Conjunctive Queries with Equations and Disequations for Databases over Semirings

Guillermo Badia[a], Carles Noguera[b], Gaia Petreni[c], Val Tannen[d]

(a): University of Queensland
(b), (c): DIISM, University of Siena
(d): University of Pennsylvania

September 2025

Goal of the talk

- Databases with annotated relations: tuple annotations used to track *provenance*, providing information on how the query results depend on atomic facts.
- *Containment problem* for conjunctive queries *with equations and disequations*: are all the answers to query P also answers to query Q?
- **Are these problems decidable?** For a positive answer: find equivalence with the existence of specific types of mappings between queries (Chandra-Merlin strategy).
- Complexity results for the containment problem.
- Containment for regular CQs over semiring-annotated databases is well-understood since Green (2011).

Take-home message: results of this talk

Klug (1988) and Van der Meyden (1997) show that containment for $\{=\neq\}-$CQs on standard databases is Π_2^p-c.

Cohen, Nutt & Sagiv (2007) give a characterization in terms of mappings between **families of queries**.

Type	Complexity	Known semirings
$\{=\neq\}-$Can. map. (identifications)	Π_2^p-c	$\mathbb{B}$ (**Klug, VdM**), Distr. Latt. (e.g. PosBool[X])
$\{=\neq\}-$Hom. coverage for rel. atoms (identifications)	Π_2^p-c	Lin[X]
$\{=\neq\}-$Injective for rel. atoms (identifications)	Π_2^p-c	Sorp[X]
$\{=\neq\}-$Surjective for rel. atoms (identifications)	Π_2^p-c	Why[X], Trio[X]
$\{=\neq\}-$Bijective for rel. atoms (identifications)	in Π_2^p and NP-hard	$\mathbb{N}[X]$, $\mathbb{B}[X]$
n/a	Undecidable	$\mathbb{N}$ (**Kolaitis et al.**)

Semirings

Definition

A **semiring** is an algebra $\mathbf{K} = (K, +, \cdot, 0, 1)$ where:

- $(K, +, 0)$ is a commutative monoid.
- $(K, \cdot, 1)$ is a monoid.
- $\cdot$ distributes over $+$:

$$x \cdot (y + z) = (x \cdot y) + (x \cdot z)$$

$$(y + z) \cdot x = (y \cdot x) + (z \cdot x)$$

- 0 is absorbing: $x \cdot 0 = 0 \cdot x = 0$.

$\mathbf{K}$ is **commutative** if $\cdot$ is commutative.
Easy examples: $\mathbb{N}$, $\mathbb{B}$ (two-element Boolean algebra).

Semirings for Provenance

Definition (Green et al. 2007)

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

Definition (Green,2009)

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

Semirings for Provenance (cont'd)

Let $f : \mathbb{N}[X] \to \mathbb{N}[X]$ be the mapping that "drops exponents", e.g.,

$$f(2x^2y + 3xy + 2z^3 + 1) = 5xy + 2z + 1.$$

Denote by $\approx_f$ the congruence relation on $\mathbb{N}[X]$ defined by

$$a \approx_f b \Longleftrightarrow f(a) = f(b).$$

Definition (Benjelloun et al., 2008)

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

The why-provenance of a tuple is the set of sets of "contributing" source tuples and it can be captured using the following semiring.

Definition (Buneman et al., 2008)

The **why-provenance** semiring for X is $(\mathrm{Why}(X), \cup, \Cup, \emptyset, \{\emptyset\})$ where $\mathrm{Why}(X) = \mathcal{P}_{\mathrm{fin}}(\mathcal{P}_{\mathrm{fin}}(X))$ and $\Cup$ denotes pairwise union:

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

Definition

The **lineage semiring** for X is $(\mathcal{P}_{\mathsf{fin}}(X) \cup \{\bot\}, +, \cdot, \bot, \emptyset)$ where

- X is a set of variables,
- $\bot + S = S + \bot = S$,
- $\bot \cdot S = S \cdot \bot = \bot$,
- $S + T = S \cdot T = S \cup T$ if $S, T \neq \bot$.

