A note on computing certain answers to queries over incomplete databases

Marcelo Arenas[1] Elena Botoeva[2]
Egor V. Kostylev[3] Vladislav Ryzhikov[2]

[1]Pontificia Universidad Católica de Chile
[2]Free University of Bozen-Bolzan
[3]University of Oxford

Complete and incomplete databases

A complete database:

Employee	name	salary
	Elena	110K
	John	70K
	Ringo	80K

Complete and incomplete databases

A complete database:

Employee	name	salary
	Elena	110K
	John	70K
	Ringo	80K

Incomplete databases:

Employee	name	salary
	Elena	110K
	John	$\perp_1$
	$\perp_2$	80K

Employee	name	salary
	Elena	110K
	John	$\perp_1$
	Ringo	$\perp_1$

Complete and incomplete databases

Fix a set **Const** of constants and a set **Null** of nulls

In an incomplete database I each k-ary relation is a finite subset of $(\mathbf{Const} \cup \mathbf{Null})^k$

- $\mathrm{adom}(I)$ is the set of constants and nulls occurring in I
- If $\mathrm{adom}(I) \subseteq \mathbf{Const}$, then I is said to be a complete database

Complete and incomplete databases

Fix a set **Const** of constants and a set **Null** of nulls

In an incomplete database I each k-ary relation is a finite subset of $(\mathbf{Const} \cup \mathbf{Null})^k$

- $\mathrm{adom}(I)$ is the set of constants and nulls occurring in I
- If $\mathrm{adom}(I) \subseteq \mathbf{Const}$, then I is said to be a complete database

We use I, I_1, I', ... to denote an incomplete database, and D, D_1, D', ... to denote a complete database

The semantics of an incomplete database

The semantics of an incomplete database I is defined in terms of its representations

- A representation is a complete database that is considered as possible interpretation of I

The semantics of an incomplete database

The semantics of an incomplete database I is defined in terms of its representations

- A representation is a complete database that is considered as possible interpretation of I

To construct a representation of I we need to assign constants to the nulls occurring in I

The semantics of an incomplete database

The semantics of an incomplete database I is defined in terms of its representations

- A representation is a complete database that is considered as possible interpretation of I

To construct a representation of I we need to assign constants to the nulls occurring in I

- A valuation of I is a function $v : \text{adom}(I) \to \textbf{Const}$ that is the identity on $\textbf{Const}(I)$

The representations of an incomplete database under the open-world assumption

Definition

The set of representations of I is defined as:

$$\llbracket I \rrbracket \;=\; \{D \mid D \text{ is a complete database and } v(I) \subseteq D \text{ for some valuation } v \text{ of } I\}$$

Some examples of representations

name	salary
Elena	110K
John	$\perp_1$
$\perp_2$	80K

Some examples of representations

name	salary
Elena	110K
John	$\perp_1$
$\perp_2$	80K

$\Longrightarrow$

$$v(\perp_1) = 120\text{K}$$
$$v(\perp_2) = \text{Paul}$$

Some examples of representations

name	salary
Elena	110K
John	$\perp_1$
$\perp_2$	80K

$\Longrightarrow$

$$v(\perp_1) = 120\text{K}$$
$$v(\perp_2) = \text{Paul}$$

$\Longrightarrow$

name	salary
Elena	110K
John	120K
Paul	80K

Some examples of representations

name	salary
Elena	110K
John	$\perp_1$
$\perp_2$	80K

$\Longrightarrow$ $v(\perp_1) = 120\text{K}$, $v(\perp_2) = \text{Paul}$ $\Longrightarrow$

name	salary
Elena	110K
John	120K
Paul	80K

$\Longrightarrow$ $v(\perp_1) = 120\text{K}$, $v(\perp_2) = \text{Paul}$

Some examples of representations

name	salary
Elena	110K
John	$\bot_1$
$\bot_2$	80K

$\Longrightarrow$ $v(\bot_1) = 120\text{K}$, $v(\bot_2) = \text{Paul}$ $\Longrightarrow$

