Certain Query Answering in Partially Consistent Databases

Sergio Greco Fabian Pijcke Jef Wijsen

Accepted at VLDB 2014

November 29, 2013

Definition (Uncertain database)

A database in which primary keys need not be satified.

Definition (Repair)

A maximal subset of tuples that satisfy primary keys.

Example

R	First	Last	Birth	Sal	City	Country
	Ed	Smith	1960	50K	Acri	Italy
	Ed	Smith	1960	60K	Acri	Italy
	An	Allen	1970	40K	Mons	Belgium

S	City	Country	Stars
	Acri	Italy	**
	Acri	Italy	***
	Mons	Belgium	***
	Mons	France	***

Definition (Uncertain database)

A database in which primary keys need not be satified.

Definition (Repair)

A maximal subset of tuples that satisfy primary keys.

Example

R

First	Last	Birth	Sal	City	Country
Ed	Smith	1960	50K	Acri	Italy
Ed	Smith	1960	60K	Acri	Italy
An	Allen	1970	40K	Mons	Belgium

S

City	Country	Stars
Acri	Italy	**
Acri	Italy	***
Mons	Belgium	***
Mons	France	***

Definition (Uncertain database)

A database in which primary keys need not be satified.

Definition (Repair)

A maximal subset of tuples that satisfy primary keys.

Example

R

First	Last	Birth	Sal	City	Country
Ed	Smith	1960	50K	Acri	Italy
Ed	Smith	1960	60K	Acri	Italy
An	Allen	1970	40K	Mons	Belgium

S

City	Country	Stars
Acri	Italy	**
Acri	Italy	***
Mons	Belgium	***
Mons	France	***

Definition (Uncertain database)

A database in which primary keys need not be satified.

Definition (Repair)

A maximal subset of tuples that satisfy primary keys.

Example

R

First	Last	Birth	Sal	City	Country
Ed	Smith	1960	60K	Acri	Italy
An	Allen	1970	40K	Mons	Belgium

S

City	Country	Stars
Acri	Italy	***
Mons	Belgium	***

Definition (The problem *CERTAINTY*(q))

For a fixed boolean query q (parameter of the problem), the problem *CERTAINTY*(q) is the following:

Input: Uncertain database **db**.

Question: Is q true in every repair of **db**?

Complexity classification problem

Input: Acyclic Boolean conjunctive query q without self-join.

Question: Determine the complexity class of *CERTAINTY*(q), in particular:

- Is *CERTAINTY*(q) first-order definable, and hence solvable in SQL?
- Is *CERTAINTY*(q) in P?
- Is *CERTAINTY*(q) coNP-complete?

Definition (The problem *CERTAINTY*(q))

For a fixed boolean query q (parameter of the problem), the problem *CERTAINTY*(q) is the following:

Input: Uncertain database **db**.

Question: Is q true in every repair of **db**?

Complexity classification problem

Input: Acyclic Boolean conjunctive query q without self-join.

Question: Determine the complexity class of *CERTAINTY*(q), in particular:

- Is *CERTAINTY*(q) first-order definable, and hence solvable in SQL?
- Is *CERTAINTY*(q) in P?
- Is *CERTAINTY*(q) coNP-complete?

Example

R	First	Last	Birth	Sal	City	Country
	Ed	Smith	1960	50K	Acri	Italy
	Ed	Smith	1960	60K	Acri	Italy
	An	Allen	1970	40K	Mons	Belgium

S	City	Country	Stars
	Acri	Italy	**
	Acri	Italy	***
	Mons	Belgium	***
	Mons	France	***

$q_1 = \exists x \exists y\ \mathrm{S}(\underline{x}, \mathrm{Belgium}, y)$ $\Rightarrow$ The answer to $\mathit{CERTAINTY}(q_1)$ is "no"

$q_2 = \exists x \exists y \exists z\ \mathrm{R}(\underline{\mathrm{Ed}, \mathrm{Smith}}, 1960, x, \mathrm{Acri}, z) \wedge \mathrm{S}(\underline{\mathrm{Acri}}, z, y)$

$\Rightarrow$ The answer to $\mathit{CERTAINTY}(q_2)$ is "yes"

