Repair Checking in Inconsistent Databases: Algorithms and Complexity

Foto Afrati[1] Phokion G. Kolaitis[2]

[1]National Technical University of Athens

[2]UC Santa Cruz & IBM Almaden Research Center

Logic and Databases

- In 1969, E.F. Codd introduced the relational data model.
- During the past 40 years, there has been a continuous and extensive interaction between logic and databases.
- Two main uses of logic in databases:
  - Logic is used as a database query language.
  - Logic is used to specify integrity constraints in databases.

The Relational Data Model

- Relational Database
  - Collection $(R_1, \ldots, R_m)$ of finite relations (tables).
  - Relational database $\sim$ Finite relational structure $\mathbf{A} = (A, R_1, \ldots, R_m)$.
- Relational Query Languages
  - Relational Algebra: Operations π, σ, $\times$, $\cup$, $\setminus$
  - Relational Calculus: (Safe) First-Order Logic
  - SQL: The standard commercial database query language based on relational algebra and relational calculus.

Integrity Constraints in Relational Databases

Extensive study of various types of integrity constraints in relational databases during the 1970s and early 1980s:

- Key constraints and functional dependencies
- Inclusion dependencies, join dependencies, multi-valued dependencies, ...

Eventually, it was realized that all these different types of dependencies can be specified in fragments of first-order logic.

Two Unifying Classes of Integrity Constraints

Definition

- *Equality-generating dependency (egd)*:
$$\forall \mathbf{x}(\phi(\mathbf{x}) \rightarrow x_i = x_j),$$
where $\phi(\mathbf{x})$ is a conjunction of atoms.

  Special Cases: Key constraints, functional dependencies.

- *Tuple-generating dependency (tgd)*:
$$\forall \mathbf{x}(\phi(\mathbf{x}) \rightarrow \exists \mathbf{y}\psi(\mathbf{x}, \mathbf{y})),$$
where $\phi(\mathbf{x})$ is a conjunction of atoms with vars. in $\mathbf{x}$, and $\psi(\mathbf{x}, \mathbf{y})$ is a conjunction of atoms with vars. in $\mathbf{x}$ and $\mathbf{y}$.

  Special Cases: Inclusion dependencies, etc.

Example

- $\forall x, y, z, w(R(x, y, z) \wedge R(x, y, w) \rightarrow z = w)$
  A key constraint written as an egd.
- $\forall x, y(P(x, y) \rightarrow \exists z Q(z, x))$
  An inclusion dependency written as a tgd.
- $\forall x, y, z(T(x, y) \wedge T(y, z) \rightarrow T(x, z))$
  Transitivity is specified by a tgd
- $\forall x, y(E(x, y) \rightarrow \exists z(F(x, z) \wedge F(z, y)))$
  A tgd that "transforms" edges to paths of length 2

Definition

- *Tuple-generating dependency (tgd)*:
$$\forall \mathbf{x}(\phi(\mathbf{x}) \rightarrow \exists \mathbf{y}\psi(\mathbf{x}, \mathbf{y})),$$
where $\phi(\mathbf{x})$ is a conjunction of atoms with vars. in $\mathbf{x}$, and $\psi(\mathbf{x}, \mathbf{y})$ is a conjunction of atoms with vars. in $\mathbf{x}$ and $\mathbf{y}$.

Note

"A Formal System for Euclid's *Elements*"
by Avigad, Denn, Muma - 2009:

All theorems in Euclid's *Elements* can be expressed by tgds!

Coping with Inconsistent Databases

- Inconsistent databases arise in a variety of contexts and for different reasons:
  - In data integration of heterogeneous data obeying different integrity constraints.
  - In data warehousing and in ETL (Extract-Transform-Load) applications, where data has to be “cleansed" before it can be processed.
  - For lack of support of particular integrity constraints.
  - ...

Coping with Inconsistent Databases

- Inconsistent databases arise in a variety of contexts and for different reasons:
  - In data integration of heterogeneous data obeying different integrity constraints.
  - In data warehousing and in ETL (Extract-Transform-Load) applications, where data has to be “cleansed" before it can be processed.
  - For lack of support of particular integrity constraints.
  - ...
- Database repairs provide a framework for coping with inconsistent databases in a principled way and without “cleansing" dirty data first.