name	salary
Elena	110K
John	120K
Paul	80K

$\Longrightarrow$ $v(\bot_1) = 120\text{K}$, $v(\bot_2) = \text{Paul}$ $\Longrightarrow$

name	salary
Elena	110K
John	120K
Paul	80K
Ringo	110K

Some examples of representations

name	salary
Elena	110K
John	$\perp_1$
Ringo	$\perp_1$

Some examples of representations

name	salary
Elena	110K
John	$\perp_1$
Ringo	$\perp_1$

$\Longrightarrow \quad v(\perp_1) = 120K$

Some examples of representations

name	salary
Elena	110K
John	$\perp_1$
Ringo	$\perp_1$

$\Longrightarrow$ $v(\perp_1) = 120\text{K}$ $\Longrightarrow$

name	salary
Elena	110K
John	120K
Ringo	120K

Some examples of representations

name	salary
Elena	110K
John	$\perp_1$
Ringo	$\perp_1$

$\Longrightarrow$ $v(\perp_1) = 120\text{K}$ $\Longrightarrow$

name	salary
Elena	110K
John	120K
Ringo	120K

$\Longrightarrow$ $v(\perp_1) = 140\text{K}$

Some examples of representations

name	salary
Elena	110K
John	$\bot_1$
Ringo	$\bot_1$

$\Longrightarrow$ $v(\bot_1) = 120\text{K}$ $\Longrightarrow$

name	salary
Elena	110K
John	120K
Ringo	120K

$\Longrightarrow$ $v(\bot_1) = 140\text{K}$ $\Longrightarrow$

name	salary
Elena	110K
John	140K
Ringo	140K
Paul	110K
George	80K

Query answering over incomplete databases

A query Q is a function that assigns to each complete database D a complete database $Q(D)$

Query answering over incomplete databases

A query Q is a function that assigns to each complete database D a complete database $Q(D)$

To define the semantics of a query over an incomplete database we need to introduce some fundamental notions

Comparing the amount of information of incomplete databases

I_2 is at least as informative as I_1 if $[\![I_2]\!] \subseteq [\![I_1]\!]$

- I is more informative if it has less representations [L16]

We use notation $I_1 \preceq I_2$ to indicate that I_2 is at least as informative as I_1

An order on incomplete databases

We have that:

name	salary
Elena	110K
John	$\bot_1$
Ringo	$\bot_2$

$\preceq$

name	salary
Elena	110K
John	$\bot_3$
Ringo	$\bot_3$

An order on incomplete databases

We have that:

name	salary
Elena	110K
John	$\bot_1$
Ringo	$\bot_2$

$\preceq$

name	salary
Elena	110K
John	$\bot_3$
Ringo	$\bot_3$

Since:

$[\![$

name	salary
Elena	110K
John	$\bot_3$
Ringo	$\bot_3$

$]\!]$ $\subseteq$ $[\![$

name	salary
Elena	110K
John	$\bot_1$
Ringo	$\bot_2$

$]\!]$

The notion of greatest lower bound

Definition

Let $\mathcal{I}$ be a set of incomplete databases

The notion of greatest lower bound

Definition

Let $\mathcal{I}$ be a set of incomplete databases

- I is a lower bound for $\mathcal{I}$ if $I \preceq I'$ for every $I' \in \mathcal{I}$

The notion of greatest lower bound

Definition

Let $\mathcal{I}$ be a set of incomplete databases

- I is a lower bound for $\mathcal{I}$ if $I \preceq I'$ for every $I' \in \mathcal{I}$
- I is a greatest lower bound for $\mathcal{I}$ if I is a lower bound for $\mathcal{I}$ and for every lower bound I' for $\mathcal{I}$, it holds that $I' \preceq I$

The notion of greatest lower bound

Definition

Let $\mathcal{I}$ be a set of incomplete databases

- I is a lower bound for $\mathcal{I}$ if $I \preceq I'$ for every $I' \in \mathcal{I}$
- I is a greatest lower bound for $\mathcal{I}$ if I is a lower bound for $\mathcal{I}$ and for every lower bound I' for $\mathcal{I}$, it holds that $I' \preceq I$

