) *For CQs P, Q the following are equivalent*:

1. $P \sqsubseteq_{\mathbb{B}} Q$.
2. $[\![P]\!]^{\mathsf{can}(P)} \leq_{\mathbb{B}} [\![Q]\!]^{\mathsf{can}(P)}$.
3. *There is a containment mapping $h : Q \to P$.*

We will also exploit the device of canonical databases, but for the provenance models we will use various *abstractly-tagged* versions. The *abstractly-tagged version* $\mathsf{ab}_K(R)$ of a K-relation R is obtained by annotating each tuple in the support of R with its own tuple id from X. For $\mathbb{N}[X]$, $\mathbb{B}[X]$, and $\mathsf{Trio}(X)$ this is simply a fresh variable x from X. For lineage the variable is nested in a singleton set, $\{x\}$, and for why-provenance the variable is doubly-nested, $\{\{x\}\}$. We will use the shorthand $\mathsf{can}_K(Q)$ to mean $\mathsf{ab}_K(\mathsf{can}(Q))$. Abstractly-tagged instances will also play a role outside of the context of canonical databases (cf. Lemmas 7.15 and 7.25).

Example 5.3 Consider again the CQ Q from Example 5.1. The abstractly-tagged canonical databases for Q for $K \in \{\mathsf{Lin}(X), \mathsf{Why}(X), \mathbb{N}[X]\}$ are shown below:

$\mathsf{can}_{\mathsf{Lin}(X)}(Q)$: $\qquad$ $\mathsf{can}_{\mathsf{Why}(X)}(Q)$:

$$R^{\mathsf{can}_{\mathsf{Lin}(X)}(Q)} = \begin{array}{|cc|c|}\hline c_x & a & \{p\} \\ \hline\end{array} \qquad R^{\mathsf{can}_{\mathsf{Why}(X)}(Q)} = \begin{array}{|cc|c|}\hline c_x & a & \{\{p\}\} \\ \hline\end{array}$$

$$S^{\mathsf{can}_{\mathsf{Lin}(X)}(Q)} = \begin{array}{|ccc|c|}\hline a & c_y & c_u & \{q\} \\ c_y & c_u & c_u & \{r\} \\ \hline\end{array} \qquad S^{\mathsf{ab}_{\mathsf{Why}(X)}(Q)} = \begin{array}{|ccc|c|}\hline a & c_y & c_u & \{\{q\}\} \\ c_y & c_u & c_u & \{\{r\}\} \\ \hline\end{array}$$

$\mathsf{can}_{\mathbb{N}[X]}(Q)$:

$$R^{\mathsf{can}_{\mathbb{N}[X]}(Q)} = \begin{array}{|cc|c|}\hline c_x & a & p \\ \hline\end{array}$$

$$S^{\mathsf{can}_{\mathbb{N}[X]}(Q)} = \begin{array}{|ccc|c|}\hline a & c_y & c_u & q \\ c_y & c_u & c_u & r \\ \hline\end{array}$$

6 Bounds from Semiring Homomorphisms

In this section we establish some initial bounds on the "relative behavior" of the various provenance models w.r.t. query containment and equivalence, based on surjective semiring homomorphisms.

A function $h : K \to K'$ can be made to transform a K-relation R into a K'-relation $h(R)$ by applying h to each tuple annotation in R. Performing this transformation component-wise on the K-relations of a K-instance $\mathfrak{A}$ transforms it into a K'-instance $h(\mathfrak{A})$. It was shown in [20] that semiring homomorphisms work nicely with UCQs on K-relations:

Proposition 6.1 ([20]) *Let $h : K \to K'$ and assume that K, K' are commutative semirings. Then $[\![\bar{Q}]\!]^{h(\mathfrak{A})} = h([\![\bar{Q}]\!]^{\mathfrak{A}})$ for all $\bar{Q} \in UCQ$ and K-instances $\mathfrak{A}$ iff h is a semiring homomorphism.*

The observations we have made in Sect. 4 about the existence of surjective semiring homomorphisms relating the various provenance models turn out to yield some easy bounds on their "relative behavior" with respect to query containment (and therefore also equivalence). We write $K_1 \Rightarrow K_2$ to mean that for all UCQs $\bar{Q}_1, \bar{Q}_2$, if $\bar{Q}_1 \sqsubseteq_{K_1} \bar{Q}_2$ then $\bar{Q}_1 \sqsubseteq_{K_2} \bar{Q}_2$. Then we have the following:

Lemma 6.2 *For naturally-ordered semirings K_1, K_2, if there exists a surjective homomorphism $h : K_1 \to K_2$, then $K_1 \Rightarrow K_2$.*

Proof Suppose that $h : K_1 \to K_2$ is a surjective semiring homomorphism and that $\bar{Q}_1 \sqsubseteq_{K_1} \bar{Q}_2$. Consider an arbitrary K_2-instance $\mathfrak{B}$. We want to show that $[\![\bar{Q}_1]\!]^{\mathfrak{B}} \le_{K_2} [\![\bar{Q}_2]\!]^{\mathfrak{B}}$. Since h is surjective, there exists a K_1-instance $\mathfrak{A}$ such that $\mathfrak{B} = h(\mathfrak{A})$. Since $\bar{Q}_1 \sqsubseteq_{K_1} \bar{Q}_2$ we have that $[\![\bar{Q}_1]\!]^{\mathfrak{A}} \le_{K_1} [\![\bar{Q}_2]\!]^{\mathfrak{A}}$. By Proposition 3.3, this implies $h([\![\bar{Q}_1]\!]^{\mathfrak{A}}) \le_{K_2} h([\![\bar{Q}_2]\!]^{\mathfrak{A}})$. But by Proposition 6.1, $h([\![\bar{Q}_1]\!]^{\mathfrak{A}}) = [\![\bar{Q}_1]\!]^{h(\mathfrak{A})} = [\![\bar{Q}_1]\!]^{\mathfrak{B}}$, and likewise, $h([\![\bar{Q}_2]\!]^{\mathfrak{A}}) = [\![\bar{Q}_2]\!]^{h(\mathfrak{A})} = [\![\bar{Q}_2]\!]^{\mathfrak{B}}$. It follows that $[\![\bar{Q}_1]\!]^{\mathfrak{B}} \le_{K_2} [\![\bar{Q}_2]\!]^{\mathfrak{B}}$. Since $\mathfrak{B}$ was chosen arbitrarily, it follows that $\bar{Q}_1 \sqsubseteq_{K_2} \bar{Q}_2$, as required. □

Based on our previous observations, we can conclude the following about the "relative behavior" of the semirings for provenance w.r.t. containment (and therefore also equivalence) of UCQs:

Theorem 6.3 *If there is a path downward from K_1 to K_2 in Fig. 3, then $K_1 \Rightarrow K_2$.*

We shall see in Sect. 7 which of the implications are strict (as indicated by the ticked arrows "$\not\Rightarrow$" in Fig. 2).

Finally, we note that using similar reasoning, it is possible to establish bounds for containment/equivalence of UCQs for *arbitrary* semirings. To state the result, we first recall the standard definition of a *positive* semiring. Given a semiring K define $\dagger : K \to \mathbb{B}$ as follows:

$$\dagger(0) \overset{\text{def}}{=} \mathsf{false}$$

$$\dagger(a) \overset{\text{def}}{=} \mathsf{true} \text{ when } a \neq 0$$

Proposition 6.4 *The following are equivalent*:

1. $\dagger$ *is a semiring homomorphism*
2. K *satisfies*
   (a) $0 \neq 1$
   (b) $a + b = 0$ *implies* $a = 0$ *or* $b = 0$
   (c) $ab = 0$ *implies* $a = 0$ *or* $b = 0$

A semiring K is called *positive* if it satisfies either of the (equivalent) statements in Proposition 6.4. This is a large class of semirings: $\mathbb{B}$, $\mathbb{N}$, $\mathsf{PosBool}(X)$, and all of the semirings for provenance we have considered in this paper are positive.

Now we are ready to state the theorem.

Theorem 6.5 *For all* K, $\mathbb{N}[X] \Rightarrow K$. *For all positive* K, $K \Rightarrow \mathbb{B}$.

Proof $\mathbb{N}[X] \Rightarrow K$ follows from similar reasoning as in Proposition 6.2, but using the universality of the provenance polynomials rather than the existence of surjective semiring homomorphisms to establish the relationship. $K \Rightarrow \mathbb{B}$ follows immediately from Proposition 6.2 using the definition of positive semiring. □

For the special case of CQs containing no self-joins, we shall see in Sect. 7.4 (cf. Corollary 7.13) that the bounds of Theorem 6.5 collapse to a uniform condition for equivalence.

7 Main Results

We are now ready to present our main results on containment and equivalence.

For $\mathsf{Lin}(X)$, $\mathsf{Why}(X)$, and $\mathbb{B}[X]$, the decision procedures for containment of CQs (and the accompanying complexity results) extend easily to UCQs because of the following general fact which was first noted for the case of set semantics in [30]:

Proposition 7.1 *If a semiring* K *is* idempotent, *then for all UCQs* $\bar{P}$, $\bar{Q}$, *we have* $\bar{P} \sqsubseteq_K \bar{Q}$ *iff for every CQ* P *in* $\bar{P}$ *there is a CQ* Q *in* $\bar{Q}$ *such that* $P \sqsubseteq_K Q$. *As a consequence, for such a given idempotent* K, *we have* K*-containment of CQs is in* C *iff* K*-containment of UCQs is in* C, *for* $C \in \{\text{PTIME}, \text{NP}\}$.

(A semiring K is called *idempotent* if addition in K is idempotent, i.e., $a+a=a$ for all $a \in K$.) $\mathsf{Lin}(X)$, $\mathsf{Why}(X)$, and $\mathbb{B}[X]$ are all idempotent semirings. $\mathbb{N}[X]$ and $\mathsf{Trio}(X)$ are not idempotent, nor is the semiring of natural numbers used for bag semantics (and the failure of Proposition 7.1 for bag semantics was noted in [8]).