Example

R

First	Last	Birth	Sal	City	Country
Ed	Smith	1960	50K	Acri	Italy
Ed	Smith	1960	60K	Acri	Italy
An	Allen	1970	40K	Mons	Belgium

S

City	Country	Stars
Acri	Italy	**
Acri	Italy	***
Mons	Belgium	***
Mons	France	***

$q_1 = \exists x \exists y\ \mathsf{S}(\underline{x}, \mathsf{Belgium}, y) \Rightarrow$ The answer to *CERTAINTY*(q_1) is "no"

$q_2 = \exists x \exists y \exists z\ \mathsf{R}(\underline{\mathsf{Ed}, \mathsf{Smith}}, 1960, x, \mathsf{Acri}, z) \wedge \mathsf{S}(\underline{\mathsf{Acri}}, z, y)$

$\Rightarrow$ The answer to *CERTAINTY*(q_2) is "yes"

Example

R

First	Last	Birth	Sal	City	Country
Ed	Smith	1960	50K	Acri	Italy
An	Allen	1970	40K	Mons	Belgium

S

City	Country	Stars
Acri	Italy	**
Mons	France	***

$q_1 = \exists x \exists y\ \mathsf{S}(\underline{x}, \mathsf{Belgium}, y) \Rightarrow$ The answer to *CERTAINTY*(q_1) is "no"

$q_2 = \exists x \exists y \exists z\ \mathsf{R}(\underline{\mathsf{Ed}, \mathsf{Smith}}, 1960, x, \mathsf{Acri}, z) \wedge \mathsf{S}(\underline{\mathsf{Acri}}, z, y)$

$\Rightarrow$ The answer to *CERTAINTY*(q_2) is "yes"

Example

R	First	Last	Birth	Sal	City	Country
	Ed	Smith	1960	50K	Acri	Italy
	Ed	Smith	1960	60K	Acri	Italy
	An	Allen	1970	40K	Mons	Belgium

S	City	Country	Stars
	Acri	Italy	**
	Acri	Italy	***
	Mons	Belgium	***
	Mons	France	***

$q_1 = \exists x \exists y\ \mathsf{S}(\underline{x}, \mathsf{Belgium}, y) \Rightarrow$ The answer to *CERTAINTY*(q_1) is "no"

$q_2 = \exists x \exists y \exists z\ \mathsf{R}(\underline{\mathsf{Ed}, \mathsf{Smith}}, 1960, x, \mathsf{Acri}, z) \wedge \mathsf{S}(\underline{\mathsf{Acri}}, z, y)$

$\Rightarrow$ The answer to *CERTAINTY*(q_2) is "yes"

Definition (First-order definable)

CERTAINTY(q) is first-order definable if there exists a first-order sentence ϕ such that for every uncertain database **db**, the following are equivalent:

- q is true in every repair of **db**; and
- ϕ is true in **db**.

We call ϕ, if it exists, a consistent FO rewriting.

Example

$$q_3 = \exists x \exists y \; \mathsf{S}(\underline{x}, \mathsf{Italy}, y)$$

$$\phi_3 = \exists x \exists y \, (\; \mathsf{S}(\underline{x}, \mathsf{Italy}, y) \wedge (\forall z_1 \forall z_2 \; \mathsf{S}(\underline{x}, z_1, z_2) \rightarrow z_1 = \mathsf{Italy}))$$

Definition (First-order definable)

CERTAINTY(q) is first-order definable if there exists a first-order sentence ϕ such that for every uncertain database **db**, the following are equivalent:

- q is true in every repair of **db**; and
- ϕ is true in **db**.

We call ϕ, if it exists, a consistent FO rewriting.

Example

$$q_3 = \exists x \exists y \; \mathsf{S}(\underline{x}, \mathsf{Italy}, y)$$

$$\phi_3 = \exists x \exists y \, (\; \mathsf{S}(\underline{x}, \mathsf{Italy}, y) \wedge (\forall z_1 \forall z_2 \; \mathsf{S}(\underline{x}, z_1, z_2) \rightarrow z_1 = \mathsf{Italy}))$$

Definition (The problem *CERTAINTY*(q, Σ))

- fixed boolean query q
- fixed set Σ of constraints:
  - functional dependencies
  - at most one key-join dependency per relation
- Input: uncertain database **db** satisfying every constraint in Σ
- Question: is q true in every repair of **db** ?