The set of all greatest lower bounds of $\mathcal{I}$ is denoted by $\mathbf{glb}(\mathcal{I})$

The notion of certain answer as object

Definition (Certain answer as object [L16])

$I^\star$ is a certain answer as object to Q over I if

The notion of certain answer as object

Definition (Certain answer as object [L16])

$I^\star$ is a certain answer as object to Q over I if

$$I^\star \quad \in \quad \mathbf{glb}(\{Q(D) \mid D \in [\![I]\!]\})$$

The notion of certain answer as object

Definition (Certain answer as object [L16])

$I^\star$ is a certain answer as object to Q over I if

$$I^\star \quad \in \quad \mathbf{glb}(\{Q(D) \mid D \in [\![I]\!]\})$$

Two greatest lower bounds I_1, I_2 of $\{Q(D) \mid D \in [\![I]\!]\}$ are equivalent in terms of the information ordering: $I_1 \preceq I_2$ and $I_2 \preceq I_1$

- We choose any greatest lower bound of $\{Q(D) \mid D \in [\![I]\!]\}$, and we talk about **the** certain answer to Q over I, which is denoted by $\mathbf{cert}(Q, I)$

Certain answers: a first example

Consider the query $Q(x, y) = R(x, y) \wedge x \neq y$ over the following incomplete database I:

R		
	a	$\perp_1$

Certain answers: a first example

Consider the query $Q(x, y) = R(x, y) \wedge x \neq y$ over the following incomplete database I:

R		
	a	$\perp_1$

We obtain the following answers over the representations of I:

Certain answers: a first example

Consider the query $Q(x, y) = R(x, y) \wedge x \neq y$ over the following incomplete database I:

R		
	a	$\bot_1$

We obtain the following answers over the representations of I:

R		
	a	a

R		
	a	b

Certain answers: a first example

Consider the query $Q(x, y) = R(x, y) \wedge x \neq y$ over the following incomplete database I:

R	
a	$\perp_1$

We obtain the following answers over the representations of I:

R	
a	a

$\Longrightarrow \quad \emptyset$

R	
a	b

Certain answers: a first example

Consider the query $Q(x, y) = R(x, y) \wedge x \neq y$ over the following incomplete database I:

R	
a	$\perp_1$

We obtain the following answers over the representations of I:

R	
a	a

$\Longrightarrow \quad \emptyset$

R	
a	b

$\Longrightarrow$

a	b

Certain answers: a first example

Consider the query $Q(x, y) = R(x, y) \wedge x \neq y$ over the following incomplete database I:

$$\begin{array}{c|cc} R & & \\ \hline & a & \perp_1 \end{array}$$

We obtain the following answers over the representations of I:

$$\begin{array}{c|cc} R & & \\ \hline & a & a \end{array} \quad \Longrightarrow \quad \emptyset$$

$$\begin{array}{c|cc} R & & \\ \hline & a & b \end{array} \quad \Longrightarrow \quad \begin{array}{|c|c|} \hline a & b \\ \hline \end{array}$$

Thus, we have that $\mathbf{cert}(Q, I)$ is the empty instance

Certain answers: a second example

Consider the same query $Q(x, y) = R(x, y) \wedge x \neq y$ but now over the following incomplete database I:

R	
a	$\perp_1$
b	$\perp_1$

Certain answers: a second example

Consider the same query $Q(x,y) = R(x,y) \wedge x \neq y$ but now over the following incomplete database I:

R		
	a	$\perp_1$
	b	$\perp_1$

We obtain the following answers over the representations of I:

Certain answers: a second example

Consider the same query $Q(x, y) = R(x, y) \wedge x \neq y$ but now over the following incomplete database I:

R	
a	$\perp_1$
b	$\perp_1$

We obtain the following answers over the representations of I:

R	
a	a
b	a

R	
a	b
b	b

R	
a	c
b	c

Certain answers: a second example

Consider the same query $Q(x, y) = R(x, y) \wedge x \neq y$ but now over the following incomplete database I:

