Towards a Foundation of Data Currency

Jef Wijsen

University of Mons

Joint work with Wenfei Fan and Floris Geerts, University of Edinburgh

TIME Symposium, 2011

Background/Motivation

- "The problem with data is that its quality quickly degenerates over time. Experts say 2 percent of records in a customer file become obsolete in one month because customers die, divorce, marry, and move." [Eck02]
- Often no reliable timestamps.
- How can we decide the "current truth" of a Boolean query?

Outline

Caveat

- Simplified version of the currency model proposed at PODS 2011 [FGW11].
- But the simplification maintains lower and upper complexity bounds of the problems presented.

Outline

Ordered Relation

Ordered relation $(I, \prec)$

Relation I equipped with a currency order $\prec$ such that:

- distinct tuples that agree on the primary key are comparable under $\prec$; and
- tuples that disagree on the primary key are incomparable.

Example

EMP	<u>*FN*</u>	<u>*LN*</u>	*Addr*	*Sal*	*Stat*
t_1:	Mary	Smith	2 Small St	50k	single
t_2:	Mary	Smith	10 Elm Ave	80k	married
t_3:	Mary	Smith	6 Main St	50k	married
t_4:	Bob	Luth	8 Cowan St	55k	married
t_5:	Bob	Luth	8 Drum St	80k	married

$t_1 \prec t_3 \prec t_2$

$t_4 \prec t_5$

Ordered Relation

Ordered relation $(I, \prec)$

Relation I equipped with a currency order $\prec$ such that:

- distinct tuples that agree on the primary key are comparable under $\prec$; and
- tuples that disagree on the primary key are incomparable.

Example

EMP	<u>*FN*</u>	<u>*LN*</u>	*Addr*	*Sal*	*Stat*
t_1:	Mary	Smith	2 Small St	50k	single
t_2:	Mary	Smith	10 Elm Ave	80k	married
t_3:	Mary	Smith	6 Main St	50k	married
t_4:	Bob	Luth	8 Cowan St	55k	married
t_5:	Bob	Luth	8 Drum St	80k	married

$t_1 \prec t_3 \prec t_2$

$t_4 \prec t_5$

Ordered Relation

Ordered relation $(I, \prec)$

Relation I equipped with a currency order $\prec$ such that:

- distinct tuples that agree on the primary key are comparable under $\prec$; and
- tuples that disagree on the primary key are incomparable.

Example

EMP	*FN*	*LN*	*Addr*	*Sal*	*Stat*
t_1:	Mary	Smith	2 Small St	50k	single
t_2:	Mary	Smith	10 Elm Ave	80k	married
t_3:	Mary	Smith	6 Main St	50k	married
t_4:	Bob	Luth	8 Cowan St	55k	married
t_5:	Bob	Luth	8 Drum St	80k	married

$t_1 \prec t_3 \prec t_2$

$t_4 \prec t_5$

Ordered Relation

Ordered relation $(I, \prec)$

Relation I equipped with a currency order $\prec$ such that:

- distinct tuples that agree on the primary key are comparable under $\prec$; and
- tuples that disagree on the primary key are incomparable.

Example

EMP	*FN*	*LN*	*Addr*	*Sal*	*Stat*
t_1:	Mary	Smith	2 Small St	50k	single
t_2:	Mary	Smith	10 Elm Ave	80k	married
t_3:	Mary	Smith	6 Main St	50k	married
t_4:	Bob	Luth	8 Cowan St	55k	married
t_5:	Bob	Luth	8 Drum St	80k	married

$t_1 \prec t_3 \prec t_2$

$t_4 \prec t_5$

There is a unique present.

Present

Present [relation]

Relation obtained by selecting all greatest tuples.

Example

EMP

FN	*LN*	*Addr*	*Sal*	*Stat*
Mary	Smith	10 Elm Ave	80k	married
Bob	Luth	8 Drum St	80k	married

Unordered Relation

Unordered relation

A relation in which primary keys need not be satisfied.
No currency order.

Example

EMP	*FN*	*LN*	*Addr*	*Sal*	*Stat*
t_1:	Mary	Smith	2 Small St	50k	single
t_2:	Mary	Smith	10 Elm Ave	80k	married
t_3:	Mary	Smith	6 Main St	50k	married
t_4:	Bob	Luth	8 Cowan St	55k	married
t_5:	Bob	Luth	8 Drum St	80k	married

Unordered Relation

Unordered relation

A relation in which primary keys need not be satisfied.
No currency order.