We also note that for idempotent semirings, containment and equivalence of UCQs are easily inter-reducible (and polynomially equivalent). This again generalizes a well-known fact for set semantics [30]:

Proposition 7.2 *For UCQs $\bar{Q}_1$, $\bar{Q}_2$ and idempotent K we have*

1. $\bar{Q}_1 \sqsubseteq_K \bar{Q}_2$ *iff* $\bar{Q}_1 \cup \bar{Q}_2 \equiv_K \bar{Q}_2$
2. $\bar{Q}_1 \equiv_K \bar{Q}_2$ *iff* $\bar{Q}_1 \sqsubseteq_K \bar{Q}_2$ *and* $\bar{Q}_2 \sqsubseteq_K \bar{Q}_1$

(The second item is just the definition of K-equivalence of UCQs.)

7.1 Lineage

Theorem 7.3 *For CQs P, Q the following are equivalent*:

1. $P \sqsubseteq_{\mathsf{Lin}(X)} Q$.
2. $[\![P]\!]^{\mathsf{can}_{\mathsf{Lin}(X)}(P)} \leq_{\mathsf{Lin}(X)} [\![Q]\!]^{\mathsf{can}_{\mathsf{Lin}(X)}(P)}$.
3. *For every atom $R(\bar{u}) \in \mathsf{body}(P)$ there is a containment mapping $h : Q \to P$ with $R(\bar{u})$ in the image of h.*

Proof (1) $\Rightarrow$ (2) is trivial. For (2) $\Rightarrow$ (3), let $t = \mathsf{head}(P)$ and let $\mathfrak{A} = \mathsf{can}_{\mathsf{Lin}(X)}(P)$. Observe that $[\![P]\!]^{\mathfrak{A}}(t) = \{x_1, \dots, x_n\}$ where $\{x_1\}, \dots, \{x_n\}$ are all the annotations occurring in $\mathfrak{A}$. Since $[\![P]\!]^{\mathfrak{A}} \leq_{\mathsf{Lin}(X)} [\![Q]\!]^{\mathfrak{A}}$, and $\{x_1, \dots, x_n\}$ is the top element in the subsemiring of $\mathsf{Lin}(X)$ generated by $\{x_1\}, \dots, \{x_n\}$, it follows that $[\![Q]\!]^{\mathfrak{A}}(t) = \{x_1, \dots, x_n\}$ as well. Now, for each such x_i, there must be a derivation $\nu : Q \to \mathfrak{A}$ of t mapping some atom $R(v_1, \dots, v_k)$ in the body of Q to a tuple $R(c_{u_1}, \dots, c_{u_k})$ such that $R^{\mathfrak{A}}(c_{u_1}, \dots, c_{u_k}) = \{x_i\}$ (and $R(u_1, \dots, u_k)$ is an atom in the body of P). But this valuation ν may also be viewed as a containment mapping from Q to P with $R(u_1, \dots, u_k)$ in the image. More precisely, from ν we define a mapping $h : \mathsf{vars}(Q) \cup \mathsf{consts}(Q) \to \mathsf{vars}(P) \cup \mathsf{consts}(Q)$ as follows:

- $h(c) \overset{\text{def}}{=} c$ for $c \in \mathsf{consts}(Q)$
- $h(x) \overset{\text{def}}{=} y$ for $x \in \mathsf{vars}(Q)$ such that $\nu(x) = c_y$ for some $c_y \in \mathcal{X}$
- $h(x) \overset{\text{def}}{=} \nu(x)$ otherwise

It is clear that h is a containment mapping from Q to P, with $R(u_1, \dots, u_k)$ in the image.

For (3) $\Rightarrow$ (1), consider an arbitrary $\mathsf{Lin}(X)$-instance $\mathfrak{A}$ and output tuple t. We want to show that for all $x \in X$, if $x \in [\![P]\!]^{\mathfrak{A}}(t)$, then $x \in [\![Q]\!]^{\mathfrak{A}}(t)$. If $x \in [\![P]\!]^{\mathfrak{A}}(t)$, then there exists a derivation $\nu : \mathsf{vars}(P) \to \mathbb{D}$ of t in $[\![P]\!]^{\mathfrak{A}}$ such that $x \in R^{\mathfrak{A}}(\nu(\bar{u}))$ for some atom $R(\bar{u})$ in the body of P. By assumption, there exists a containment mapping $h : Q \to P$ such that $R(\bar{u}) = R(h(\bar{v}))$ for some atom $R(\bar{v})$ in the body of Q. But then $\nu \circ h$ is a derivation of t in $[\![Q]\!]^{\mathfrak{A}}$, and moreover, $x \in R((\nu \circ h)(\bar{v}))$. It follows that $x \in [\![Q]\!]^{\mathfrak{A}}(t)$, as required. $\square$

It is easy to find examples of CQs P, Q such that there is a containment mapping $h : Q \to P$, but condition (3) above is not satisfied, e.g.:

$$P(x, y) \text{ :- } R(x, y), R(x, z) \qquad Q(x, y) \text{ :- } R(x, y)$$

There is no containment mapping $h : Q \to P$ with $R(x, z)$ in the image of h, so $P \not\sqsubseteq_{\mathsf{Lin}(X)} Q$. However, one can find containment mappings $h' : P \to Q$ and $h'' : Q \to P$ in both directions, so by Theorem 5.2, $P \equiv_{\mathbb{B}} Q$. This justifies the "$\not\Rightarrow$" between lineage and $\mathsf{PosBool}(X)/\mathbb{B}$ in Figs. 2(a)–(d).

Note that the above example seems to contradict[7] Theorem 4.8 of [13] which claims that $P \equiv_{\mathsf{Lin}(X)} Q$ iff $P \equiv_{\mathbb{B}} Q$. In fact, the contradiction is explained by the fact that the definition of lineage given in that paper only makes sense for CQs without self-joins. We have already seen (Corollary 7.13) that for this class of queries, K-equivalence is the same as isomorphism, for any positive K (including the lineage semiring).

Also, condition (3) of Theorem 7.3 was identified previously in [8] as a necessary (but not sufficient) condition for bag containment of CQs. This justifies the "$\not\Rightarrow$" between $\mathbb{N}$ and lineage in Fig. 2(a).

While the conditions for checking lineage containment and set containment of CQs or UCQs are different, the complexity turns out to be the same:

Corollary 7.4 *The problems of checking* $\mathsf{Lin}(X)$*-containment of CQs* (*or UCQs*) *and of checking* $\mathsf{Lin}(X)$*-equivalence of CQs* (*or UCQs*) *are all* NP*-complete.*

Proof For CQs, membership in NP follows from part (3) of Theorem 7.3. To establish NP-hardness, we use a variation on the reduction from graph 3-coloring showing NP-hardness of ordinary set-containment of CQs [6]. As there, we use the fact that a directed graph $G = (V, E)$ is 3-colorable iff there exists a graph homomorphism $h : G \to C$, where C is the complete directed graph (without self-loops) on 3 vertices. Next, given a directed graph $G = (V, E)$, we encode G as the Boolean CQ Q_G whose body contains an atom $E(u, v)$ for each edge in the graph, and we similarly encode C as a Boolean CQ Q_C. To complete the reduction showing NP-hardness of ordinary set containment, it suffices to observe that there exists a graph homomorphism $h : G \to C$ iff there exists a containment mapping $h' : Q_G \to Q_C$. With $\mathsf{Lin}(X)$-containment, there is one more step: we assume w.l.o.g. that $\mathsf{vars}(Q_G)$ and $\mathsf{vars}(Q_C)$ are disjoint, and we let Q'_G be the Boolean CQ whose body is the conjunction of the bodies of Q_G and Q_C. It is clear that there exists a containment mapping $h : Q_G \to Q_C$ iff there exists a containment mapping $h' : Q'_G \to Q_C$. Moreover, any containment mapping $h' : Q'_G \to Q_C$ has in its image every atom in the body of Q_C, thus satisfying condition (3) of Theorem 7.3. It follows that checking $\mathsf{Lin}(X)$-containment of CQs is NP-hard.

For UCQs, NP-hardness follows immediately from NP-hardness for CQs, and membership in NP follows using Proposition 7.1. □

[7] The example and this observation are due to James Cheney and Wang-Chiew Tan.

7.2 Why-Provenance

To characterize $\mathsf{Why}(X)$-containment of CQs, we define the concept of *onto containment mappings*. A mapping $h : Q \to P$ is an *onto containment mapping* if it is a containment mapping and $\mathsf{body}(P) \leq_{\mathbb{N}} h(\mathsf{body}(Q))$.

Theorem 7.5 *For CQs P, Q the following are equivalent*:

1. $P \sqsubseteq_{\mathsf{Why}(X)} Q$.
2. $[\![P]\!]^{\mathsf{can}_{\mathsf{Why}(X)}(P)} \leq_{\mathsf{Why}(X)} [\![Q]\!]^{\mathsf{can}_{\mathsf{Why}(X)}(P)}$.
3. *There is an onto containment mapping $h : Q \to P$.*

Proof (1) $\Rightarrow$ (2) is trivial. The remaining two cases are similar to the proof of Theorem 7.3.

For (2) $\Rightarrow$ (3), let $t = \mathsf{head}(P)$ and let $\mathfrak{A} = \mathsf{can}_{\mathsf{Why}(X)}(P)$. Observe that $\{x_1, \ldots, x_n\} \in [\![P]\!]^{\mathfrak{A}}(t)$, where $\{\{x_1\}\}, \ldots, \{\{x_n\}\}$ are all the annotations of tuples in $\mathfrak{A}$. Since $[\![P]\!]^{\mathfrak{A}} \leq_{\mathsf{Why}(X)} [\![Q]\!]^{\mathfrak{A}}$, it follows that $\{x_1, \ldots, x_n\} \in [\![Q]\!]^{\mathfrak{A}}(t)$ as well. Now, to produce $\{x_1, \ldots, x_n\}$ for t in the query result, there must be a justifying valuation $\nu : Q \to \mathfrak{A}$ mapping each atom $R(\bar{u})$ in the body of Q to a tuple $R(\bar{x})$ in $\mathfrak{A}$, such that the valuation is surjective on $\mathfrak{A}$. This valuation may also be viewed as an onto containment mapping from Q to P.