R		
	a	$\perp_1$
	b	$\perp_1$

We obtain the following answers over the representations of I:

R		
	a	a
	b	a

$\Longrightarrow$

b	a

R		
	a	b
	b	b

R		
	a	c
	b	c

Certain answers: a second example

Consider the same query $Q(x, y) = R(x, y) \land x \neq y$ but now over the following incomplete database I:

R	
a	$\perp_1$
b	$\perp_1$

We obtain the following answers over the representations of I:

R	
a	a
b	a

$\Longrightarrow$

b	a

R	
a	b
b	b

$\Longrightarrow$

a	b

R	
a	c
b	c

Certain answers: a second example

Consider the same query $Q(x, y) = R(x, y) \wedge x \neq y$ but now over the following incomplete database I:

$$\begin{array}{c|c|c|}
R & & \\
\hline
 & a & \bot_1 \\
 & b & \bot_1 \\
\hline
\end{array}$$

We obtain the following answers over the representations of I:

$$\begin{array}{c|c|c|}
R & & \\
\hline
 & a & a \\
 & b & a \\
\hline
\end{array} \quad \Longrightarrow \quad \begin{array}{|c|c|}
\hline
b & a \\
\hline
\end{array}$$

$$\begin{array}{c|c|c|}
R & & \\
\hline
 & a & b \\
 & b & b \\
\hline
\end{array} \quad \Longrightarrow \quad \begin{array}{|c|c|}
\hline
a & b \\
\hline
\end{array}$$

$$\begin{array}{c|c|c|}
R & & \\
\hline
 & a & c \\
 & b & c \\
\hline
\end{array} \quad \Longrightarrow \quad \begin{array}{|c|c|}
\hline
a & c \\
b & c \\
\hline
\end{array}$$

Certain answers: a second example

What is a greatest lower bound of the following set?

$$\left\{ \quad \begin{array}{|c|c|}\hline b & a \\ \hline \end{array}, \quad \begin{array}{|c|c|}\hline a & b \\ \hline \end{array}, \quad \begin{array}{|c|c|}\hline a & c \\ b & c \\ \hline \end{array}, \quad \ldots \quad \right\}$$

Certain answers: a second example

What is a greatest lower bound of the following set?

$$\left\{ \quad \boxed{\begin{array}{c|c} b & a \end{array}}, \quad \boxed{\begin{array}{c|c} a & b \end{array}}, \quad \boxed{\begin{array}{c|c} a & c \\ b & c \end{array}}, \quad \cdots \quad \right\}$$

In this case we have that **cert**(Q, I) is the following incomplete database:

$\perp_1$	$\perp_2$

Certain answers: a second example

What is a greatest lower bound of the following set?

$$\left\{ \quad \begin{array}{|c|c|}\hline b & a \\ \hline\end{array}\,, \quad \begin{array}{|c|c|}\hline a & b \\ \hline\end{array}\,, \quad \begin{array}{|c|c|}\hline a & c \\ b & c \\ \hline\end{array}\,, \quad \ldots \quad \right\}$$

In this case we have that **cert**(Q, I) is the following incomplete database:

$$\begin{array}{|c|c|}\hline \perp_1 & \perp_2 \\ \hline\end{array}$$

Thus, for every $D \in [\![I]\!]$ we know that $Q(D)$ contains at least one tuple

- We do not have more information that can be stored in the form of an incomplete database

Certain answers: a second example

What is a greatest lower bound of the following set?

$$\left\{ \quad \begin{array}{|c|c|}\hline b & a \\ \hline\end{array}, \quad \begin{array}{|c|c|}\hline a & b \\ \hline\end{array}, \quad \begin{array}{|c|c|}\hline a & c \\ b & c \\ \hline\end{array}, \quad \ldots \quad \right\}$$

In this case we have that $\mathbf{cert}(Q, I)$ is the following incomplete database:

$$\begin{array}{|c|c|}\hline \bot_1 & \bot_2 \\ \hline\end{array}$$

Thus, for every $D \in [\![I]\!]$ we know that $Q(D)$ contains at least one tuple

- We do not have more information that can be stored in the form of an incomplete database
- The situation will be different if we use conditional tables as we can add the condition $\bot_1 \neq \bot_2$

Can $\mathbf{cert}(Q, I)$ be computed?