Example

EMP	<u>*FN*</u>	<u>*LN*</u>	*Addr*	*Sal*	*Stat*
t_1:	Mary	Smith	2 Small St	50k	single
t_2:	Mary	Smith	10 Elm Ave	80k	married
t_3:	Mary	Smith	6 Main St	50k	married
t_4:	Bob	Luth	8 Cowan St	55k	married
t_5:	Bob	Luth	8 Drum St	80k	married

This leaves six possible presents.

[Currently] Certain

Completion (of an unordered relation)

Ordered relation obtained by adding a currency order $\prec$ to an unordered relation.
Every completion gives a possible present.

Definition

A Boolean query is [currently] certain if it is true in each possible present.

Certainty

It is certain that *"Bob is married."*

Certainty

The problem of certainty

For any fixed Boolean first-order query q, CERTAINTY(q) is the following problem:

Input An unordered relation

Question Is q [currently] certain?

Certainty

The problem of certainty

For any fixed Boolean first-order query q, CERTAINTY(q) is the following problem:

Input An unordered relation

Question Is q [currently] certain?

Data complexity

- CERTAINTY(q) is in **coNP**.
- There exists a Boolean conjunctive query q such that CERTAINTY(q) is **coNP**-complete.

Outline

1. Data Currency Model
   - Basic Model
   - Extension: Currency Constraints
   - Extension: Copying
2. More on CERTAINTY(q)
   - Deciding FO Definability of CERTAINTY(q)
   - Technical Development
   - Discussion and Extensions

Currency Constraints

Definition

Currency constraints are denial constraints that restrict the set of legal completions (and hence restrict the set of possible presents).

Currency constraints

- Salaries don't decrease.

$$\neg\exists s\exists t\big(s.FN = t.FN \wedge s.LN = t.LN \wedge \\ s.Sal < t.Sal \wedge t \prec s\big)$$

- Mary never divorced.

$$\neg\exists s\exists t\big(s.FN = t.FN = \text{Mary} \wedge s.LN = t.LN = \text{Smith} \wedge \\ s.Stat = \text{single} \wedge t.Stat = \text{married} \wedge t \prec s\big)$$

Consistent

Consistency w.r.t. a set Σ of denials

We call an unordered relation I consistent if it has a consistent completion (i.e., if $(I, \prec) \models \Sigma$ for some currency order $\prec$ on I).

Inconsistency

EMP	$\underline{FN}$	$\underline{LN}$	$Addr$	Sal	$Stat$
s_1:	Mary	Smith	2 Small St	80k	single
s_2:	Mary	Smith	10 Elm Ave	50k	married

$$\neg\exists s \exists t \big(s.FN = t.FN \wedge s.LN = t.LN \wedge \\ s.Sal < t.Sal \wedge t \prec s \big)$$

$$\neg\exists s \exists t \big(s.FN = t.FN = \text{Mary} \wedge s.LN = t.LN = \text{Smith} \wedge \\ s.Stat = \text{single} \wedge t.Stat = \text{married} \wedge t \prec s \big)$$

Consistency

The problem of consistency

For any fixed set Σ of currency constraints, CONSISTENCY(Σ) is the following problem:

Input An unordered relation I

Question Is I consistent?

Consistency

The problem of consistency

For any fixed set Σ of currency constraints, CONSISTENCY(Σ) is the following problem:

Input An unordered relation I

Question Is I consistent?

Data complexity

- CONSISTENCY(Σ) is in **NP**.
- There exists a set Σ of currency constraints such that CONSISTENCY(Σ) is **NP**-complete.

Outline

1. Data Currency Model
   - Basic Model
   - Extension: Currency Constraints
   - Extension: Copying

2. More on CERTAINTY(q)
   - Deciding FO Definability of CERTAINTY(q)
   - Technical Development
   - Discussion and Extensions

Background/Motivation

Deciding the "current truth" (of a Boolean query) in:

Data integration: Conflicting facts are provided by a large number of sources [DBES09].

Data replication: Applications use out-of-date replicas to improve scalability, availability, and performance [GLR05].

Copying

Example

CACHE	*FN*	*LN*	*Addr*	*Sal*	*Stat*
s_1:	Mary	Smith	2 Small St	50k	single
s_2:	Mary	Smith	10 Elm Ave	70k	married

EMP	*FN*	*LN*	*Addr*	*Sal*	*Stat*
t_1:	Mary	Smith	2 Small St	60k	single
t_2:	Mary	Smith	6 Main St	60k	married

The cache is a finite set of unordered tuples.