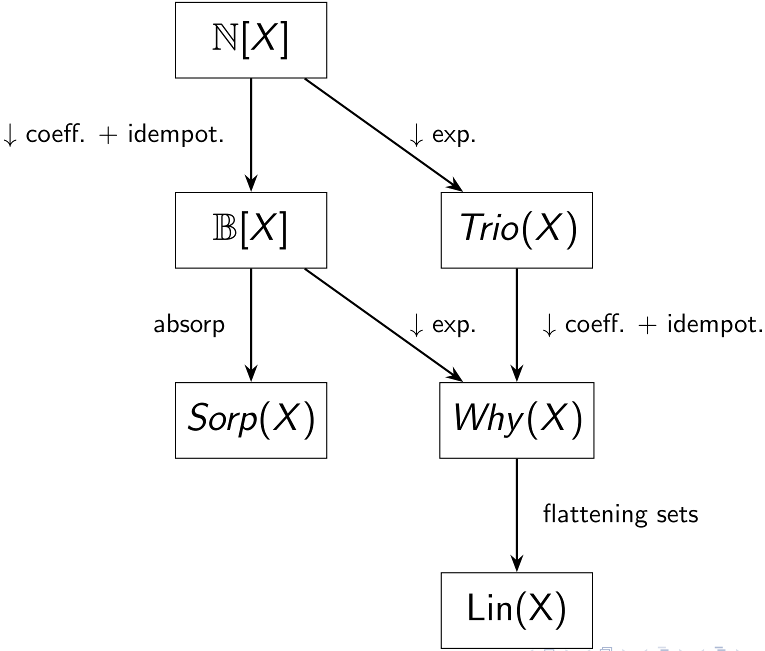
A commutative semiring $\mathbf{K} = (K, +, \cdot, 0, 1)$ is **absorptive** if for every $a, b \in K$

$$a + ab = a.$$

Denote by $\approx$ the smallest congruence on $\mathbb{N}[X]$ that identifies polynomials according to absorption.

Definition

The **absorptive** semiring for X, $\mathrm{Sorp}(X)$, is the quotient semiring of $\mathbb{N}[X]$ by $\approx$.


ℕ[X]
↓ coeff. + idempot.
↓ exp.
𝔹[X]
Trio(X)
absorp
↓ exp.
↓ coeff. + idempot.
Sorp(X)
Why(X)
flattening sets
Lin(X)

Fix a countable domain $\mathbb{D}$ of individuals and a semiring $\mathbf{K} = (K, +, \cdot, 0, 1)$.

Definition

An n-ary K-**relation** is a function $R : \mathbb{D}^n \to K$ such that its support, defined by

$$supp(R) = \{t : t \in \mathbb{D}^n, R(t) \neq 0\}$$

is finite.

A $\mathbb{B}$-relation:

Name	City	
James Bond	*Brisbane*	1
James Bond	*Tokyo*	0
Ethan Hunt	*Fukuoka*	1

Set semantics:
2 tuples

A $\mathbb{N}$-relation:

Name	City	
James Bond	*Brisbane*	5
James Bond	*Tokyo*	0
Ethan Hunt	*Fukuoka*	3

Bag semantics:
8 tuples

Definition

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.

Definition

A K-**instance** is a mapping from predicate symbols to 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 $\mathfrak{A}$.

Example

Where $\mathbb{N}$ is the semiring of natural numbers:

$R^{\mathfrak{A}} \stackrel{\text{def}}{=}$

a	b	2
d	b	1
b	c	1

$S^{\mathfrak{A}} \stackrel{\text{def}}{=}$

b	g	f	3
d	a	b	1

Definition

A **conjunctive query** (CQ) is an expression of the form

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

where

- $Q(\bar{u})$ is the **head** of the query (head(Q)),
- the multiset (bag) of **atoms** $R_1(\bar{u}_1), \ldots, R_n(\bar{u}_n)$ is the **body** of the query (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; each variable appearing in the head also appears somewhere in the body.

Think of CQs as existential formulas where only conjunctions are allowed!

Definition

A **valuation** is a function $v : \mathrm{vars}(Q) \to \mathbb{D}$.

Valuations operate component-wise on tuples in the expected way.

Let Q be a CQ

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

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

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

and the sums and products are in K.

The Natural Order

Let $(K, +, \cdot, 0, 1)$ be a semiring and define

$$a \leq b \iff \exists c : a + c = b.$$

When $\leq$ is a partial order we say that K is **naturally-ordered**.

Example

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.