For (3) $\Rightarrow$ (1), consider an arbitrary $\mathsf{Why}(X)$-instance $\mathfrak{A}$ and output tuple t. Let $R_1(\bar{u}_1), \ldots, R_n(\bar{u}_n)$ be the body of P, and let $S_1(\bar{v}_1), \ldots, S_m(\bar{v}_m)$ be the body of Q. We want to show that for all $X' \subseteq X$, if $X' \in [\![P]\!]^{\mathfrak{A}}(t)$, then $X' \in [\![P]\!]^{\mathfrak{A}}(t)$. Now, if $X' \in [\![P]\!]^{\mathfrak{A}}(t)$, then there exists a justifying valuation $\nu : \mathsf{vars}(P) \to \mathbb{D}$ for t in $[\![P]\!]^{\mathfrak{A}}$ such that $X' = \Cup_{i=1}^{n} R_i{}^{\mathfrak{A}}(\nu(\bar{u}_i))$. By assumption, there exists an onto containment mapping $h : Q \to P$. But $\nu \circ h : \mathsf{vars}(Q) \to \mathbb{D}$ is a valuation justifying $X' \in [\![Q]\!]^{\mathfrak{A}}(t)$. □

The existence of an onto containment mapping is a strictly stronger requirement than condition (3) of Theorem 7.3. For example, consider the queries

$$P(x) \text{ :- } R(x, y), R(x, x) \qquad Q(u) \text{ :- } R(u, v)$$

There is no onto containment mapping from Q to P, hence $P \not\sqsubseteq_{\mathsf{Why}(X)} Q$, but one can find containment mappings satisfying condition (3) of Theorem 7.3 in both directions: g that sends x to u and y to v is a containment mapping from P to Q, while h that sends u to x and v to y is a containment mapping from Q to P. Thus $P \equiv_{\mathsf{Lin}(X)} Q$. This justifies the "$\not\Rightarrow$" between why-prov. and lineage in Fig. 2(a)–(d).

We note that the existence of onto containment mappings was identified in [8] as a sufficient (but not necessary) condition for bag containment of CQs. This justifies the "$\not\Rightarrow$" between $\mathsf{Why}(X)$ and $\mathbb{N}$ in Fig. 2(a).

The existence of onto containment mappings in both directions leads to a simple characterization of $\mathsf{Why}(X)$-equivalence of CQs:

Theorem 7.6 *For CQs P, Q, $P \equiv_{\mathsf{Why}(X)} Q$ iff $P \cong Q$.*

Proof Clearly isomorphism implies K-equivalence for any K, in particular for why provenance. In the other direction, if $P \equiv_{\mathsf{Why}(X)} Q$ by Theorem 7.5 there must exist onto containment mappings $h : Q \to P$ and $g : P \to Q$. But since both mappings are surjective they must also be injective. It follows that $P \cong Q$. □

It was shown in [8] that bag equivalence of CQs is also the same as isomorphism, hence the "$\Leftrightarrow$" between $\mathbb{N}$ and $\mathsf{Why}(X)$ in Fig. 2(b). Also, note that there are $\mathsf{Lin}(X)$-equivalent CQs which are not isomorphic, for example:

$$P(x) \ \text{:-}\ R(x, y) \qquad Q(x) \ \text{:-}\ R(x, y), R(x, z)$$

Thus we have the "$\not\Rightarrow$" between $\mathsf{Why}(X)$ and $\mathsf{Lin}(X)$ in Fig. 2(b).

For UCQs $\bar{P}$ and $\bar{Q}$, we note that Theorem 7.6 does not imply that for UCQs $\bar{P} \equiv_{\mathsf{Why}(X)} \bar{Q}$ iff $\bar{P} \cong \bar{Q}$ (and indeed this is not the case). As an example, suppose P, Q are non-isomorphic CQs such that $P \sqsubseteq_{\mathsf{Why}(X)} Q$. Then $(P, Q) \equiv_{\mathsf{Why}(X)} (Q)$ but $(P, Q) \ncong (Q)$.

Corollary 7.7 *Checking* $\mathsf{Why}(X)$*-containment for CQs or UCQs and* $\mathsf{Why}(X)$*-equivalence for UCQs is* NP*-complete. Checking* $\mathsf{Why}(X)$*-equivalence for CQs is* GI*-complete.*

Proof For $\mathsf{Why}(X)$-containment of CQs, membership in NP follows from part (3) of Theorem 7.5, and this along with Proposition 7.1 implies the same for UCQs.

To show NP-hardness of $\mathsf{Why}(X)$-containment of CQs (and therefore also UCQs), we use a straightforward variation on the reduction from graph 3-coloring from Corollary 7.4. Given a directed graph $G = (V, E)$, we again construct a Boolean CQ Q_G encoding G as in Corollary 7.4, and Boolean CQ Q_C encoding the complete graph on 3 vertices. This time, however, we now also construct a Boolean CQ Q'_G whose body is the disjoint union of Q_G and Q_C; we include the copy of Q_C to ensure that any containment mapping from Q_G to Q_C can be extended to an onto containment mapping from Q'_G to Q_C. We observe that there exists a 3-coloring of G iff there exists an onto containment mapping from Q'_G to Q_C to complete the proof of NP-hardness.

GI-completeness of $\mathsf{Why}(X)$-equivalence follows immediately from Theorem 7.6. □

7.3 $\mathbb{B}[X]$-Provenance

To characterize $\mathbb{B}[X]$-containment of CQs we will need another variation on containment mappings, which we call *exact containment mappings*. A containment mapping $h : Q \to P$ is an *exact containment mapping* if it induces a bijection of atoms in the bodies of P and Q. Note that there is an exact containment mapping from Q to P iff P can be obtained from Q (up to isomorphism) by *unifying* variables in Q.

Theorem 7.8 *For CQs* P, Q, *the following are equivalent*:

1. $P \sqsubseteq_{\mathbb{B}[X]} Q$.

2. $[\![P]\!]^{\mathsf{can}_{\mathbb{B}[X]}(P)} \leq_{\mathbb{B}[X]} [\![Q]\!]^{\mathsf{can}_{\mathbb{B}[X]}(P)}$.
3. *There is an exact containment mapping* $h : Q \to P$.

Proof (1) $\Rightarrow$ (2) is trivial. (3) $\Rightarrow$ (1) is straightforward to check. For (2) $\Rightarrow$ (3), we assume for simplicity that $\mathsf{body}(P)$ contains no duplicate atoms, but the argument generalizes easily. Suppose (2) holds. Then in particular,

$$[\![P]\!]^{\mathsf{can}_{\mathbb{B}[X]}(P)}(t) \leq [\![Q]\!]^{\mathsf{can}_{\mathbb{B}[X]}(P)}(t),$$

where t is the tuple of distinguished variables in $\mathsf{head}(P)$. Also, the polynomial $[\![P]\!]^{\mathsf{can}_{\mathbb{B}[X]}(P)}(t)$ contains as a term (i.e., with Boolean coefficient true) the product $x_1 \cdots x_n$ of all tags occurring in $\mathsf{can}_{\mathbb{B}[X]}(P)$, since this term is derived via the "identity" valuation ν that maps constants to themselves and variables $x \in \mathsf{vars}(P)$ to c_x. Since containment holds, the polynomial $[\![Q]\!]^{\mathsf{can}_{\mathbb{B}[X]}(P)}(t)$ must also contain the same term. Working backwards, there must be some derivation $\nu : \mathsf{vars}(Q) \to \mathbb{D} \cup \mathcal{X}$ of the term. Moreover, in order to yield all variables $x_1, \ldots, x_n$ in the term, ν must map the atoms of $\mathsf{body}(Q)$ surjectively onto the tuples of $\mathsf{can}_{\mathbb{B}[X]}(P)$; and in order for all the exponents in the term to all equal one, the mapping of atoms to tuples must be injective. An exact containment mapping h from Q to P can be constructed from ν via the same procedure given in the proof of Theorem 7.3. □

Every exact containment mapping is also an onto containment mapping, but the converse is not true. For example, the mapping $h : Q \to P$ which sends w to u, z to v, and everything else to itself in

$$P(x, y) \text{ :- } R(x, y), S(u, v) \qquad Q(x, y) \text{ :- } R(x, y), S(u, v), S(w, z)$$

is an onto containment mapping, but not an exact containment mapping. This justifies the "$\not\Rightarrow$" between $\mathbb{B}[X]$ and $\mathsf{Why}(X)$ in Fig. 2(a),(c). To justify the "$\not\Rightarrow$" between $\mathbb{B}[X]$ and $\mathsf{Why}(X)$ in Fig. 2(d), consider P, Q as above and define the UCQs $\bar{P} = (P)$ and $\bar{Q} = (P, Q)$. Then $\bar{P} \equiv_{\mathsf{Why}(X)} \bar{Q}$ but $\bar{P} \not\equiv_{\mathbb{B}[X]} \bar{Q}$.

Like $\mathsf{Why}(X)$-equivalence, $\mathbb{B}[X]$-equivalence of CQs turns out to be the same as isomorphism:

Theorem 7.9 *For CQs* P, Q, $P \equiv_{\mathbb{B}[X]} Q$ *iff* $P \cong Q$.

This justifies the "$\Leftrightarrow$" between $\mathsf{Why}(X)$ and $\mathbb{B}[X]$ in Fig. 2(b).