Copying

Example

CACHE	*FN*	*LN*	*Addr*	*Sal*	*Stat*
s_1:	Mary	Smith	2 Small St	50k	single
s_2:	Mary	Smith	10 Elm Ave	70k	married

EMP	*FN*	*LN*	*Addr*	*Sal*	*Stat*
t_1:	Mary	Smith	2 Small St	60k	single
t_2:	Mary	Smith	6 Main St	60k	married

The cache is a finite set of unordered tuples.

Should we extend *EMP* with extra tuples from the cache in order to get the "current truth" of a Boolean query?

Compare:

- "Is Mary married?"
- "Does Mary earn 60k?"

Copying Problems

Caveat

Extending with all “cache” tuples may result in inconsistency.

Copying Problems

Caveat

Extending with all "cache" tuples may result in inconsistency.

Consistent extension w.r.t. a set Σ of denials

An unordered relation can be extended with "cache" tuples.
We call such extension maximal consistent if

- it is consistent; and
- any further proper extension is inconsistent.

Copying Problems

Caveat

Extending with all "cache" tuples may result in inconsistency.

Consistent extension w.r.t. a set Σ of denials

An unordered relation can be extended with "cache" tuples.
We call such extension maximal consistent if

- it is consistent; and
- any further proper extension is inconsistent.

Don't confuse:

- Completion $\rightsquigarrow$ adding currency order $\prec$
- Extension $\rightsquigarrow$ importing "cache" tuples

Certainty Preservation

The problem of certainty preservation

For any fixed Boolean first-order query q and set Σ of currency constraints, $\mathsf{CPP}(q, \Sigma)$ is the following problem:

Input
- a consistent unordered relation in which q is [currently] certain
- a "cache"

Question Is q [currently] certain in at least one maximal consistent extension?

Certainty Preservation

The problem of certainty preservation

For any fixed Boolean first-order query q and set Σ of currency constraints, CPP(q, Σ) is the following problem:

Input
- a consistent unordered relation in which q is [currently] certain
- a "cache"

Question Is q [currently] certain in at least one maximal consistent extension?

Data complexity

- CPP(q, Σ) is in Σ_2^p.
- There exists a Boolean conjunctive query q and set Σ such that CPP(q, Σ) is Σ_2^p-complete.

Σ_2^p Algorithm

Outline

1. Guess an ordered relation $(I, \prec)$.
2. Verify I is an extension and $(I, \prec) \models \Sigma$ (in **P**).
3. Verify maximality, i.e., adding an extra cache tuple to I results in inconsistency (at most linearly many queries of **coNP** oracle).
4. Verify q is [currently] certain in I (one query of **coNP** oracle).

Hardness is by reduction from the complement of $\forall\exists 3CNF$.

Outline

Back to the Basic Model/Motivation

- No currency constraints.
- No copying.
- What you get if you strip off timestamps from a temporal relation.
- We have seen that even in this basic model, there exists a Boolean conjunctive query q such that CERTAINTY(q) is **coNP**-hard.
- Can we identify queries q for which CERTAINTY(q) is tractable (or, even better, first-order definable)?

Unordered Database Again

Unordered database

A database in which primary keys need not be satisfied.

Possible present

In the absence of currency constraints, a possible present is any maximal subset of tuples that satisfy primary keys.

Unordered Database Again

Unordered database

A database in which primary keys need not be satisfied.

Possible present

In the absence of currency constraints, a possible present is any maximal subset of tuples that satisfy primary keys.

Every tuple was once true

R	*FN*	*LN*	*Dname*
	Mary	Smith	R&D
	Mary	Smith	MIS

T	*Dname*	*Budget*
	R&D	60K
	MIS	60K
	Toys	60K
	Toys	70K

$\rightsquigarrow$ 2×2 possible presents

Certainty

Definition

A Boolean query is certain if it is true in each possible present.

Certainty

Definition

A Boolean query is certain if it is true in each possible present.

Example

R	<u>*FN*</u>	<u>*LN*</u>	*Dname*
	Mary	Smith	R&D
	Mary	Smith	MIS

T	<u>*Dname*</u>	*Budget*
	R&D	60K
	MIS	60K
	Toys	60K
	Toys	70K

$$
\begin{aligned}
q_1 &= \exists x(R(\underline{\text{Mary}, x}, \text{R\&D})) \\
q_2 &= \exists x \exists y(R(\underline{\text{Mary}, x}, y) \wedge T(\underline{y}, \text{60K}))
\end{aligned}
$$

$\rightsquigarrow$ q_1 is not certain, q_2 is certain

Problem Statement

CERTAINTY(q)

For fixed Boolean query q, the problem CERTAINTY(q) is:

Given an unordered database, is q certain?