Database Repairs

Definition (Arenas, Bertossi, Chomicki – 1999)

Σ a set of integrity constraints and r an inconsistent database. A database r' is a *repair* of r w.r.t. Σ if

- r' is a consistent database (i.e., $r' \models \Sigma$);
- r' differs from r in a minimal way.

Database Repairs

Definition (Arenas, Bertossi, Chomicki – 1999)

Σ a set of integrity constraints and r an inconsistent database. A database r' is a *repair* of r w.r.t. Σ if

- r' is a consistent database (i.e., $r' \models \Sigma$);
- r' differs from r in a minimal way.

Fact

Several different types of repairs have been considered:

- Subset-repairs;
- $\oplus$-repairs (symmetric-difference-repairs);
- Cardinality-based repairs;
- Attribute-based repairs.

Types of Repairs

Definition

Σ a set of integrity constraints and r an inconsistent database.

- r' is a *subset-repair* of r w.r.t. Σ if $r' \subset r$, $r' \models \Sigma$, and there is no r'' such that $r' \subset r'' \subset r$ and $r'' \models \Sigma$.
- r' is a *$\oplus$-repair* of r w.r.t. Σ if $r' \models \Sigma$ and there is no r'' such that $r \oplus r'' \subset r \oplus r'$ and $r'' \models \Sigma$.

Types of Repairs

Definition

Σ a set of integrity constraints and r an inconsistent database.

- r' is a *subset-repair* of r w.r.t. Σ if $r' \subset r$, $r' \models \Sigma$, and there is no r'' such that $r' \subset r'' \subset r$ and $r'' \models \Sigma$.
- r' is a *$\oplus$-repair* of r w.r.t. Σ if $r' \models \Sigma$ and there is no r'' such that $r \oplus r'' \subset r \oplus r'$ and $r'' \models \Sigma$.

Fact

- If $r' \subset r$, then r' is a subset-repair of r if and only if r' is a $\oplus$-repair of r.
- If Σ is a set of functional dependencies, then every $\oplus$-repair is also a subset-repair.

Example

Relation schema R, instance $r = \{R(a,b), R(a,c), R(b,c)\}$

- $\Sigma = \{R(x,y) \wedge R(x,z) \rightarrow y = z\}$
  r has two $\oplus$-repairs (and subset repairs) w.r.t. Σ:
  - $r_1 = \{R(a,b), R(b,c)\}$
    and
  - $r_2 = \{R(a,c), R(b,c)\}$.

Example

Relation schema R, instance $r = \{R(a,b), R(a,c), R(b,c)\}$

- $\Sigma = \{R(x,y) \wedge R(x,z) \rightarrow y = z\}$
  r has two $\oplus$-repairs (and subset repairs) w.r.t. Σ:
  - $r_1 = \{R(a,b), R(b,c)\}$
    and
  - $r_2 = \{R(a,c), R(b,c)\}$.

- $\Sigma' = \{R(x,y) \rightarrow R(y,x)\}$
  r has eight $\oplus$-repairs w.r.t. Σ':
  - $r_1 = \emptyset$ (a subset repair)
  - $r_2 = \{R(a,b), R(b,a)\}$ (not a subset repair)
  - $r_3 = \{R(a,b), R(b,a), R(a,c), R(c,a)\}$
  - ...
    Exponentially many repairs, in general.

Possible Worlds and Certain Answers

Definition

Suppose that with every instance r over some schema **S**, we have associated a set $\mathcal{W}(r)$ of instances over some other (possibly different) schema **T** (the set of *possible worlds* of r).

If q is a query over **T**, then the *certain answers of q on r w.r.t. $\mathcal{W}(r)$* is

$$\text{certain}(q, r, \mathcal{W}(r)) = \bigcap \{q(r') : r' \in \mathcal{W}(r)\}.$$

Note

The certain answers is the standard semantics of queries in the context of *incomplete information*.