If Q is a union of conjunctive queries, then $\mathbf{cert}(Q, I)$ can be computed by using naïve evaluation

- Q is evaluated by treating the nulls in I as constants

Can **cert**(Q, I) be computed?

If Q is a union of conjunctive queries, then **cert**(Q, I) can be computed by using naïve evaluation

- Q is evaluated by treating the nulls in I as constants

Our research question

How can **cert**(Q, I) be computed if Q is a union of conjunctive queries with inequalities?

Can $\mathbf{cert}(Q, I)$ be computed?

If Q is a union of conjunctive queries, then $\mathbf{cert}(Q, I)$ can be computed by using naïve evaluation

- Q is evaluated by treating the nulls in I as constants

Our research question

How can $\mathbf{cert}(Q, I)$ be computed if Q is a union of conjunctive queries with inequalities?

- Can this be done efficiently?

Take-home message: there is no efficient algorithm

Take-home message: there is no efficient algorithm

Proposition

Given a union of conjunctive queries with inequalities Q, there exists a double-exponential algorithm that, given an incomplete database I, computes $\mathbf{cert}(Q, I)$

Take-home message: there is no efficient algorithm

Proposition

Given a union of conjunctive queries with inequalities Q, there exists a double-exponential algorithm that, given an incomplete database I, computes $\mathbf{cert}(Q, I)$

Theorem

There exists a conjunctive query with inequalities Q and a family of incomplete databases $\{I_n\}_{n \geq 0}$

Take-home message: there is no efficient algorithm

Proposition

Given a union of conjunctive queries with inequalities Q, there exists a double-exponential algorithm that, given an incomplete database I, computes $\mathbf{cert}(Q, I)$

Theorem

There exists a conjunctive query with inequalities Q and a family of incomplete databases $\{I_n\}_{n\geq 0}$ such that the size of the smallest element in $\mathbf{glb}(\{Q(D) \mid D \in [\![I_n]\!]\})$ *grows exponentially in the size of I_n*

Take-home message: there is no efficient algorithm

Proposition

Given a union of conjunctive queries with inequalities Q, there exists a double-exponential algorithm that, given an incomplete database I, computes $\mathbf{cert}(Q, I)$

Theorem

There exists a conjunctive query with inequalities Q and a family of incomplete databases $\{I_n\}_{n\geq 0}$ such that the size of the smallest element in $\mathbf{glb}(\{Q(D) \mid D \in [\![I_n]\!]\})$ *grows exponentially in the size of I_n*

- *Therefore, no matter how* $\mathbf{cert}(Q, I_n)$ *is chosen, its size grows exponentially in the size of I_n*

Thank you!

Computing $\mathbf{cert}(Q, I)$ for queries with inequalities

Let Q be a union of conjunctive queries with inequalities

Computing $\mathbf{cert}(Q, I)$ for queries with inequalities

Let Q be a union of conjunctive queries with inequalities

Proposition

For every incomplete database I, there exists $\mathcal{D}_I \subseteq \{Q(D) \mid D \in [\![I]\!]\}$ such that $\mathcal{D}_I$ is finite and $\mathbf{cert}(Q, I)$ is a greatest lower bound of $\mathcal{D}_I$

Computing $\mathbf{cert}(Q, I)$ for queries with inequalities

Let Q be a union of conjunctive queries with inequalities

Proposition

For every incomplete database I, there exists $\mathcal{D}_I \subseteq \{Q(D) \mid D \in [\![I]\!]\}$ such that $\mathcal{D}_I$ is finite and $\mathbf{cert}(Q, I)$ is a greatest lower bound of $\mathcal{D}_I$

Proof idea: For every null n fix a constant c_n

We only need to consider representations of I where the nulls n occurring in I are replaced by c_n (plus two extra fixed constants)