Problem Statement

CERTAINTY(q)

For fixed Boolean query q, the problem CERTAINTY(q) is:

Given an unordered database, is q certain?

Complexity for conjunctive queries

- For $q_3 = \exists x \exists y \exists z (S(\underline{x}, z) \wedge T(\underline{y}, z))$,
  CERTAINTY(q_3) is **coNP**-hard.

Problem Statement

CERTAINTY(q)

For fixed Boolean query q, the problem CERTAINTY(q) is:

Given an unordered database, is q certain?

Complexity for conjunctive queries

- For $q_3 = \exists x \exists y \exists z (S(\underline{x}, z) \wedge T(\underline{y}, z))$,
  CERTAINTY(q_3) is **coNP**-hard.
- CERTAINTY(q_1) is first-order expressible (see later). That is, "Is q_1 certain?" can be encoded in FO.

Problem Statement

CERTAINTY(q)

For fixed Boolean query q, the problem CERTAINTY(q) is:
Given an unordered database, is q certain?

Complexity for conjunctive queries

- For $q_3 = \exists x \exists y \exists z (S(\underline{x}, z) \land T(\underline{y}, z))$,
CERTAINTY(q_3) is **coNP**-hard.
- CERTAINTY(q_1) is first-order expressible (see later). That is, "Is q_1 certain?" can be encoded in FO.

Research problem

Find algorithm for the following decision problem:
Given Boolean conjunctive query q,
is CERTAINTY(q) first-order expressible (and hence in $\mathbf{AC}^0$)?

State of the art

[Wij10a]

An algorithm for the following decision problem:

Given Boolean acyclic conjunctive query q without self-join, is CERTAINTY(q) first-order expressible?

If answer is "yes," we can construct the first-order expression.

State of the art

[Wij10a]

An algorithm for the following decision problem:

Given Boolean acyclic conjunctive query q without self-join, is CERTAINTY(q) first-order expressible?

If answer is "yes," we can construct the first-order expression.

Remaining restrictions

- q without self-join $\iff$ no duplicate relation names in q
- q acyclic $\iff$ q has a join tree

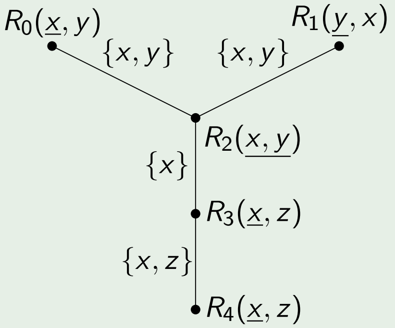
Join Tree

Definition

A join tree for conjunctive query q is an undirected tree whose vertices are the atoms of q, such that:

> Connectedness Cond. if the same variable x occurs in two distinct atoms F and G, then x occurs in each vertex on the (unique) path linking F and G.

Example



Outline

1. Data Currency Model
   - Basic Model
   - Extension: Currency Constraints
   - Extension: Copying
2. More on CERTAINTY(q)
   - Deciding FO Definability of CERTAINTY(q)
   - Technical Development
   - Discussion and Extensions

First-order Expressibility

CERTAINTY(q_1) is first-order expressible

$$q_1 = \exists x(R(\underline{\mathsf{Mary}, x}, \mathsf{R\&D}))$$

$$\varphi_1 = \exists x\big[\, R(\underline{\mathsf{Mary}, x}, \mathsf{R\&D}) \wedge \forall z \big(\, R(\underline{\mathsf{Mary}, x}, z) \rightarrow z = \mathsf{R\&D}\big)\big]$$

For every unordered database,
q_1 certain $\iff$ φ_1 true.

First-order Expressibility

CERTAINTY(q_2) is first-order expressible

$$q_2 \quad = \quad \exists x \exists y (R(\underline{\mathrm{Mary}, x}, y) \wedge T(\underline{y}, 60\mathrm{K}))$$

$$\begin{aligned}\varphi_2 = \exists x \exists y \Big[& R(\underline{\mathrm{Mary}, x}, y) \wedge \\ & \forall y \Big(R(\underline{\mathrm{Mary}, x}, y) \rightarrow \big[T(\underline{y}, 60\mathrm{K}) \wedge \\ & \qquad \forall z (T(\underline{y}, z) \rightarrow z = 60\mathrm{K}) \big] \Big) \Big]\end{aligned}$$

For every unordered database,
q_2 certain $\iff$ φ_2 true.