Repairs and Consistent Answers

Definition (Arenas, Bertossi, Chomicki)

Fix a particular type of repairs (say, subset repairs or $\oplus$-repairs)
Let Σ be a set of integrity constraints, let q be a query, and let r be an instance.
The *consistent answers of q on r w.r.t. Σ*, denoted by $\mathrm{cons}_\Sigma(q, r)$, is the set $\mathrm{certain}(q, r, \mathcal{W}(r))$, where $\mathcal{W}(r)$ is the set of all repairs of r, i.e.,

$$\mathrm{cons}_\Sigma(q, r) = \bigcap \{q(r') : r' \text{ is a repair of } r\}.$$

Example (Revisited)

Relation schema R, instance $r = \{R(a,b), R(a,c), R(b,c)\}$
$\Sigma = \{R(x,y) \wedge R(x,z) \rightarrow y = z\}$
Recall that r has two $\oplus$-repairs (and subset repairs) w.r.t. Σ:

- $r_1 = \{R(a,b), R(b,c)\}$
  and
- $r_2 = \{R(a,c), R(b,c)\}$.

Then

- If $q(x)$ is the query $\exists z R(x,z)$, then

$$\text{cons}_\Sigma(q,r) = \{a,b\}.$$

- If $q'(x)$ is the query $\exists z R(z,x)$, then

$$\text{cons}_\Sigma(q',r) = \{c\}.$$

Data Complexity of Consistent Query Answering

Problem

Let Σ be a set of integrity constraints and q a fixed database query. Given an instance r, compute $\text{cons}_\Sigma(q, I)$.

Note

- In general, r may have exponentially many repairs.
- What can we say about the computational complexity of this problem?

Data Complexity of Consistent Query Answering

Theorem (Chomicki and Marcinkowski - 2003)

There exist a set Σ of two functional dependencies (in fact, key constraints) and a Boolean conjunctive query q such that the following problem is coNP-complete:
Given an instance r, is $\text{cons}_\Sigma(q, r)$ true?

Proof.

- Functional dependencies: $A \rightarrow BC$, $B \rightarrow AC$.
- Boolean conjunctive query: $\exists x \exists y R(x, y, b)$, where b is a constant.
- Reduction from 3-COLORABILITY.

Data Complexity of Consistent Query Answering

Theorem (Staworko - 2007)

There exist a set Σ consisting of one functional dependency and two universal constraints, and a Boolean conjunctive query q such that the following problem is Π_2^p-complete:
Given an instance r, is $\mathrm{cons}_\Sigma(q, r)$ true?

Note

- A *universal constraint* is a first-order formula of the form
  $$\forall x_1 \cdots \forall x_k (A_1 \vee \cdots \vee A_m \vee \neg A_{m+1} \vee \cdots \vee \neg A_n \vee \psi),$$
  where each A_i is an atom and ψ is a quantifier-free formula built from $=, \neq, <, >$.
- The proof uses a schema with a single relation symbol of arity 18 and a reduction from Π_2^p-SAT.

Data Complexity of Consistent Query Answering

Extensive study over the past decade for various classes of integrity constraints and for different types of repairs.

- Intractability results (coNP-hardness, Π_2^p-hardness)
- Tractability results for restricted classes of conjunctive queries:
  - Polynomial-time algorithms.
  - Rewriting to first-order queries.
- Prototype systems for consistent query answering:
  - Hippo (Chomicki et al.)
  - ConQuer (Fuxman)

Note

For overviews, see the invited paper by J. Chomicki in ICDT 2007 and the Ph.D. theses of A. Fuxman (2007) and S. Staworko (2007).

Algorithmic Problems about Inconsistent Databases

- **The Consistent Query Answering Problem:**
  - Consistent query answering has been investigated in depth.
- **The Repair Checking Problem:**
  - Given r and r', tell whether or not r' is a repair of r.
  - Repair checking is a data cleaning problem that underlies consistent query answering.
  - So far, repair checking has received less attention than consistent query answering.

Repair Checking vs. Consistent Query Answering

Proposition (Chomicki and Marcinkowski - 2003)

Let Σ be a set of integrity constraints containing all inclusion dependencies. There is a Boolean query q such that the $\oplus$-repair checking problem w.r.t. Σ has a logspace-reduction to the complement of the consistent query answering problem for q w.r.t. Σ

Note

Thus, in many cases, lower bounds for the complexity of the $\oplus$-repair checking problem yield lower bounds for the complexity of the consistent query answering problem.