Theorem

Given Σ (as above) and an acyclic boolean conjunctive query q without self-join, it is decidable whether CERTAINTY(q, Σ) is first-order definable. Furthermore, we can construct a consistent FO rewriting if it exists.

Definition (The problem $CERTAINTY(q, \Sigma)$)

- fixed boolean query q
- fixed set Σ of constraints:
  - functional dependencies
  - at most one key-join dependency per relation
- Input: uncertain database **db** satisfying every constraint in Σ
- Question: is q true in every repair of **db** ?

Theorem

Given Σ (as above) and an acyclic boolean conjunctive query q without self-join, it is decidable whether $CERTAINTY(q, \Sigma)$ is first-order definable. Furthermore, we can construct a consistent FO rewriting if it exists.

Definition (The problem $CERTAINTY(q, \Sigma)$)

- fixed boolean query q
- fixed set Σ of constraints:
  - functional dependencies
  - at most one key-join dependency per relation
- Input: uncertain database **db** satisfying every constraint in Σ
- Question: is q true in every repair of **db** ?

Theorem

Given Σ (as above) and an acyclic boolean conjunctive query q without self-join, it is decidable whether $CERTAINTY(q, \Sigma)$ is first-order definable. Furthermore, we can construct a consistent FO rewriting if it exists.

Example

$$q_4 = \exists x \exists y \; \mathsf{S}(\underline{x}, \mathsf{Italy}, y), \Sigma_4 = \{S : \mathsf{City}_1 \rightarrow \mathsf{Country}_2\}$$

$$\phi_4 = \exists x \exists y \; (\; \mathsf{S}(\underline{x}, \mathsf{Italy}, y) \wedge (\forall z_1 \forall z_2 \; \mathsf{S}(\underline{x}, z_1, z_2) \rightarrow z_1 = \mathsf{Italy}) \;)$$

Example

$$q_4 = \exists x \exists y \; \mathsf{S}(\underline{x}, \mathsf{Italy}, y), \Sigma_4 = \{S : \mathsf{City}_1 \rightarrow \mathsf{Country}_2\}$$

$$\phi_4 = \exists x \exists y \; (\; \mathsf{S}(\underline{x}, \mathsf{Italy}, y) \wedge (\forall z_1 \forall z_2 \; \mathsf{S}(\underline{x}, z_1, z_2) \rightarrow z_1 = \mathsf{Italy}) \;)$$

Example

$$q_5 = \exists u \exists v \exists w \exists x \exists y \exists z \; \mathsf{R}(\underline{u, v}, w, x, y, z) \wedge \mathsf{S}(\underline{y}, z, ***)$$

There exists no consistent FO rewriting for q_5.

Example

$$q_6 = \exists u \exists v \exists w \exists x \exists y \exists z \; \mathsf{R}(\underline{u, v}, w, x, y, z) \wedge \mathsf{S}(\underline{y}, z, ***)$$

$$\Sigma_6 = \{R : \mathsf{City}_5 \rightarrow \mathsf{Country}_6\}$$

$$\begin{aligned} \phi \;=\; & \exists u \exists v \exists w \exists x \exists y \exists z \Big(\mathsf{R}(\underline{u, v}, v, x, y, z) \wedge \Big(\\ & \forall w \forall x \forall y \forall z \big(\mathsf{R}(\underline{u, v}, w, x, y, z) \rightarrow \mathsf{S}(\underline{y}, z, ***) \wedge (\\ & \forall z_1 \forall z_2 \; \mathsf{S}(\underline{y}, z_1, z_2) \rightarrow z_1 = z \wedge z_2 = ***) \big) \Big) \Big) \end{aligned}$$

Example

$$q_5 = \exists u \exists v \exists w \exists x \exists y \exists z \; \mathrm{R}(\underline{u, v}, w, x, y, z) \wedge \mathrm{S}(\underline{y}, z, ***)$$

There exists no consistent FO rewriting for q_5.