Order is Important

Changing the order

$$q_2 = \exists x \exists y (T(\underline{y}, 60\mathrm{K}) \land R(\underline{\mathrm{Mary}, x}, y))$$

$$\begin{aligned}\varphi_2' = \; & \exists y \Big[T(\underline{y}, 60\mathrm{K}) \land \\ & \forall z \Big(T(\underline{y}, z) \rightarrow z = 60\mathrm{K} \land \\ & \qquad \exists x \big[R(\underline{\mathrm{Mary}, x}, y) \land \\ & \qquad \forall w \big(R(\underline{\mathrm{Mary}, x}, w) \rightarrow w = y \big) \big] \Big) \Big]\end{aligned}$$

For every unordered database, φ_2' true $\implies$ q_2 certain;
but the converse does not hold:

Order is Important

Changing the order

$$q_2 = \exists x \exists y (T(\underline{y}, 60K) \land R(\underline{\text{Mary}, x}, y))$$

$$\begin{aligned}\varphi_2' = \; & \exists y \Big[T(\underline{y}, 60K) \land \\ & \forall z \Big(T(\underline{y}, z) \rightarrow z = 60K \land \\ & \qquad \exists x \big[R(\underline{\text{Mary}, x}, y) \land \\ & \qquad \forall w \big(R(\underline{\text{Mary}, x}, w) \rightarrow w = y \big) \big] \Big) \Big]\end{aligned}$$

For every unordered database, φ_2' true $\Longrightarrow$ q_2 certain;
but the converse does not hold:

T	*Dname*	*Budget*
	R&D	60K
	MIS	60K

R	*FN*	*LN*	*Dname*
	Mary	Smith	R&D
	Mary	Smith	MIS

q_2 certain
φ_2' false

Determining the Order

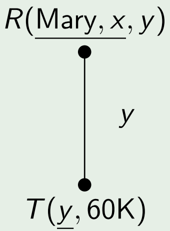
Attack graph

For each Boolean acyclic conjunctive query q, without self-join, we compute a directed graph, called attack graph:

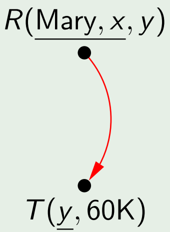
- The vertices are the atoms of q.
- The directed edges, to be defined later on, are such that a first-order expression for CERTAINTY(q) can be obtained by applying our "$\exists\forall$-rewrite function" on any topological sort of the attack graph.

Getting the Idea

Join tree



Attack graph



Functional Dependencies Implied by Primary Keys

Definition

We write vars($\vec{x}$) for the set of variables occurring in $\vec{x}$.

For single atom $F = R(\underline{\vec{x}}, \vec{y})$:

$$\mathcal{K}(F) := \text{vars}(\vec{x}) \rightarrow \text{vars}(\vec{x}\vec{y})$$

Extension to conjunctive query q:

$$\mathcal{K}(q) := \{\mathcal{K}(F) \mid F \text{ atom of } q\}$$

Example

$$\begin{aligned} q_2 &= \exists x \exists y (R(\underline{\text{Mary}, x}, y) \wedge T(\underline{y}, 60\text{K})) \\ \mathcal{K}(q_2) &= \{x \rightarrow xy, y \rightarrow y\} \end{aligned}$$

Attack Graph of a Join Tree

Attack graph

The attack graph of join tree τ contains a directed edge from $R(\underline{\vec{x}}, \vec{y})$ to another atom F if for each label L on the path that links $R(\underline{\vec{x}}, \vec{y})$ and F in τ:

$$\mathcal{K}(q \setminus \{R(\underline{\vec{x}}, \vec{y})\}) \not\models \mathrm{vars}(\vec{x}) \rightarrow L$$