Checking for the existence of an exact containment mapping turns out to have the same complexity as checking for the existence of a containment mapping:

Corollary 7.10 *Checking* $\mathbb{B}[X]$*-containment of CQs or UCQs, or* $\mathbb{B}[X]$*-equivalence of UCQs, is* NP*-complete. Checking* $\mathbb{B}[X]$*-equivalence of CQs is* GI*-complete.*

Proof (Sketch) GI-completeness of checking $\mathbb{B}[X]$-equivalence of CQs follows from Theorem 7.9 and the fact that we can view directed graphs as Boolean CQs (and vice versa), as in the proof of Corollary 7.4. The proofs of the other results are again similar to Corollary 7.4. □

7.4 Provenance Polynomials

We now prove the results for $\mathbb{N}[X]$-containment. For CQs, this turns out to be the same as for $\mathbb{B}[X]$-containment (thus justifying the "$\Leftrightarrow$" between $\mathbb{N}[X]$ and $\mathbb{B}[X]$ in Fig. 2(a)):

Theorem 7.11 *For CQs P, Q the following are equivalent*:

1. $P \sqsubseteq_{\mathbb{N}[X]} Q$.
2. $[\![P]\!]^{\mathsf{can}_{\mathbb{N}[X]}(P)} \leq_{\mathbb{N}[X]} [\![Q]\!]^{\mathsf{can}_{\mathbb{N}[X]}(P)}$.
3. *There is an exact containment mapping $h : Q \to P$.*

Proof (1) $\Rightarrow$ (2) is trivial, and (2) $\Rightarrow$ (3) is exactly the same as in Theorem 7.8. For (3) $\Rightarrow$ (1) some additional care is required because addition in $\mathbb{N}[X]$ is not idempotent. We need to make sure that the coefficient of an arbitrary term in the polynomial $[\![Q]\!]^{\mathfrak{A}}(t)$, for some arbitrary $\mathbb{N}[X]$-instance $\mathfrak{A}$ and tuple t, is at least as large as the coefficient of the same term in the polynomial $[\![P]\!]^{\mathfrak{A}}(t)$. To check this, it suffices to observe that for any valuations $\nu, \nu' : \mathsf{vars}(P) \to \mathbb{D}$ justifying a monomial term in $[\![P]\!]^{\mathfrak{A}}(t)$, the valuations $\nu \circ h$ and $\nu' \circ h$ justify the same monomial in $[\![Q]\!]^{\mathfrak{A}}(t)$; and moreover, since h is surjective on the variables of P, if $\nu \neq \nu'$ then $\nu \circ h \neq \nu' \circ h$. Hence every justification for $[\![P]\!]^{\mathfrak{A}}(t)$ corresponds to a *unique* justification for $[\![Q]\!]^{\mathfrak{A}}(t)$. Since addition in $\mathbb{N}$ is monotone this implies the required inequality for the term coefficients. $\square$

Since $\mathbb{N}[X]$-containment of CQs holds exactly when $\mathbb{B}[X]$-containment holds, the same is true for $\mathbb{N}[X]$-equivalence:

Theorem 7.12 *Let P, Q be two CQs. Then $P \equiv_{\mathbb{N}[X]} Q$ iff $P \cong Q$.*

This justifies the "$\Leftrightarrow$" between $\mathbb{N}[X]$ and $\mathbb{B}[X]$ (and therefore also $\mathsf{Why}(X)$ and $\mathbb{N}$) in Fig. 2(b).

This result actually has wider implications beyond $\mathbb{N}[X]$-equivalence: for the special case of CQs containing no self-joins, it can be used to show that the bounds of Theorem 6.5 collapse to a uniform condition for equivalence:

Corollary 7.13 (of Theorem 6.5 and Theorem 7.12) *If CQs P, Q contain no self-joins* (*i.e.*, *neither query's body contains multiple occurrences of a predicate symbol*), *then for any positive K, we have $P \equiv_K Q$ iff $P \cong Q$.*

Proof It is clear that $P \cong Q$ implies K-equivalence of P and Q for any K. In the other direction, assume P, Q contain no self-joins and suppose $P \equiv_K Q$ for some positive K. Then by Theorem 6.5, $P \equiv_{\mathbb{B}} Q$. Therefore (using Theorem 5.2) there exist containment mappings $g : P \to Q$ and $h : Q \to P$. Since neither query has a self-join, it is not hard to see that g and h must both be bijective, and hence either serves as the required isomorphism between P and Q. $\square$

Therefore, for conjunctive queries without self-joins, every "$\Rightarrow$" in Fig. 2(b) becomes a "$\Leftrightarrow$".

Next we consider $\mathbb{N}[X]$-containment of UCQs. Using similar reasoning as in Theorem 7.11, it is not hard to see that a weaker version of the Sagiv-Yannakakis property for set-containment of UCQs [30] holds for $\mathbb{N}[X]$:

Lemma 7.14 *For UCQs $\bar{P}, \bar{Q}$, if $\bar{P} \sqsubseteq_{\mathbb{N}[X]} \bar{Q}$, then for every $P_i \in \bar{P}$ there exists $Q_j \in \bar{Q}$ s.t. $P_i \sqsubseteq_{\mathbb{N}[X]} Q_j$.*

Proof Consider an arbitrary $P_i \in \bar{P}$. Since $\bar{P} \sqsubseteq_{\mathbb{N}[X]} \bar{Q}$, then in particular, we have $[\![\bar{P}]\!]^{\mathsf{can}_{\mathbb{N}[X]}(P_i)}(t) \leq_{\mathbb{N}[X]} [\![\bar{Q}]\!]^{\mathsf{can}_{\mathbb{N}[X]}(P_i)}(t)$ where t is the head of P_i. Clearly, the term $x_1 + \cdots + x_k$, where $x_1, \ldots, x_k$ are the abstract tags of $\mathsf{can}_{\mathbb{N}[X]}(P_i)$, will have non-zero coefficient in $[\![\bar{P}]\!]^{\mathsf{can}_{\mathbb{N}[X]}(P_i)}(t)$, and therefore also in $[\![\bar{Q}]\!]^{\mathsf{can}_{\mathbb{N}[X]}(P_i)}(t)$. As a result there must be some CQ $Q_j \in \bar{Q}$ such that there exists a valuation $\nu : \mathsf{vars}(Q_j) \to \mathbb{D} \cup \mathcal{X}$ justifying the term. Using the same reasoning as in Theorem 7.11, we construct from this valuation an exact containment mapping from Q_j to P_i. It follows, using Theorem 7.11, that $P_i \sqsubseteq_{\mathbb{N}[X]} Q_j$. □

A natural question to ask is whether the lemma above can be strengthened to require that each $P_i \in \bar{P}$ correspond to a *unique* $Q_j \in \bar{Q}$; as this is clearly also a sufficient condition for containment, this would therefore yield a decision procedure for containment. However, the strengthened version is not true: consider the UCQs $\bar{P} = (P_1, P_2)$ and $\bar{Q} = (Q_1)$ where

$$P_1 \text{ :- } R(x, y), R(x, x) \qquad Q_1 \text{ :- } R(x, y), R(u, u)$$
$$P_2 \text{ :- } R(x, y), R(y, y)$$

Clearly, there is no unique assignment of the CQs in $\bar{P}$ to CQs in $\bar{Q}$; both P_1 and P_2 must be assigned to Q_1. Nevertheless, we can show that $\bar{P} \sqsubseteq_{\mathbb{N}[X]} \bar{Q}$. To see this, we note that $\bar{Q}$ is $\mathbb{N}[X]$-equivalent to the UCQ $\bar{Q}'$ with inequalities

$$Q'_1 \text{ :- } R(x, y), R(x, x)$$
$$Q'_2 \text{ :- } R(x, y), R(y, y)$$
$$Q'_3 \text{ :- } R(x, y), R(u, u), \quad x \neq u,\ y \neq u$$

It is clear that $\bar{P} \equiv_{\mathbb{N}} \bar{Q}'$, since $P_1 \cong Q'_1$ and $P_2 \cong Q'_2$.

Another natural idea is to check containment of $\bar{P}$ in $\bar{Q}$ by evaluating both queries on the canonical database for $\bar{P}$, in analogy with Theorem 7.11; unfortunately, one can easily find counterexamples showing that this procedure is unsound.

However, we are able to show that $\mathbb{N}[X]$-containment of UCQs is decidable in PSPACE by establishing a "small counterexample" property. In particular we show that if $\bar{P} \not\sqsubseteq_{\mathbb{N}[X]} \bar{Q}$, then $[\![\bar{P}]\!]^{\mathfrak{A}} \not\leq_{\mathbb{N}[X]} [\![\bar{Q}]\!]^{\mathfrak{A}}$ for some $\mathbb{N}[X]$-instance $\mathfrak{A}$ whose size is bounded by the sizes of $\bar{P}$ and $\bar{Q}$.

We begin by bounding the sizes of the *annotations* of the required counterexample, by observing that if a counterexample exists, then its abstractly-tagged version is also a counterexample:

Lemma 7.15 *For any naturally-ordered semiring K, if $\bar{P}, \bar{Q} \in UCQ$ and $[\![\bar{P}]\!]^{\mathfrak{A}} \not\leq_K [\![\bar{Q}]\!]^{\mathfrak{A}}$ for some K-instance $\mathfrak{A}$, then $[\![\bar{P}]\!]^{\mathfrak{B}} \not\leq_{\mathbb{N}[X]} [\![\bar{Q}]\!]^{\mathfrak{B}}$ where $\mathfrak{B} = \mathsf{ab}_{\mathbb{N}[X]}(\mathfrak{A})$.*

Proof Suppose $[\![\bar{P}]\!]^{\mathfrak{A}} \not\leq_K [\![\bar{Q}]\!]^{\mathfrak{A}}$ for some K-instance $\mathfrak{A}$. Let K' denote the subsemiring of K generated by the annotations actually occurring in $\mathfrak{A}$. Let $\mathfrak{B} = \mathsf{ab}_{\mathbb{N}[X]}(\mathfrak{A})$, and let $\nu : X \to K'$ be the valuation which maps the variables used in $\mathfrak{B}$ to the corresponding annotations in $\mathfrak{A}$. By the universality of $\mathbb{N}[X]$, ν can be extended uniquely to a semiring homomorphism $\mathsf{Eval}_\nu : \mathbb{N}[X] \to K'$. Observe that $\mathsf{Eval}_\nu(\mathfrak{B}) = \mathfrak{A}$. It follows, using Proposition 6.1, that $\mathsf{Eval}_\nu([\![\bar{P}]\!]^{\mathfrak{B}}) = [\![\bar{P}]\!]^{\mathfrak{A}}$ and $\mathsf{Eval}_\nu([\![\bar{Q}]\!]^{\mathfrak{B}}) = [\![\bar{Q}]\!]^{\mathfrak{A}}$. Moreover, observe that Eval_ν is surjective on K'. But since $[\![\bar{P}]\!]^{\mathfrak{A}} \not\leq_K [\![\bar{Q}]\!]^{\mathfrak{A}}$ and Eval_ν is surjective, Proposition 3.3 implies that $[\![\bar{P}]\!]^{\mathfrak{B}} \not\leq_{\mathbb{N}[X]} [\![\bar{Q}]\!]^{\mathfrak{B}}$, as required. □