Definition

Let K be a naturally-ordered semiring and let R_1, R_2 be two K-relations.
R_1 is **contained** in R_2 ($R_1 \leq_K R_2$) iff

$$\forall t \in \mathbb{D}^n,\ R_1(t) \leq R_2(t)$$

Definition

Consider two queries P, Q.
P is **contained** in Q ($P \sqsubseteq_K Q$) iff

$$\forall K\text{-instance } \mathfrak{A},\ [\![P]\!]^{\mathfrak{A}} \leq_K [\![Q]\!]^{\mathfrak{A}}$$

CQ with equations and disequations

Definition

A $\{=, \neq\}$-**CQ** is simply a CQ where literals of the form $x = y$ and $x \neq y$ are allowed in the body of the query. (**In evaluating the query in a semiring these literals take only values** 0 **or** 1 **in the usual manner.**)

We focus on queries that are:

- **safe**: the only variables allowed are those in the active domain of the query.
- **consistent**: $x \neq x$ does not follow logically from the body of the query for any variable x.

Completions and identifications

Definition

Given a $\{=,\neq\}$-CQ Q, a **completion** Q' of Q comes from adding either $x = y$ or $x \neq y$ for every couple of variables x, y that appear in a relational atom of Q, as long as the new query is consistent.

Consider the equivalence relation between variables of a completion Q' given by

$$x \equiv y \text{ iff } x = y \text{ is a logical consequence of the body of } Q'.$$

A **canonical substitution** maps all elements in an equivalence class to a representative.

Definition (Cohen et al., 2007)

An **identification** Q^{id} of a completion Q' comes by eliminating all equations by applying a canonical substitution to Q'.

Consider the queries

$$q := \exists x, y(R(x, y) \wedge R(y, x))$$

and

$$p := \exists x, y(R(x, y) \wedge x = y).$$

Observe that q has the following two possible identifications:

1. $\exists x, y(R(x, y) \wedge R(y, x) \wedge x \neq y)$
2. $\exists x(R(x, x) \wedge R(x, x))$,

where $\exists x(R(x, x) \wedge R(x, x))$ comes from the completion $\exists x, y(R(x, y) \wedge R(y, x) \wedge x = y)$ and the canonical substitution that sends y to x.
Similarly, p (which is already a completion of itself) has only the following identification:

1. $\exists x R(x, x)$.

Containment mappings

Definition (Cohen et al., 2007)

Given two $\{=, \neq\}$-CQs, q_1 and q_2, a $\{=, \neq\}$-**containment mapping** h from q_1 to q_2 is a function from the variables of q_1 to q_2 that preserves literals in the following sense:

- if the atom $l(\overline{x})$ appears in q_1, the atom $l(h(\overline{x}))$ appears in q_2,
- if the equation (disequation) $l(\overline{x})$ appears in q_1, the atom $l(h(\overline{x}))$ is a **logical consequence** of the body of q_2.

A containment mapping is **one-to-one** or **injective** for relational atoms if the multiset of images of atoms of Q_1 is bag-contained in the multiset of relational atoms of Q_2.

Also, h is **surjective** for relational atoms if the multiset of relational atoms of Q_2 is equal to the multiset of images of atoms of Q_1. (Surjective on relational atoms gives also surjective as a mapping on variables.)

Exact for relational atoms means being both surjective and injective.

Canonical database

Given an identification q^{id} of some $\{=,\neq\}$-CQ q, one can build its **canonical database** $D^{q^{id}}$ as follows.

- For any relation R of the schema of q^{id} we let $R_{D^{q^{id}}}$ contain the tuple of $(x_1, \ldots, x_n)$ iff $R(x_1, \ldots, x_n)$ is an atom in q^{id}.
- By construction, if the identification $\{=,\neq\}$-CQ q^{id} is a formula $\phi(u)$, then u is an answer to the query ϕ in the database $D^{q^{id}}$.

The Boolean case

Theorem (Klug (1988), Kolaitis et al. (1998), Cohen et al. (2007))

For each $\{=, \neq\}$*-CQs* Q_1, Q_2 *with the same tuple of free variables* $\overline{u}$*, the following are equivalent:*

1. $Q_1 \sqsubseteq_{\mathbb{B}} Q_2$.
2. *For every identification* Q_1^{id} *of* Q_1*, there is a* $\{=, \neq\}$*-containment mapping* $h_{Q_1^{id}} : Q_2 \longrightarrow Q_1^{id}$.