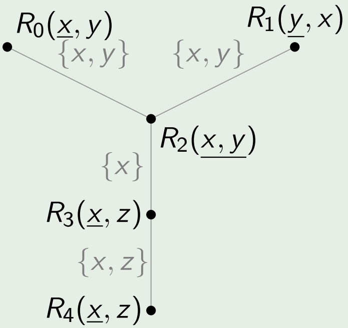
Attack Graph of a Join Tree

Example

Attack graph

The attack graph of join tree τ contains a directed edge from $R(\underline{\vec{x}}, \vec{y})$ to another atom F if for each label L on the path that links $R(\underline{\vec{x}}, \vec{y})$ and F in τ:

$$\mathcal{K}(q \setminus \{R(\underline{\vec{x}}, \vec{y})\}) \not\models \mathrm{vars}(\vec{x}) \rightarrow L$$



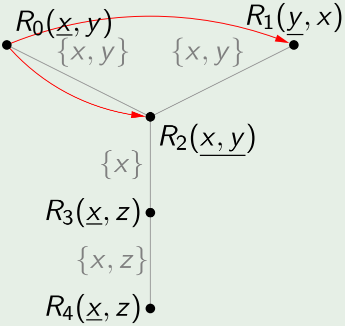
Attack Graph of a Join Tree

Example

Attack graph

The attack graph of join tree τ contains a directed edge from $R(\underline{\vec{x}}, \vec{y})$ to another atom F if for each label L on the path that links $R(\underline{\vec{x}}, \vec{y})$ and F in τ:

$$\mathcal{K}(q \setminus \{R(\underline{\vec{x}}, \vec{y})\}) \not\models \mathrm{vars}(\vec{x}) \rightarrow L$$



$$\mathcal{K}(q \setminus \{R_0(\underline{x}, y)\}) \equiv \{y \rightarrow x, x \rightarrow z\} \models x \rightarrow xz$$

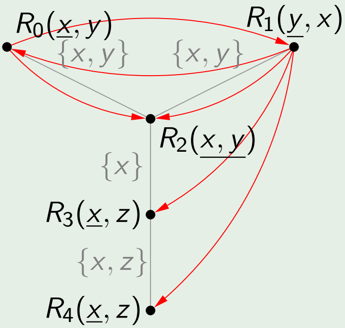
Attack Graph of a Join Tree

Example

Attack graph

The attack graph of join tree τ contains a directed edge from $R(\underline{\vec{x}}, \vec{y})$ to another atom F if for each label L on the path that links $R(\underline{\vec{x}}, \vec{y})$ and F in τ:

$$\mathcal{K}(q \setminus \{R(\underline{\vec{x}}, \vec{y})\}) \not\models \mathrm{vars}(\vec{x}) \rightarrow L$$



$$\mathcal{K}(q \setminus \{R_1(\underline{y}, x)\}) \equiv \{x \rightarrow y, x \rightarrow z\} \models y \rightarrow y$$

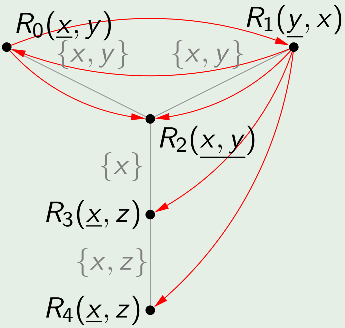
Attack Graph of a Join Tree

Example

Attack graph

The attack graph of join tree τ contains a directed edge from $R(\underline{\vec{x}}, \vec{y})$ to another atom F if for each label L on the path that links $R(\underline{\vec{x}}, \vec{y})$ and F in τ:

$$\mathcal{K}(q \setminus \{R(\underline{\vec{x}}, \vec{y})\}) \not\models \mathrm{vars}(\vec{x}) \rightarrow L$$



$$\mathcal{K}(q \setminus \{R_3(\underline{x}, z)\}) \equiv \{x \rightarrow y, y \rightarrow x, x \rightarrow z\} \models x \rightarrow xyz$$

Main Result

Acyclicity of attack graph is both sufficient and necessary for first-order expressibility.

Theorem

Let q be a Boolean conjunctive query, without self join.
Let τ be a join tree for q.

Main Result

Acyclicity of attack graph is both sufficient and necessary for first-order expressibility.

Theorem

Let q be a Boolean conjunctive query, without self join.
Let τ be a join tree for q.

Sufficient If the attack graph of τ is acyclic, then CERTAINTY(q) is first-order expressible (and its first-order definition can be obtained by applying our "∃∀-rewrite function" on any topological sort of the attack graph).

Main Result

Acyclicity of attack graph is both sufficient and necessary for first-order expressibility.

Theorem

Let q be a Boolean conjunctive query, without self join.
Let τ be a join tree for q.