Of course, the lemma holds in particular for $K = \mathbb{N}[X]$. Next, we show that the *number of tuples* in a counterexample can also be bounded:

Theorem 7.16 *$\bar{P} \not\sqsubseteq_{\mathbb{N}[X]} \bar{Q}$ iff $[\![\bar{P}]\!]^{\mathfrak{A}} \not\leq_{\mathbb{N}[X]} [\![\bar{Q}]\!]^{\mathfrak{A}}$ for some abstractly-tagged $\mathbb{N}[X]$-instance $\mathfrak{A}$ containing at most k tuples, where k is the degree of $\bar{Q}$.*

Proof "$\Leftarrow$" is trivial. For "$\Rightarrow$", suppose $\bar{P} \not\sqsubseteq_{\mathbb{N}[X]} \bar{Q}$. Then for some $\mathbb{N}[X]$-instance $\mathfrak{A}$, we have $[\![\bar{P}]\!]^{\mathfrak{A}} \not\leq_{\mathbb{N}[X]} [\![\bar{Q}]\!]^{\mathfrak{A}}$. By Lemma 7.15, we may assume that $\mathfrak{A}$ is abstractly-tagged. Choose some tuple t such that $[\![\bar{P}]\!]^{\mathfrak{A}}(t) \not\leq [\![\bar{Q}]\!]^{\mathfrak{A}}(t)$. There must be some monomial μ in the polynomial $[\![\bar{P}]\!]^{\mathfrak{A}}(t)$ with coefficient a such that the same monomial μ in the polynomial $[\![\bar{Q}]\!]^{\mathfrak{A}}(t)$ has coefficient b and $a > b$. Now let $\mathfrak{A}'$ be the $\mathbb{N}[X]$-instance obtained from $\mathfrak{A}$ by discarding (by setting their annotations in $\mathfrak{A}'$ to 0) any tuples whose annotations do not occur in μ. Note that $\mathfrak{A}'$ has at most k tuples. Moreover, the coefficients for μ in the polynomials for $[\![\bar{P}]\!]^{\mathfrak{A}'}(t)$ and $[\![\bar{Q}]\!]^{\mathfrak{A}'}(t)$ are unchanged. Hence $[\![\bar{P}]\!]^{\mathfrak{A}'}(t) \not\leq [\![\bar{Q}]\!]^{\mathfrak{A}'}(t)$, and therefore, $[\![\bar{P}]\!]^{\mathfrak{A}'} \not\leq_{\mathbb{N}[X]} [\![\bar{Q}]\!]^{\mathfrak{A}'}$. This completes the proof. □

Theorem 7.16 leads immediately to a decision procedure for checking $\mathbb{N}[X]$-containment of UCQs: simply test $[\![\bar{P}]\!]^{\mathfrak{A}} \leq_{\mathbb{N}[X]} [\![\bar{Q}]\!]^{\mathfrak{A}}$ for all instances $\mathfrak{A}$ containing at most k tuples; there are still infinitely many of these, but only finitely many up to isomorphism. This can be carried out effectively by considering, for instance, instances over the first $(kn)^{kn}$ values of the domain, where n is the maximum arity of a relation in the schema. (If $\bar{P}$ and $\bar{Q}$ contain constants, these must be included among the values considered as well.) Moreover, one can check that this can be done using only polynomial space:[8]

Corollary 7.17 *$\mathbb{N}[X]$-containment of UCQs is decidable in* PSPACE.

We leave the exact complexity of the problem open.

[8]The reader may notice that for classical set semantics, the preceding argument leads immediately to a $\mathrm{coNP}^{\mathrm{NP}} = \Pi_2^p$ bound on the complexity, by guessing the counterexample instance, then evaluating both queries. However, with $\mathbb{N}[X]$ semantics, the combined complexity of query evaluation is $\#P$-hard (and in PSPACE), rather than NP-complete. Thus we only get a $\mathrm{coNP}^{\mathrm{PSPACE}} = \mathrm{PSPACE}$ bound.

Finally, what about $\mathbb{N}[X]$-equivalence of UCQs? Theorem 7.16 tells us that it is decidable, but not much else. However, it turns out we can use Theorem 7.11 along with Lemma 7.14 to show that, as with CQs, $\mathbb{N}[X]$-equivalence of UCQs is the same as isomorphism.

Theorem 7.18 *For UCQs $\bar{P}, \bar{Q}$, we have $\bar{P} \equiv_{\mathbb{N}[X]} \bar{Q}$ iff $\bar{P} \cong \bar{Q}$.*

In the proof we make use of the following simple proposition which states that removing $\mathbb{N}[X]$-equivalent CQs from $\mathbb{N}[X]$-equivalent UCQs yields $\mathbb{N}[X]$-equivalent UCQs:

Proposition 7.19 *Let $\bar{P}, \bar{Q} \in UCQ$ and suppose $\bar{P} \equiv_{\mathbb{N}[X]} \bar{Q}$. Then for all $P \in \bar{P}$, $Q \in \bar{Q}$, if $P \cong Q$, then $\bar{P}' \equiv_{\mathbb{N}[X]} \bar{Q}'$, where $\bar{P}'$ ($\bar{Q}'$) is the UCQ obtained from $\bar{P}$ ($\bar{Q}$) by removing P (Q).*

Proof (of Theorem 7.18) "$\Leftarrow$" is trivial. For "$\Rightarrow$" we argue by induction on $|\bar{P}|+|\bar{Q}|$. In the base case, $|\bar{P}|+|\bar{Q}|=0$, and the queries are trivially $\mathbb{N}[X]$-equivalent and isomorphic. In the inductive case, consider $\bar{P}=(P_1,\ldots,P_n)$ and $\bar{Q}=(Q_1,\ldots,Q_m)$ with $n+m>0$, and assume inductively that for all $\bar{P}', \bar{Q}'$ s.t. $|\bar{P}'|+|\bar{Q}'|<n+m$, if $\bar{P}' \equiv_{\mathbb{N}[X]} \bar{Q}'$ then $\bar{P}' \cong \bar{Q}'$. If $\bar{P} \equiv_{\mathbb{N}[X]} \bar{Q}$, then using Lemma 7.14, one can show that there exists a pair of non-empty sequences $i_1,\ldots,i_k$ and $j_1,\ldots,j_k$ such that $P_{i_1} \sqsubseteq_{\mathbb{N}[X]} Q_{j_1} \sqsubseteq_{\mathbb{N}[X]} \cdots \sqsubseteq_{\mathbb{N}[X]} P_{i_k} \sqsubseteq_{\mathbb{N}[X]} Q_{j_k}$ and $Q_{j_k} \sqsubseteq_{\mathbb{N}[X]} P_{i_1}$. It follows that all the CQs in the sequence are $\mathbb{N}[X]$-equivalent, and hence (by Theorem 7.16) isomorphic. In particular, we have $P_{i_1} \cong Q_{j_1}$. Denote by $\bar{P}'$ the UCQ obtained by removing P_{i_1} from $\bar{P}$, and denote by $\bar{Q}'$ the UCQ obtained by removing Q_{j_1} from $\bar{Q}$. By Proposition 7.19, we have $\bar{P}' \equiv_{\mathbb{N}[X]} \bar{Q}'$. Using the induction hypothesis, this implies $\bar{P}' \cong \bar{Q}'$. Since $\bar{P}' \cong \bar{Q}'$ and $P_{i_1} \cong Q_{j_1}$, it follows that $\bar{P} \cong \bar{Q}$ as required. □

Since $\mathbb{B}[X]$ is idempotent, but $\mathbb{N}[X]$ is not, it is easy to find examples of $\bar{P}, \bar{Q}$ where $\bar{P} \equiv_{\mathbb{B}[X]} \bar{Q}$ but $\bar{P} \not\equiv_{\mathbb{N}[X]} \bar{Q}$, e.g., $\bar{P}=(P)$ and $\bar{Q}=(P,P)$ where P is an arbitrary CQ. This justifies the "$\nRightarrow$" between $\mathbb{N}[X]$ and $\mathbb{B}[X]$ in Fig. 2(c) and Fig. 2(d).

7.5 Bag Semantics

In this section, we discuss some further connections between provenance annotations and bag semantics.

We note that by Theorem 6.5, $\mathbb{N}[X]$-containment of UCQs implies bag-containment. Since the former is decidable and the latter is not, it follows that there exist UCQs for which bag-containment holds but $\mathbb{N}[X]$-containment does not. This justifies the "$\nRightarrow$" between $\mathbb{N}[X]$ and $\mathbb{N}$ in Fig. 2(d). Also, we can show that:

Proposition 7.20 *For containment of UCQs, we have*

1. $\mathbb{N} \nRightarrow \mathbb{B}[X]$ *and* $\mathbb{B}[X] \nRightarrow \mathbb{N}$
2. $\mathbb{N} \nRightarrow \mathsf{Why}(X)$ *and* $\mathsf{Why}(X) \nRightarrow \mathbb{N}$

Proof Consider the UCQs listed below:

$$\begin{array}{lll} U_1 \text{ :- } R(x) & P \text{ :- } R(x) & Q \text{ :- } R(x), R(y) \\ U_2 \text{ :- } R(x) & & \end{array}$$

For the first claim, to show $\mathbb{N} \not\Rightarrow \mathbb{B}[X]$, we observe that $(P) \sqsubseteq_{\mathbb{N}} (Q)$ but $(P) \not\sqsubseteq_{\mathbb{B}[X]} (Q)$; and to show $\mathbb{B}[X] \not\Rightarrow \mathbb{N}$, we observe that $(U_1, U_2) \equiv_{\mathbb{B}[X]} (P)$ but $(U_1, U_2) \not\sqsubseteq_{\mathbb{N}} (P)$.