Example

$$q_6 = \exists u \exists v \exists w \exists x \exists y \exists z \; \mathrm{R}(\underline{u, v}, w, x, y, z) \wedge \mathrm{S}(\underline{y}, z, ***)$$

$$\Sigma_6 = \{R : \mathrm{City}_5 \rightarrow \mathrm{Country}_6\}$$

$$\begin{aligned} \phi \;=\; & \exists u \exists v \exists w \exists x \exists y \exists z \Big(\mathrm{R}(\underline{u, v}, v, x, y, z) \wedge \Big(\\ & \forall w \forall x \forall y \forall z \big(\mathrm{R}(\underline{u, v}, w, x, y, z) \rightarrow \mathrm{S}(\underline{y}, z, ***) \wedge \big(\\ & \forall z_1 \forall z_2 \; \mathrm{S}(\underline{y}, z_1, z_2) \rightarrow z_1 = z \wedge z_2 = ***) \big) \big) \Big) \end{aligned}$$

Example

$$q_5 = \exists u \exists v \exists w \exists x \exists y \exists z \; \mathrm{R}(\underline{u, v}, w, x, y, z) \wedge \mathrm{S}(\underline{y}, z, ***)$$

There exists no consistent FO rewriting for q_5.

Example

$$q_6 = \exists u \exists v \exists w \exists x \exists y \exists z \; \mathrm{R}(\underline{u, v}, w, x, y, z) \wedge \mathrm{S}(\underline{y}, z, ***)$$

$$\Sigma_6 = \{R : \mathrm{City}_5 \rightarrow \mathrm{Country}_6\}$$

$$\begin{aligned} \phi \;=\; & \exists u \exists v \exists w \exists x \exists y \exists z \Big(\mathrm{R}(\underline{u, v}, v, x, y, z) \wedge \Big(\\ & \forall w \forall x \forall y \forall z \big(\mathrm{R}(\underline{u, v}, w, x, y, z) \rightarrow \mathrm{S}(\underline{y}, z, ***) \wedge \big(\\ & \forall z_1 \forall z_2 \; \mathrm{S}(\underline{y}, z_1, z_2) \rightarrow z_1 = z \wedge z_2 = ***) \big) \big) \Big) \end{aligned}$$

Example

$$q_5 = \exists u \exists v \exists w \exists x \exists y \exists z \; \mathrm{R}(\underline{u, v}, w, x, y, z) \wedge \mathrm{S}(\underline{y}, z, ***)$$

There exists no consistent FO rewriting for q_5.

Example

$$q_6 = \exists u \exists v \exists w \exists x \exists y \exists z \; \mathrm{R}(\underline{u, v}, w, x, y, z) \wedge \mathrm{S}(\underline{y}, z, ***)$$