The case of distributive lattices

Theorem

Let P and Q be $\{=, \neq\}$-CQs with the same tuple of free variables $\overline{u}$, and K a bounded distributive lattice. Then, the following are equivalent:

1. $P \sqsubseteq_K Q$.
2. *For every identification P^{id} of P, there is a $\{=, \neq\}$-containment mapping $h_{P^{id}} : Q \longrightarrow P^{id}$.*

The case of various provenance semirings

Use the **abstractly tagged version of canonical databases** introduced by Green (2011). (E.g. for $\mathbb{N}[X]$, each tuple of the canonical database gets annotated with a different $p \in X$.)

Theorem

For $\{=, \neq\}$-CQs P, Q with the same tuple of free variables $\overline{u}$, the following are equivalent where $K \in \{\mathbb{B}[X], \mathbb{N}[X]\}$:

1. $P \sqsubseteq_K Q$,
2. *For every identification P^{id} of P, we have that* $[\![P^{id}]\!]^{can_K(P^{id})} \leq [\![Q]\!]^{can_K(P^{id})}$.
3. *For every identification P^{id} of P, there is an $\{=, \neq\}$-containment mapping $h_{P^{id}} : Q \longrightarrow P^{id}$* **exact for relational atoms**.

Theorem

For $\{=,\neq\}$*-CQs* P, Q *with the same tuple of free variables* $\overline{u}$*, the following are equivalent:*

1. $P \sqsubseteq_{Sorp[X]} Q$,
2. *For every identification* P^{id} *of* P*, we have that* $[\![P^{id}]\!]^{can_{Sorp[X]}(P^{id})} \leq [\![Q]\!]^{can_{Sorp[X]}(P^{id})}$.
3. *For every identification* P^{id} *of* P*, there is an* $\{=,\neq\}$*-containment mapping* $h_{P^{id}} : Q \longrightarrow P^{id}$ **injective for relational atoms**.

Theorem

For $\{=,\neq\}$*-CQs* P, Q *with the same tuple of free variables* $\overline{u}$*, the following are equivalent where* $K \in \{Why[X], Trio[X]\}$*:*

1. $P \sqsubseteq_K Q$,
2. *For every identification* P^{id} *of* P*, we have that* $[\![P^{id}]\!]^{can_K(P^{id})} \leq [\![Q]\!]^{can_K(P^{id})}$.
3. *For every identification* P^{id} *of* P*, there is an* $\{=,\neq\}$*-containment mapping* $h_{P^{id}} : Q \longrightarrow P^{id}$ **onto for relational atoms**.

Theorem

For $\{=,\neq\}$-CQs P, Q with the same tuple of free variables $\overline{u}$, the following are equivalent:

1. $P \sqsubseteq_{Lin[X]} Q$,
2. *For every identification P^{id} of P, we have that* $[\![P^{id}]\!]^{can_{Lin[X]}(P^{id})} \leq [\![Q]\!]^{can_{Lin[X]}(P^{id})}$.
3. *For every identification P^{id} of P, and every relational atom $R(\overline{y})$ of P^{id} there is a $\{=,\neq\}$-containment mapping $h_{P^{id}} : Q \longrightarrow P^{id}$ with $R(\overline{y})$ in the image of $h_{P^{id}}$.*

Theorem

The containment problems for $\{=,\neq\}$-CQs over $Lin[X]$, $Trio[X]$, $Why[X]$, $Sorp[X]$, $\mathbb{N}[X]$ and $\mathbb{B}[X]$ are in Π_2^p.

Theorem (Van der Meyden 1997)

The following problem is Π_2^p-hard: Given two safe conjunctive $\{=,\neq\}$-queries Q_1, Q_2, is it true that for every identification Q_1^{id} of Q_1 there is a $\{=,\neq\}$-canonical mapping $h_{Q_1^{id}} : Q_2 \longrightarrow Q_1^{id}$?

Theorem

The containment problem for $\{=,\neq\}$-CQs over $Lin[X]$, $Sorp[X]$, $Why[X]$ and $Trio[X]$, is Π_2^p-complete.

Next steps

- Extend these results to Unions (i.e. disjunctions) of CQs;
- Add negated atoms;
- Go beyond containment and study equivalence.