For the second claim, we observe that $(U_1, U_2) \equiv_{\mathsf{Why}(X)} (P)$ (but, again, $(U_1, U_2) \not\sqsubseteq_{\mathbb{N}} (P)$), hence $\mathsf{Why}(X) \not\Rightarrow \mathbb{N}$. At the same time, we already showed in Sect. 7.2 that for CQs, $\mathsf{Why}(X) \Rightarrow \mathbb{N}$. It follows that $\mathbb{N} \not\Rightarrow \mathsf{Why}(X)$ for UCQs. □

Next, the "$\Leftrightarrow$" between $\mathbb{N}$ and $\mathbb{N}[X]$ in Fig. 2(d) follows from the following result:

Theorem 7.21 *For UCQs $\bar{P}, \bar{Q}$ we have $\bar{P} \equiv_{\mathbb{N}} \bar{Q}$ iff $\bar{P} \equiv_{\mathbb{N}[X]} \bar{Q}$*

Proof $\mathbb{N}[X] \Rightarrow \mathbb{N}$ follows from Theorem 6.5. We prove $\mathbb{N} \Rightarrow \mathbb{N}[X]$ by contrapositive. Suppose $\bar{P} \not\equiv_{\mathbb{N}[X]} \bar{Q}$. Then for some $\mathbb{N}[X]$-instance $\mathfrak{A}$ and tuple t, we have $[\![\bar{P}]\!]^{\mathfrak{A}}(t) = A$ and $[\![\bar{Q}]\!]^{\mathfrak{A}}(t) = B$ and $A \neq B$. Since A and B are non-identical polynomials, one can always find a valuation $\nu : X \to \mathbb{N}$ such that $\mathsf{Eval}_\nu(A) \neq \mathsf{Eval}_\nu(B)$. By Proposition 6.1, we have $[\![\bar{P}]\!]^{\nu(\mathfrak{A})}(t) \neq [\![\bar{Q}]\!]^{\nu(\mathfrak{A})}(t)$. Since $\nu(\mathfrak{A})$ is an $\mathbb{N}$-instance, it follows that $\bar{P} \not\equiv_{\mathbb{N}} \bar{Q}$, as required. □

By Theorem 7.18 it follows from the above that bag equivalence of UCQs is also the same as isomorphism. This provides a new proof of a result due originally to Cohen et al. [11].[9]

7.6 Trio

For CQs, $\mathsf{Trio}(X)$-containment turns out to coincide with $\mathsf{Why}(X)$-containment:

Theorem 7.22 *For CQs P, Q we have $P \sqsubseteq_{\mathsf{Trio}(X)} Q$ iff $P \sqsubseteq_{\mathsf{Why}(X)} Q$.*

Proof We must show that Theorem 7.5 holds verbatim when $\mathsf{Why}(X)$ is replaced by $\mathsf{Trio}(X)$. (1) $\Rightarrow$ (2) continues to hold trivially.

For (2) $\Rightarrow$ (3), we replay the same argument from Theorem 7.5, *mutatis mutandis*, as follows. Let $t = \mathsf{head}(P)$ and let $\mathfrak{A} = \mathsf{can}_{\mathsf{Trio}(X)}(P)$. Observe that $x_1 + \cdots + x_n$ is a term with non-zero coefficient in $[\![P]\!]^{\mathfrak{A}}(t)$, where $x_1, \ldots, x_n$ are all the annotations of tuples in $\mathfrak{A}$. Since $[\![P]\!]^{\mathfrak{A}} \leq_{\mathsf{Trio}(X)} [\![Q]\!]^{\mathfrak{A}}$, it follows that $x_1 + \cdots + x_n$ is a term with non-zero coefficient in $[\![Q]\!]^{\mathfrak{A}}(t)$ as well. Now, to produce that term for t in the query result, there must be a justifying valuation $\nu : Q \to \mathfrak{A}$ mapping each atom $R(u_1, \ldots, u_k)$ in the body of Q to a tuple $R(c_{v_1}, \ldots, c_{v_k})$ in $\mathfrak{A}$, such that $R(v_1, \ldots, v_k)$ is an atom in

[9] The paper by Cohen et al. [11] stated the result that *bag-set* equivalence of UCQs (called *disjunctive queries* there) is the same as isomorphism, and added as an observation that this also holds for bag semantics. The outline of the proof of the bag-set semantics result is provided in [9].

the body of P, and where the valuation is surjective on $\mathfrak{A}$. This valuation may also be viewed as a containment mapping from Q to P, using the standard procedure given in the proof of Theorem 7.3. Moreover, since the valuation is surjective on $\mathfrak{A}$, the corresponding containment mapping must also be an onto containment mappings.

For (3) $\Rightarrow$ (1), the argument is much like that of Theorem 7.11. Let h be an onto containment mapping from Q to P, and consider an arbitrary $\mathsf{Trio}(X)$-instance $\mathfrak{A}$ and output tuple t. Let $R_1(\bar{u}_1), \ldots, R_n(\bar{u}_n)$ be the body of P, and let $S_1(\bar{v}_1), \ldots, S_m(\bar{v}_m)$ be the body of Q. We want to show that for every monomial term $\mu = x_1 \cdots x_k$, if M has coefficient c in $[\![P]\!]^{\mathfrak{A}}(t)$, then μ has coefficient $c' \geq c$ in $[\![P]\!]^{\mathfrak{A}}(t)$. To show this, it suffices to observe that for any valuations $\nu, \nu' : \mathsf{vars}(P) \to \mathbb{D}$ justifying μ in $[\![P]\!]^{\mathfrak{A}}(t)$, the valuations $\nu \circ h$ and $\nu' \circ h$ justify the same monomial in $[\![Q]\!]^{\mathfrak{A}}(t)$; and moreover, since h is surjective on the variables of P, if $\nu \neq \nu'$ then $\nu \circ h \neq \nu' \circ h$. Hence every justification for $[\![P]\!]^{\mathfrak{A}}(t)$ corresponds to a *unique* justification for $[\![Q]\!]^{\mathfrak{A}}(t)$. Since addition in $\mathbb{N}$ is monotone this implies the required inequality for the term coefficients. $\square$

Therefore, Theorem 7.5 (resp., Theorem 7.6) applies to $\mathsf{Trio}(X)$-containment (resp., equivalence) as well, and we have a "$\Leftrightarrow$" between $\mathsf{Trio}(X)$ and $\mathsf{Why}(X)$ in Fig. 2(a) and Fig. 2(b).

To establish the decidability of $\mathsf{Trio}(X)$-equivalence of UCQs, we note that $\mathsf{Trio}(X)$ contains an embedded copy of $\mathbb{N}$, hence $\mathsf{Trio}(X) \Rightarrow \mathbb{N}$ for UCQs. Combined with Theorem 7.21 this implies:

Theorem 7.23 *For UCQs $\bar{P}, \bar{Q}$ we have $\bar{P} \equiv_{\mathsf{Trio}(X)} \bar{Q}$ iff $\bar{P} \cong \bar{Q}$.*

This justifies the "$\Leftrightarrow$" between $\mathsf{Trio}(X)$ and $\mathbb{N}$ in Fig. 2(d).

To establish decidability in PSPACE of $\mathsf{Trio}(X)$-containment of UCQs, we follow a similar argument as used for $\mathbb{N}[X]$ in Sect. 7.4. However, this time the argument is complicated by the fact that unlike $\mathbb{N}[X]$, $\mathsf{Trio}(X)$ cannot easily "simulate its own computations" by factoring them through calculations involving abstractly-tagged databases. As a simple example, consider the CQs

$$Q_1(x, y) \text{ :- } R(x, z), R(z, y) \qquad Q_2(x, y) \text{ :- } R(x, y)$$

and consider the following $\mathsf{Trio}(X)$-relation R and its abstractly-tagged version:

$$R \stackrel{\text{def}}{=} \begin{array}{|cc|c|} \hline \mathrm{a} & \mathrm{a} & 2 \\ \mathrm{b} & \mathrm{c} & pq \\ \hline \end{array} \qquad \mathsf{ab}_{\mathsf{Trio}(X)}(R) = \begin{array}{|cc|c|} \hline \mathrm{a} & \mathrm{a} & r \\ \mathrm{b} & \mathrm{c} & s \\ \hline \end{array}$$

Note that $[\![Q_1]\!]^R(\mathrm{a}, \mathrm{a}) = 4$ while $[\![Q_2]\!]^R(\mathrm{a}, \mathrm{a}) = 2$. On the other hand, $[\![Q_1]\!]^{\mathsf{ab}_{\mathsf{Trio}(X)}(R)}(\mathrm{a}, \mathrm{a}) = [\![Q_2]\!]^{\mathsf{ab}_{\mathsf{Trio}(X)}(R)}(\mathrm{a}, \mathrm{a}) = r$. (Recall that $\mathsf{Trio}(X)$ "drops exponents" in its computations.) Hence, we do not have enough information to recover the answers over the original table.

To overcome this limitation, we introduce variation of abstractly-tagged instances called *k-abstractly tagged instances*. The tags in these instances are *sums* of k fresh variables from X. As we shall see, these turn out to suffice for $\mathsf{Trio}(X)$ to "simulate its own computations" for UCQs of degree $\leq k$.