Aim of this Work

Embark on a systematic investigation of the algorithmic aspects of the repair checking problem

- Study classes of integrity constraints that have been considered in information integration and data exchange.
- Study subset-repairs and $\oplus$-repairs.
- Introduce and study *CC-repairs* (*component-cardinality repairs*), a new type of cardinality-based repairs that have a Pareto-optimality character.

Types of Constraints

Definition

- *Equality-generating dependency (egd)*: $\forall \mathbf{x}(\phi(\mathbf{x}) \rightarrow x_i = x_j)$, where $\phi(\mathbf{x})$ is a conjunction of atoms.
- *Denial constraint*: $\forall \mathbf{x} \neg(\alpha(\mathbf{x}) \wedge \beta(\mathbf{x}))$, where $\alpha(\mathbf{x})$ is a non-empty conjunction of atoms and $\beta(\mathbf{x})$ is a conjunction of comparison atoms $x_i = x_j$, $x_i \neq x_j$, $x_i < x_j$, $x_i \leq x_j$.

Example

- Every functional dependency is an egd, but not vice versa: $\forall x, y, z(\mathrm{MOTHER}(z, x) \wedge \mathrm{MOTHER}(w, x) \rightarrow z = w)$.
- Every egd is (logically equivalent) to a denial constraint, but not vice versa: $\forall x, y \neg(\mathrm{MOTHER}(x, y) \wedge x = y))$

Types of Constraints

Definition

- *Tuple-generating dependency (tgd)*:
$$\forall \mathbf{x}(\phi(\mathbf{x}) \rightarrow \exists \mathbf{y}\psi(\mathbf{x}, \mathbf{y})),$$
where $\phi(\mathbf{x})$ is a conjunction of atoms with vars. in $\mathbf{x}$, and $\psi(\mathbf{x}, \mathbf{y})$ is a conjunction of atoms with vars. in $\mathbf{x}$ and $\mathbf{y}$.

- *Full tgd*: a tgd with no existential quantifiers in rhs.
$$\forall \mathbf{x}(\phi(\mathbf{x}) \rightarrow \psi(\mathbf{x})),$$
where $\phi(\mathbf{x})$ and $\psi(\mathbf{x})$ are conjunctions of atoms.

- *LAV (local-as-view) tgd*: a tgd in which lhs is a single atom.
$$\forall \mathbf{x}(P(\mathbf{x}) \rightarrow \exists \mathbf{y}\psi(\mathbf{x}, \mathbf{y})).$$
Note: Every inclusion dependency is a LAV tgd, but not vice versa.

Types of Constraints

Example

(dropping universal quantifiers)

- The following is a tgd

$$(\text{MOTHER}(z,x) \wedge \text{MOTHER}(z,y) \rightarrow \exists u(\text{FATHER}(u,x) \wedge \text{FATHER}(u,y))) \quad .$$

- The following are full tgds:

$$(\text{SIBLING}(x,y) \rightarrow \text{SIBLING}(y,x))$$

$$(\text{MOTHER}(z,x) \wedge \text{MOTHER}(z,y) \rightarrow \text{SIBLING}(x,y))$$

- The following is a LAV tgd:

$$(\text{SIBLING}(x,y) \rightarrow \exists z(\text{MOTHER}(z,x) \wedge \text{MOTHER}(z,y)))$$

Types of Repairs

Definition

Σ a set of integrity constraints and r an inconsistent database.

- r' is a *subset-repair* of r w.r.t. Σ if $r' \subset r$, $r' \models \Sigma$, and there is no r'' such that $r' \subset r'' \subset r$ and $r'' \models \Sigma$.
- r' is a *$\oplus$-repair* of r w.r.t. Σ if $r' \models \Sigma$ and there is no r'' such that $r \oplus r'' \subset r \oplus r'$ and $r'' \models \Sigma$.

Fact

- If $r' \subset r$, then r' is a subset-repair of r if and only if r' is a $\oplus$-repair of r.
- If Σ is a set of denial constraints, then every $\oplus$-repair is also a subset-repair.

Earlier Work on Subset Repairs - Tractability Results

Theorem

- folklore
  If Σ is a set of denial constraints, then the subset-repair checking problem w.r.t. Σ is in LOGSPACE.