- No new tuples are added

Computing the greatest lower bound of $\mathcal{D}_I$

glb($\mathcal{D}_I$) can be computed by using a product of instances [HN04,L11,tCD15]

- Based on a well-known product of graphs

Computing the greatest lower bound of $\mathcal{D}_I$

glb$(\mathcal{D}_I)$ can be computed by using a product of instances [HN04,L11,tCD15]

- Based on a well-known product of graphs

The product $I_1 \times I_2$ is an incomplete database I such that for each relation R:

$$R^I \quad = \quad \{(a_1 \times b_1, \ldots, a_k \times b_k) \mid (a_1, \ldots, a_k) \in R^{I_1} \text{ and } (b_1, \ldots, b_k) \in R^{I_2}\},$$

Computing the greatest lower bound of $\mathcal{D}_I$

$\mathbf{glb}(\mathcal{D}_I)$ can be computed by using a product of instances [HN04,L11,tCD15]

- Based on a well-known product of graphs

The product $I_1 \times I_2$ is an incomplete database I such that for each relation R:

$$R^I \quad = \quad \{(a_1 \times b_1, \ldots, a_k \times b_k) \mid (a_1, \ldots, a_k) \in R^{I_1} \text{ and } (b_1, \ldots, b_k) \in R^{I_2}\},$$

where for every $a, b \in (\mathbf{Const} \cup \mathbf{Null})$:

$$a \times b \quad = \quad \begin{cases} a & \text{if } a = b \\ n_{a,b} & \text{if } a \neq b \text{, where } n_{a,b} \text{ is a null} \end{cases}$$

Two examples of the product

For one of the examples presented before we have that:

$$\boxed{b \mid a} \quad \times \quad \boxed{a \mid b} \quad = \quad \boxed{n_{b,a} \mid n_{a,b}}$$

Two examples of the product

For one of the examples presented before we have that:

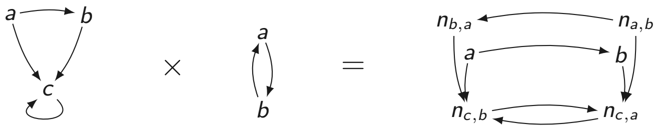
$$\boxed{b \mid a} \times \boxed{a \mid b} = \boxed{n_{b,a} \mid n_{a,b}} = \boxed{\perp_1 \mid \perp_2}$$

Two examples of the product

For one of the examples presented before we have that:

$$\begin{array}{|c|c|}\hline b & a \\ \hline\end{array} \times \begin{array}{|c|c|}\hline a & b \\ \hline\end{array} = \begin{array}{|c|c|}\hline n_{b,a} & n_{a,b} \\ \hline\end{array} = \begin{array}{|c|c|}\hline \perp_1 & \perp_2 \\ \hline\end{array}$$

A more involved example:



Computing $\mathbf{cert}(Q, I)$ for queries with inequalities (cont'd)

Let Q be a union of conjunctive queries with inequalities

Proposition

For every incomplete database I, it holds that $\mathbf{cert}(Q, I)$ can be computed as $\prod \mathcal{D}_I$

How good is the solution?

$\mathcal{D}_I$ can contain an exponential number of complete databases

- This number is exponential in the size of I (the query Q is assumed to be fixed)

How good is the solution?

$\mathcal{D}_I$ can contain an exponential number of complete databases

- This number is exponential in the size of I (the query Q is assumed to be fixed)

Hence, $\prod \mathcal{D}_I$ can be of double-exponential size in the size of I

Bibliography

[HN04] Pavol Hell and Jaroslav Nesetril. *Graphs and Homomorphisms*. Oxford University Press, 2004.

[L11] Leonid Libkin. *Incomplete information and certain answers in general data models*. In PODS 2011.

[L16] Leonid Libkin. *Certain answers as objects and knowledge*. Artificial Intelligence 232:1?19, 2016.

[tCD15] Balder ten Cate and V Dalmau. *The product homomorphism problem and applications*. In ICDT 2015.