Sufficient If the attack graph of τ is acyclic, then CERTAINTY(q) is first-order expressible (and its first-order definition can be obtained by applying our "∃∀-rewrite function" on any topological sort of the attack graph).

Necessary If the attack graph of τ is cyclic, then CERTAINTY(q) is not first-order expressible.

Proof of Inexpressibility Result

Lemma

Let q be a Boolean conjunctive query, without self join.
Let τ be a join tree for q.
If the attack graph of τ is cyclic, then it has a cycle of size 2.

Then we can assume two distinct atoms, say F and G, such that the attack graph contains a directed edge from F to G, and a directed edge from G to F.

Two Databases that Locally Look the Same

Outline

1. Data Currency Model
   - Basic Model
   - Extension: Currency Constraints
   - Extension: Copying
2. More on CERTAINTY(q)
   - Deciding FO Definability of CERTAINTY(q)
   - Technical Development
   - Discussion and Extensions

NonBoolean Queries

NonBoolean queries

Our results apply to nonBoolean queries: treat free variables as new constants.

NonBoolean Queries

NonBoolean queries

Our results apply to nonBoolean queries: treat free variables as new constants.

NonBoolean query

$$q_2(v) \;=\; \exists x \exists y (R(\underline{\mathrm{Mary}, x}, y) \wedge T(\underline{y}, v))$$

$$\begin{aligned}\varphi_2(v) = \exists x \exists y \Big[& R(\underline{\mathrm{Mary}, x}, y) \wedge \\ & \forall y \Big(R(\underline{\mathrm{Mary}, x}, y) \rightarrow \big[\, T(\underline{y}, v) \wedge \\ & \qquad \forall z (\, T(\underline{y}, z) \rightarrow z = v)\big]\Big)\Big]\end{aligned}$$

For every unordered database, for every constant a,
$q_2(a)$ certain $\iff$ $\varphi_2(a)$ true.

Self-join

Acyclic conjunctive queries with self-join

$$
\begin{aligned}
q_4 &= \exists x \exists y \exists z(T(\underline{x}, y) \wedge T(\underline{y}, z)) \\
q_5 &= \exists x \exists y(T(\underline{x}, y) \wedge T(\underline{y}, c))
\end{aligned}
$$

We know from earlier work [Wij09]:

- CERTAINTY(q_4) is first-order expressible;
- CERTAINTY(q_5) is not first-order expressible.

Attack graphs cannot distinguish q_4 and q_5.

Cyclic Queries

Cyclic conjunctive query without self-join

Deciding first-order expressibility of CERTAINTY(q) for cyclic q remains open. For example,

$$q_6 = \exists x \exists y \exists z (R_0(\underline{x}, y), R_1(\underline{y}, z), R_2(\underline{z}, x))$$

It is not known whether CERTAINTY(q_6) is tractable.

Tractability of CERTAINTY(q)

For q ranging over the class of Boolean acyclic conjunctive queries without self-join:

- we can decide whether CERTAINTY(q) is first-order expressible (and hence in $\mathbf{AC}^0$);
- but can we decide whether CERTAINTY(q) is in **P**?

Tractability of CERTAINTY(q)

For q ranging over the class of Boolean acyclic conjunctive queries without self-join:

- we can decide whether CERTAINTY(q) is first-order expressible (and hence in $\mathbf{AC}^0$);
- but can we decide whether CERTAINTY(q) is in $\mathbf{P}$?

Boundaries do not coincide

There exists q such that CERTAINTY(q) is in $\mathbf{P}$ but not first-order expressible [Wij10b].

Tractability of CERTAINTY(q)

For q ranging over the class of Boolean acyclic conjunctive queries without self-join:

- we can decide whether CERTAINTY(q) is first-order expressible (and hence in $\mathbf{AC}^0$);
- but can we decide whether CERTAINTY(q) is in **P**?

Boundaries do not coincide

There exists q such that CERTAINTY(q) is in **P** but not first-order expressible [Wij10b].

Dichotomy conjecture

CERTAINTY(q) is in **P** or is **coNP**-complete.

For queries with exactly 2 atoms, this dichotomy has recently been proved by Kolaitis and Pena.

The Counting Problem $\natural$CERTAINTY(q)

Definition

For a fixed Boolean query q, the problem $\natural$CERTAINTY(q) is:

Given an unordered database,

how many possible presents satisfy q?

The Counting Problem $\natural$CERTAINTY(q)

Definition

For a fixed Boolean query q, the problem $\natural$CERTAINTY(q) is:

Given an unordered database,

how many possible presents satisfy q?