To make this precise, we order the variables of $X = \{x_1, x_2, \dots\}$, and we define the set $S_k \subseteq \mathsf{Trio}(X)$ of sums of k fresh variables from X:

$$S_k \overset{\text{def}}{=} \{(x_1 + \cdots + x_k), (x_{k+1} + \cdots + x_{2k}), \dots\}$$

The *k-abstractly tagged version* $\mathsf{ab}_k(\mathfrak{A})$ of a $\mathsf{Trio}(X)$-instance $\mathfrak{A}$ is the $\mathsf{Trio}(X)$-instance obtained by replacing each tuple's annotation in $\mathfrak{A}$ with a fresh element of S_k. For example, considering again the $\mathsf{Trio}(X)$-relation R from earlier, we have

$$\mathsf{ab}_k(R) = \begin{array}{|cc|c|} \hline \mathrm{a} & \mathrm{a} & r+s \\ \mathrm{b} & \mathrm{c} & u+v \\ \hline \end{array}$$

Now, we define $\mathsf{Trio}_k(X) \subseteq \mathsf{Trio}(X)$ to be the set of annotations from $\mathsf{Trio}(X)$ obtained via calculations of degree $\leq k$ involving elements of S_k:

$$\mathsf{Trio}_k(X) \overset{\text{def}}{=} \{\mathsf{Eval}_\nu(a) : a \in \mathbb{N}[X], \nu : X \to S_k \text{ s.t. } a \text{ has degree} \leq k \text{ and } \nu \text{ is injective}\}$$

Proposition 7.24 *Suppose valuation $\nu : X \to S_k$ is injective. Then for any $a, b \in \mathbb{N}[X]$ of degree $\leq k$, $a = b$ iff $\mathsf{Eval}_\nu(a) = \mathsf{Eval}_\nu(b)$. As a consequence, any element $c \in \mathsf{Trio}_k(X)$ decomposes uniquely into a sum of products of elements of S_k.*

Proof Obviously, $a = b$ implies $\mathsf{Eval}_\nu(a) = \mathsf{Eval}_\nu(b)$. Now suppose $\mathsf{Eval}_\nu(a) = \mathsf{Eval}_\nu(b) = c_1\mu_1 + \cdots + c_k\mu_k$. □

To illustrate, consider the previous example, and note that $[\![Q_1]\!]^{\mathsf{ab}_k(R)}(\mathrm{a}, \mathrm{a}) = r + 2rs + s$ which decomposes uniquely to the expression $(r + s)(r + s)$ involving only tags from $\mathsf{ab}_k(R)$. On the other hand, $[\![Q_2]\!]^{\mathsf{ab}_k(R)}(\mathrm{a}, \mathrm{a}) = r + s$ and now we have enough information to recover the results of $[\![Q_1]\!]^R$ and $[\![Q_2]\!]^R$ (by replacing $r + s$ with 2 in the calculations).

We are now ready to state the analogue of Lemma 7.15 for $\mathsf{Trio}(X)$:

Lemma 7.25 *Suppose $\bar{P}, \bar{Q} \in UCQ$ have degree $\leq k$, and suppose $\mathfrak{A}$ is a $\mathsf{Trio}(X)$-instance such that $[\![\bar{P}]\!]^{\mathfrak{A}} \not\leq_{\mathsf{Trio}(X)} [\![\bar{Q}]\!]^{\mathfrak{A}}$. Then $[\![\bar{P}]\!]^{\mathsf{ab}_k(\mathfrak{A})} \not\leq_{\mathsf{Trio}(X)} [\![\bar{Q}]\!]^{\mathsf{ab}_k(\mathfrak{A})}$.*

Proof (Sketch) Let $\mathfrak{A}' = \mathsf{ab}_k(\mathfrak{A})$ and let $\nu : S_k \to \mathsf{Trio}(X)$ be the valuation which records how the annotations from $\mathfrak{A}$ were replaced by elements of S_k. (Thus $\nu(\mathfrak{A}') = \mathfrak{A}$.) We claim that ν extends to a mapping $h_\nu : \mathsf{Trio}_k(X) \to \mathsf{Trio}(X)$ that behaves like a semiring homomorphism, so long as the computations stay within $\mathsf{Trio}_k(X)$:

1. $h_\nu(0) = 0$ and $h_\nu(1) = 1$
2. for all $a, b \in \mathsf{Trio}_k(X)$, $h_\nu(a + b) = h_\nu(a) + h_\nu(b)$
3. for all $a, b \in \mathsf{Trio}_k(X)$, if $a \cdot b \in \mathsf{Trio}_k(X)$ then $h_\nu(a \cdot b) = h_\nu(a) \cdot h_\nu(b)$

Using Lemma 7.25, we can show that h_ν exists and is uniquely determined by the above properties. Since $\bar{P}, \bar{Q}$ have degree $\leq k$, it is not hard to see that all annotations in $[\![\bar{P}]\!]^{\mathfrak{A}'}$ and $[\![\bar{Q}]\!]^{\mathfrak{A}'}$ are in $\mathsf{Trio}_k(X)$. Also, it is not hard to show that h_ν enjoys two relevant properties of proper semiring homomorphisms with respect to query evaluation:

- h_ν is compatible with the natural order: $\forall a, b \in \mathsf{Trio}_k(X)$, if $a \leq_{\mathsf{Trio}(X)} b$ then $h_\nu(a) \leq_{\mathsf{Trio}(X)} h_\nu(b)$
- Query evaluation commutes with h_ν: $h_\nu([\![\bar{P}]\!]^{\mathfrak{A}'}) = [\![\bar{P}]\!]^{h_\nu(\mathfrak{A}')} = [\![\bar{P}]\!]^{\mathfrak{A}}$ and $h_\nu([\![\bar{Q}]\!]^{\mathfrak{A}'}) = [\![\bar{Q}]\!]^{h(\mathfrak{A}')} = [\![\bar{Q}]\!]^{\mathfrak{A}}$

As a consequence, we have that $[\![\bar{P}]\!]^{\mathfrak{A}'} \leq_{\mathsf{Trio}(X)} [\![\bar{Q}]\!]^{\mathfrak{A}'}$ implies $[\![\bar{P}]\!]^{\mathfrak{A}} \leq_{\mathsf{Trio}(X)} [\![\bar{Q}]\!]^{\mathfrak{A}}$. □

Thus, we have established a bound on the size of annotations of the counterexamples that need to be considered. The last step is to bound the number of tuples and establish our "small counterexample property":

Theorem 7.26 *For UCQs $\bar{P}, \bar{Q}$ of degree $\leq k$, if $\bar{P} \not\sqsubseteq_{\mathsf{Trio}(X)} \bar{Q}$, then $[\![\bar{P}]\!]^{\mathfrak{A}} \not\leq_{\mathsf{Trio}(X)} [\![\bar{Q}]\!]^{\mathfrak{A}}$ for a k-abstractly tagged $\mathsf{Trio}(X)$-instance $\mathfrak{A}$ containing at most k tuples.*

Proof "$\Leftarrow$" is trivial. For "$\Rightarrow$", suppose $\bar{P} \not\sqsubseteq_{\mathsf{Trio}(X)} \bar{Q}$. Then for some $\mathsf{Trio}(X)$-instance $\mathfrak{A}$, we have $[\![\bar{P}]\!]^{\mathfrak{A}} \not\leq_{\mathsf{Trio}(X)} [\![\bar{Q}]\!]^{\mathfrak{A}}$. Assume that $\bar{P}, \bar{Q}$ have degree $\leq k$. By Lemma 7.25, we may assume that $\mathfrak{A}$ is k-abstractly tagged. From this point on, the proof is nearly identical to that of Theorem 7.16. Choose some tuple t such that $[\![\bar{P}]\!]^{\mathfrak{A}}(t) \not\leq_{\mathsf{Trio}(X)} [\![\bar{Q}]\!]^{\mathfrak{A}}(t)$. There must be some monomial μ in the polynomial $[\![\bar{P}]\!]^{\mathfrak{A}}(t)$ with coefficient a such that the same monomial μ in the polynomial $[\![\bar{Q}]\!]^{\mathfrak{A}}(t)$ has coefficient b and $a > b$. Now let $\mathfrak{A}'$ be the $\mathsf{Trio}(X)$-instance obtained from $\mathfrak{A}$ by discarding any tuples whose annotations do not contain a variable from μ. Observe that $\mathfrak{A}'$ has at most k tuples. Moreover, the coefficients for μ in the polynomials for $[\![\bar{P}]\!]^{\mathfrak{A}'}(t)$ and $[\![\bar{Q}]\!]^{\mathfrak{A}'}(t)$ are unchanged. Hence $[\![\bar{P}]\!]^{\mathfrak{A}'}(t) \not\leq_{\mathsf{Trio}(X)} [\![\bar{Q}]\!]^{\mathfrak{A}'}(t)$, and therefore, $[\![\bar{P}]\!]^{\mathfrak{A}'} \not\leq_{\mathsf{Trio}(X)} [\![\bar{Q}]\!]^{\mathfrak{A}'}$. This completes the proof. □

Corollary 7.27 $\mathsf{Trio}(X)$*-containment of UCQs is decidable in* PSPACE.

Finally, we note that one can find examples of UCQs showing that $\mathbb{N}[X] \not\Rightarrow \mathsf{Trio}(X)$ and $\mathsf{Trio}(X) \not\Rightarrow \mathbb{N}$, as indicated in Fig. 2(d).

8 Related Work

The seminal paper by Chandra and Merlin [6] introduced the fundamental concepts of containment mappings and canonical databases in showing the decidability of containment of CQs under set semantics and identifying its complexity as NP-complete. The extension to UCQs is due to Sagiv and Yannakakis [30]. We have built upon the techniques from these papers.

The papers by Ioannidis and Ramakrishnan [24] and Chaudhuri and Vardi [8] initiated the study of query containment under bag semantics. Chaudhuri and Vardi showed that bag-equivalence of CQs is the same as isomorphism, established the Π_2^p-hardness of checking bag-containment of CQs, and gave partial conditions for checking bag-containment (see Sect. 7 for further connections with our results).[10] Ioannidis and Ramakrishnan showed that bag-containment of UCQs is undecidable and introduced a framework of annotations from algebraic structures similar in spirit to the semiring annotations we consider.