Earlier Work on Subset Repairs - Tractability Results

Theorem

- folklore
  If Σ is a set of denial constraints, then the subset-repair checking problem w.r.t. Σ is in LOGSPACE.
- Chomicki and Marcinkowski – 2005
  If Σ is the union of an acyclic set of inclusion dependencies and a set of functional dependencies, then the subset-repair checking problem w.r.t. Σ is in PTIME; in fact, it is in LOGSPACE.

Earlier Work on Subset Repairs - Tractability Results

Theorem

- folklore
  If Σ is a set of denial constraints, then the subset-repair checking problem w.r.t. Σ is in LOGSPACE.
- Chomicki and Marcinkowski – 2005
  If Σ is the union of an acyclic set of inclusion dependencies and a set of functional dependencies, then the subset-repair checking problem w.r.t. Σ is in PTIME; in fact, it is in LOGSPACE.
- Staworko – 2007
  If Σ is a set of full tgds and egds, then the subset-repair checking problem w.r.t. Σ is in PTIME.

Earlier Work on Subset Repairs - Intractability Results

Theorem (Chomicki and Marcinkowski - 2005)

There is a set Σ consisting of one inclusion dependency and one functional dependency such that the subset-repair checking problem w.r.t. Σ is coNP-complete.

Earlier Work on Subset Repairs - Intractability Results

Theorem (Chomicki and Marcinkowski - 2005)

There is a set Σ consisting of one inclusion dependency and one functional dependency such that the subset-repair checking problem w.r.t. Σ is coNP-complete.

Proof.

- The inclusion dependency is
$$R(x_1, x_2, x_3, x_4) \rightarrow \exists y_1, y_2 y_3 R(y_1, y_2, x_4, y_3)$$
(i.e., $R[A_3] \subseteq R[A_4]$)
- The functional dependency is
$$A_1 \rightarrow A_2$$
- Reduction from SAT.

Weakly Acyclic Sets of Tgds

Fact

- Acyclic sets of inclusion dependencies and set of full tgds are special cases of weakly acyclic sets of tgds.
- Weakly acyclic sets of tgds are known to have good algorithmic behavior in data exchange and data integration.

Definition

- The *position graph* of a set Σ of tgds:
  - The nodes are the pairs (R, A), where R is a relation symbol and A is an attribute of R. Such a pair (R, A) is called a *position*.
  - Let $\phi(\mathbf{x}) \rightarrow \exists \mathbf{y} \psi(\mathbf{x}, \mathbf{y})$ be a tgd in Σ and let x in $\mathbf{x}$ be a variable that also occurs in $\psi(\mathbf{x}, \mathbf{y})$. For every occurrence of x in $\phi(\mathbf{x})$ in position (R, A_i), add the following edges:

    (i) For every occurrence of x in $\psi(\mathbf{x}, \mathbf{y})$ in position (S, B_j), add an edge $(R, A_i) \rightarrow (S, B_j)$;

    (ii) In addition, for every existentially quantified variable y in $\mathbf{y}$ and for every occurrence of y in $\psi(\mathbf{x}, \mathbf{y})$ in position (T, C_k), add a *special edge* $(R, A_i) \stackrel{*}{\rightarrow} (T, C_k)$.
- Σ is *weakly acyclic* if the position graph has no cycle going through a special edge.
- A tgd θ is *weakly acyclic* if $\{\theta\}$ is weakly acyclic.

Weakly Acyclic Sets of Tgds

Fact

- Every acyclic set of inclusion dependencies is a weakly acyclic set (the position graph is acyclic)
- Every set of full tgds is weakly acyclic (the position graph has no special edges).

Example

$$\Sigma = \{D(e,m) \rightarrow M(m), M(m) \rightarrow \exists e D(e,m)\}$$

is a weakly acyclic, but cyclic, set of inclusion dependencies.

Position graph:

$$D.1 \overset{*}{\leftarrow} M.1 \leftrightarrows D.2$$

Weakly Acyclic Sets of Tgds

Example

The inclusion dependency

$$R(x_1, x_2, x_3, x_4) \rightarrow \exists y_1, y_2 y_3 R(y_1, y_2, x_4, y_3),$$
$$(\text{i.e., } R[A_4] \subseteq R[A_3])$$

is not weakly acyclic because its position graph contains a special self-loop $R.4 \stackrel{*}{\rightarrow} R.4$.