Example

R	$\underline{FN}$	$\underline{LN}$	$Dname$
	Mary	Smith	R&D
	Mary	Smith	MIS
	Mary	Smith	Toys

T	$\underline{Dname}$	$Budget$
	R&D	60K
	MIS	60K
	Toys	60K
	Toys	70K

$$q_2 \quad = \quad \exists x \exists y (R(\underline{\text{Mary}, x}, y) \wedge T(\underline{y}, 60\text{K}))$$

$\rightsquigarrow$ q_2 is true in 5 possible presents (out of 6)

The Counting Problem $\natural$CERTAINTY(q)

Definition

For a fixed Boolean query q, the problem $\natural$CERTAINTY(q) is:

Given an unordered database,

how many possible presents satisfy q?

Example

R	FN	LN	Dname
	Mary	Smith	Toys

T	Dname	Budget
	R&D	60K
	MIS	60K
	Toys	70K

$$q_2 = \exists x \exists y (R(\underline{\text{Mary}, x}, y) \wedge T(\underline{y}, 60\text{K}))$$

$\rightsquigarrow$ q_2 is true in 5 possible presents (out of 6)

Complexity Dichotomy for $\natural$CERTAINTY(q)

Dichotomy [MW11]

For every Boolean conjunctive query q without self-join, at least one of the following holds:

- $\natural$CERTAINTY(q) is in **P**; or
- $\natural$CERTAINTY(q) is $\natural$**P**-complete under polynomial-time Turing reductions.

Complexity Dichotomy for $\natural$CERTAINTY(q)

Dichotomy [MW11]

For every Boolean conjunctive query q without self-join, at least one of the following holds:

- $\natural$CERTAINTY(q) is in **P**; or
- $\natural$CERTAINTY(q) is $\natural$**P**-complete under polynomial-time Turing reductions.

Other dichotomy

A similar dichotomy holds in probabilistic databases [DRS11].

Summary of Main Results

- Uncertainty about currency leads to high data complexity. There exist conjunctive queries q, q' and sets Σ, Σ' of denials such that:
  - CERTAINTY(q) is **coNP**-hard
  - CONSISTENCY(Σ) is **NP**-hard
  - CPP(q', Σ'), certainty preservation under copying, is Σ_2^p-hard
- But:
  - For acyclic conjunctive queries q without self-join, we can decide whether CERTAINTY(q) is first-order definable (and hence in $\mathbf{AC}^0$).
  - For conjunctive queries q without self-join, we can decide whether $\natural$CERTAINTY(q) is in **P**.

References I

Xin Luna Dong, Laure Berti-Equille, and Divesh Srivastava.
Truth discovery and copying detection in a dynamic world.
PVLDB, 2(1):562–573, 2009.

Nilesh N. Dalvi, Christopher Re, and Dan Suciu.
Queries and materialized views on probabilistic databases.
J. Comput. Syst. Sci., 77(3):473–490, 2011.

Wayne W. Eckerson.
Data quality and the bottom line: Achieving business success through a commitment to high quality data.
The Data Warehousing Institute, 2002.

Wenfei Fan, Floris Geerts, and Jef Wijsen.
Determining the currency of data.
In Maurizio Lenzerini and Thomas Schwentick, editors, *PODS*, pages 71–82. ACM, 2011.

Hongfei Guo, Per-Åke Larson, and Raghu Ramakrishnan.
Caching with 'good enough' currency, consistency, and completeness.
In Klemens Böhm, Christian S. Jensen, Laura M. Haas, Martin L. Kersten, Per-Åke Larson, and Beng Chin Ooi, editors, *VLDB*, pages 457–468. ACM, 2005.

Dany Maslowski and Jef Wijsen.
On counting database repairs.
In *Proceedings of the 4th International Workshop on Logic in Databases*, LID '11, pages 15–22, New York, NY, USA, 2011. ACM.

References II

Jef Wijsen.
On the consistent rewriting of conjunctive queries under primary key constraints.
Inf. Syst., 34(7):578–601, 2009.

Jef Wijsen.
On the first-order expressibility of computing certain answers to conjunctive queries over uncertain databases.
In Jan Paredaens and Dirk Van Gucht, editors, *PODS*, pages 179–190. ACM, 2010.

Jef Wijsen.
A remark on the complexity of consistent conjunctive query answering under primary key violations.
Information Processing Letters, 110(21):950 – 955, 2010.