In Sect. 7.5 we have discussed the results of Cohen et al. [11] and Cohen [9] on bag equivalence and bag-set equivalence of UCQs. The decidability of bag-containment of CQs remains open. Recent progress was made on the problem by Jayram et al. [25] who established the undecidability of checking bag-containment of CQs with inequalities.

Semiring-annotated relations are also related to the lattice-annotated relations used in *parametric databases* by Lakshmanan and Shiri [27]. That paper also studied query containment and equivalence, giving a number of positive decidability results. None of our provenance models fall into this framework (with the exception of $\mathsf{PosBool}(X)$, cf. Theorem 4.7).

We have already mentioned in Sect. 4 the paper by Grahne et al. [16], which studied containment and equivalence of positive relational queries on bilattice-annotated relations.

Green et al. [18] proposes $\mathbb{Z}$-relations, which are relations whose tuples are annotated with integer counts (positive or negative), and shows that $\mathbb{Z}$-equivalence is decidable for the full relational algebra (including difference). The proof makes essential use of the earlier results for bag semantics [8, 11].

Tan [34] showed that query containment is decidable for CQs on relations with *where-provenance* information. Our results here on why-provenance complement the where provenance results (why-provenance and where-provenance were introduced together in [4]).

Green et al. [20] showed that when K is a distributive lattice, K-containment of UCQs is the same as set containment of UCQs. This was essentially a rediscovery of an earlier result due to Buneman et al. [4] presented there in the context of queries over tree-structured data with *minimal witness why-provenance* (see Sect. 4). The result was generalized to complex values and XML trees in [14].

Cohen [10] recently initiated the study of query optimization under *combined semantics*, which generalizes bag semantics and bag-set semantics by enriching the relational algebra with a *duplicate elimination* operator. "Duplicate elimination" also makes sense for K-relations in the form of the *support* operator:

$$\mathsf{supp}(R) \overset{\text{def}}{=} \lambda t. \begin{cases} 0 & \text{if } R(t) = 0 \\ 1 & \text{otherwise} \end{cases}$$

For $K = \mathbb{N}$, this is duplicate elimination; for $K = \mathsf{PosBool}(X)$ it corresponds to the poss operator of [1] which returns the "possible" tuples of an incomplete relation.

[10]Chaudhuri and Vardi [8] also introduced the study of *bag-set semantics*, and showed that bag-set equivalence of CQs (without repeated atoms in the body) is the same as isomorphism. This was essentially a rediscovery of a well-known result in graph theory due to Lovász [28] (see also [22]), who showed that for finite relational structures F, G, if $|\mathrm{Hom}(F, H)| = |\mathrm{Hom}(G, H)|$ for all finite relational structures H, where $\mathrm{Hom}(A, B)$ is the set of homomorphisms $h : A \to B$, then $F \cong G$. In database terminology, this says that bag-set equivalence of Boolean CQs (without repeated atoms in the body) is the same as isomorphism.

It would be interesting to see whether the decidability results presented here can be extended to queries using supp.

Finally, the work in AI on *soft constraint satisfaction problems* [2] is closely related to the framework of K-relations. Their constraints over semirings are in fact the same as our K-relations and the two operations on constraints correspond indeed to relational join and projection. The semirings used in [2] are such that $+$ is idempotent and 1 is a top element in the resulting order. This rules out $\mathbb{N}$, $\mathbb{B}[X]$, $\mathbb{N}[X]$, and $\mathsf{Trio}(X)$.

9 Conclusion

We have mapped out some of the foundations of query optimization for databases with provenance information, by giving positive decidability results and complexity characterizations for checking K-containment/equivalence for CQs/UCQs, for various semirings K used to track provenance information. We also used these results to establish some necessary and some sufficient conditions for K-containment of CQs for *any* semiring K, and we showed that for the special case of CQs without self-joins and positive K, K-equivalence is the same as isomorphism. We also highlighted connections between query containment under set and bag semantics and containment under the various provenance semantics.

Moving beyond UCQs, it would be interesting to consider the same questions for Datalog programs on K-relations [20]. Unlike with UCQs, it is easy to see that $\mathbb{N}[X]$-equivalence of Datalog programs does not reduce to isomorphism, and it seems likely that the undecidability results for set semantics [33] will carry over to the forms of provenance information we have considered here. On the other hand, the positive decidability results concerning containment/equivalence of a Datalog program and a UCQ [7] might also carry over. We conjecture that when K is a distributive lattice, K-containment of Datalog programs holds exactly when the same holds for ordinary set semantics.

We assumed a Datalog-style representation for UCQs, which is expressively equivalent to the positive relational algebra ($\mathcal{RA}^+$) on K-relations, but exponentially less concise. Under set semantics, it is well-known [30] that checking containment of $\mathcal{RA}^+$ queries is correspondingly harder (Π_2^p-complete rather than NP-complete). An obvious question is how the move to an algebraic representation affects the results presented here.

Finally, semiring annotations also make sense for a positive version of XQuery on unordered XML data, as shown in [14]. It would be worthwhile to investigate how the same issues of query containment and equivalence considered here play out for annotated XML.

Acknowledgements James Cheney, Zack Ives, Grigoris Karvounarakis, and Stijn Vansummeren offered useful comments on earlier revisions of this paper. Val Tannen suggested many of the semirings and their constructions described in Sect. 4 and offered guidance and encouragement in preparing this paper. We thank the anonymous referees from ICDT 2009 for bringing the papers [11, 12] and [28] to our attention, and we thank Gösta Grahne for pointing out [16] and [27].

References

1. Antova, L., Koch, C., Olteanu, D.: From complete to incomplete information and back. In: SIGMOD (2007)
2. Bistarelli, S.: Semirings for Soft Constraint Solving and Programming. Springer, Berlin (2004)
3. Buneman, P., Cheney, J., Tan, W.-C., Vansummeren, S.: Curated databases. In: PODS (2008)
4. Buneman, P., Khanna, S., Tan, W.-C.: Why and where: A characterization of data provenance. In: ICDT (2001)
5. Buneman, P., Khanna, S., Tan, W.C.: On propagation of deletions and annotations through views. In: PODS (2002)
6. Chandra, A.K., Merlin, P.M.: Optimal implementation of conjunctive queries in relational data bases. In: STOC, pp. 77–90 (1977)
7. Chaudhuri, S., Vardi, M.Y.: On the equivalence of recursive and nonrecursive datalog programs. In: PODS (1992)
8. Chaudhuri, S., Vardi, M.Y.: Optimization of *real* conjunctive queries. In: PODS (1993)
9. Cohen, S.: Containment of aggregate queries. SIGMOD Rec. **34**(1), 77–85 (2005)
10. Cohen, S.: Equivalence of queries combining set and bag-set semantics. In: PODS (2006)
11. Cohen, S., Nutt, W., Serebrenik, A.: Rewriting aggregate queries using views. In: PODS (1999)
12. Cohen, S., Sagiv, Y., Nutt, W.: Equivalences among aggregate queries with negation. ACM Trans. Comput. Log. **6**(2), 328–360 (2005)
13. Cui, Y., Widom, J., Wiener, J.L.: Tracing the lineage of view data in a warehousing environment. TODS, **25**(2) (2000)
14. Foster, J.N., Green, T.J., Tannen, V.: Annotated XML: Queries and provenance. In: PODS (2008)
15. Fuhr, N., Rölleke, T.: A probabilistic relational algebra for the integration of information retrieval and database systems. TOIS **14**(1), 32–66 (1997)
16. Grahne, G., Spyratos, N., Stamate, D.: Semantics and containment of queries with internal and external conjunctions. In: ICDT (1997)
17. Green, T.J.: Containment of conjunctive queries on annotated relations. In: ICDT (2009)
18. Green, T.J., Ives, Z.G., Tannen, V.: Reconcilable differences. In: ICDT (2009)
19. Green, T.J., Karvounarakis, G., Ives, Z.G., Tannen, V.: Update exchange with mappings and provenance. In: VLDB (2007)
20. Green, T.J., Karvounarakis, G., Tannen, V.: Provenance semirings. In: PODS (2007)
21. Green, T.J., Tannen, V.: Models for incomplete and probabilistic information. In: IIDB, March 2006 (2006)
22. Hell, P., Nešetřil, J.: Graphs and Homomorphisms. Oxford University Press, Oxford (2004)
23. Imieliński, T., Witold Lipski, J.: Incomplete information in relational databases. J. ACM, **31**(4) (1984)
24. Ioannidis, Y.E., Ramakrishnan, R.: Containment of conjunctive queries: Beyond relations as sets. TODS **20**(3), 288–324 (1995)
25. Jayram, T.S., Kolaitis, P.G., Vee, E.: The containment problem for *real* conjunctive queries with inequalities. In: PODS (2006)
26. Köbler, J., Schöning, U., Torán, J.: The Graph Isomorphism Problem: its Structural Complexity. Birkhäuser, Basel (1993)
27. Lakshmanan, L.V.S., Shiri, N.: A parametric approach to deductive databases with uncertainty. IEEE Trans. Knowl. Data Eng. **13**(4), 554–570 (2001)
28. Lovász, L.: Operations with structures. Acta Math. Hung. **18**(3–4), 321–328 (1967)
29. Nutt, W., Sagiv, Y., Shurin, S.: Deciding equivalences among aggregate queries. In: PODS (1998)
30. Sagiv, Y., Yannakakis, M.: Equivalences among relational expressions with the union and difference operators. J. ACM **27**(4), 633–655 (1980)
31. Sarma, A.D., Theobald, M., Widom, J.: Exploiting lineage for confidence computation in uncertain and probabilistic databases. In: ICDE (2008)
32. Senellart, P., Abiteboul, S.: On the complexity of managing probabilistic XML data. In: PODS (2007)
33. Shmueli, O.: Equivalence of datalog queries is undecidable. J. Logic Programming **15** (1993)
34. Tan, W.-C.: Containment of relational queries with annotation propagation. In: DBPL, September 2003 (2003)
35. Zimányi, E.: Query evaluation in probabilistic relational databases. TCS, **171**(1–2) (1997)