$$\Sigma_6 = \{R : \mathrm{City}_5 \rightarrow \mathrm{Country}_6\}$$

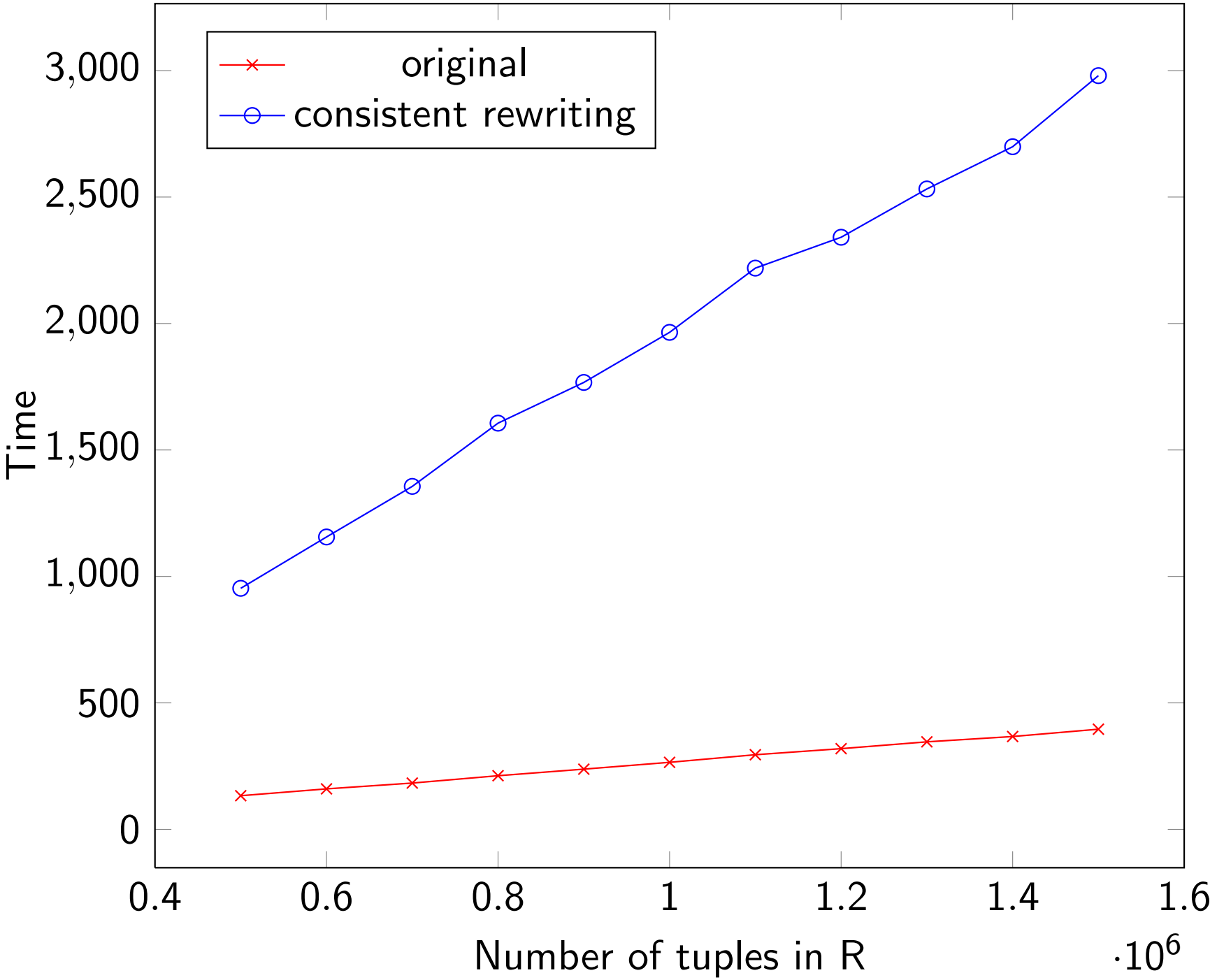
$$\begin{aligned} \phi \quad = \quad & \exists u \exists v \exists w \exists x \exists y \exists z \Big(\mathrm{R}(\underline{u, v}, v, x, y, z) \wedge \Big(\\ & \forall w \forall x \forall y \forall z \Big(\mathrm{R}(\underline{u, v}, w, x, y, z) \rightarrow \mathrm{S}(\underline{y}, z, ***) \wedge \Big(\\ & \forall z_1 \forall z_2 \; \mathrm{S}(\underline{y}, z_1, z_2) \rightarrow z_1 = z \wedge z_2 = *** \Big) \Big) \Big) \Big) \end{aligned}$$

Original

```
SELECT R.Sal FROM R, S
WHERE R.City = S.City AND R.Country = S.Country AND S.Stars = '***';
```

Consistent FO rewriting, encoded in SQL

```
SELECT R1.Sal FROM R AS R1
WHERE NOT EXISTS (
   SELECT * FROM R AS R2
   WHERE R2.First = R1.First AND R2.Last = R1.Last
     AND (R2.Sal <> R1.Sal OR NOT EXISTS (
      SELECT * FROM S AS S1
      WHERE S1.City = R2.City AND S1.Country = R2.Country
        AND S1.Stars = '***'
        AND NOT EXISTS (
         SELECT * FROM S AS S2
         WHERE S2.City = S1.City
           AND (S2.Country <> S1.Country OR S2.Stars <> '***')))));
```


original
consistent rewriting
3,000
2,500
2,000
1,500
1,000
500
0
Time
0.4
0.6
0.8
1
1.2
1.4
1.6
Number of tuples in R
$\cdot 10^6$