Weakly Acyclic Sets of Tgds

Fact

Weakly acyclic sets of tgds have good algorithmic behavior in data exchange and data integration. Specifically, there are PTIME algorithms for:

- Computing a canonical universal solution;
- Computing the core of the universal solutions;
- Computing the certain answers of conjunctive queries.

Weakly Acyclic Sets of Tgds

Fact

Weakly acyclic sets of tgds have good algorithmic behavior in data exchange and data integration. Specifically, there are PTIME algorithms for:

- Computing a canonical universal solution;
- Computing the core of the universal solutions;
- Computing the certain answers of conjunctive queries.

Problem

Does the good algorithmic behavior of weakly acyclic sets of tgds extend to the repair checking problem?
(None of the earlier results on repair checking precludes this.)

Weak Acyclicity and Subset Repairs: Intractability

Theorem

There is a weakly acyclic set Σ of tgds such that the subset-repair checking problem w.r.t. Σ is coNP-complete.

Weak Acyclicity and Subset Repairs: Intractability

Theorem

There is a weakly acyclic set Σ of tgds such that the subset-repair checking problem w.r.t. Σ is coNP-complete.

Proof.

coNP-hardness via a reduction from POSITIVE 1-IN-3-SAT

- Σ consists of the (non-LAV) tgd
  $A(w) \wedge P(x,y,z) \rightarrow$
  $\exists u_1,u_2,u_3(T(x,u_1) \wedge T(y,u_2) \wedge T(z,u_3) \wedge S(u_1,u_2,u_3))$
  and the two full tgds:
  $T(x,u) \wedge T(x,u') \wedge D(u,u') \rightarrow S(u,u,u),\ T(x,u) \rightarrow A(u)$.
- Σ is weakly acyclic: all special edges are from pos. of P to pos. of T and S; no position of P has an incoming edge.

Weak Acyclicity and Subset Repairs: Intractability

Compare

Theorem

There is a weakly acyclic set Σ of tgds such that the subset-repair checking problem w.r.t. Σ is coNP-complete.

with

Weak Acyclicity and Subset Repairs: Intractability

Compare

Theorem

There is a weakly acyclic set Σ of tgds such that the subset-repair checking problem w.r.t. Σ is coNP-complete.

with

Theorem (Chomicki and Marcinkowski - 2005)

There is a set Σ consisting of one inclusion dependency and one functional dependency such that the subset-repair checking problem w.r.t. Σ is coNP-complete.

Note: The inclusion dependency is

$$R(x_1, x_2, x_3, x_4) \rightarrow \exists y_1, y_2 y_3 R(y_1, y_2, x_4, y_3),$$

which is not weakly acyclic (special self-loop on $R.4$).

Weak Acyclicity and Subset Repairs: Tractability

Theorem

If Σ is the union of a weakly acyclic set of LAV tgds and a set of egds, then the subset-repair checking problem w.r.t. Σ is in PTIME; in fact, it is in LOGSPACE.

Weak Acyclicity and Subset Repairs: Tractability

Theorem

If Σ is the union of a weakly acyclic set of LAV tgds and a set of egds, then the subset-repair checking problem w.r.t. Σ is in PTIME; in fact, it is in LOGSPACE.

Proof Idea.

- Key property of LAV tgds: only single facts *fire* a tgd (and no combinations of facts). Hence, LAV tgds are preserved under unions of models.
- Key property of weakly acyclic sets of tgds: The solution aware chase terminates in polynomial time.

Note: The solution aware chase was used in the study of peer data exchange (Fuxman, K ..., Miller, Tan - 2005). □

Weak Acyclicity and Subset Repairs: Tractability

Lemma

Let Σ be the union of a weakly acyclic set of LAV tgds and a set of egds. Then there is a constant c such that the following holds.
Let r, r' be two instances such that $r' \models \Sigma$, and let t be a fact in $r \setminus r'$ such that there is a non-empty set $A \subset r \setminus r'$ such that $t \in A$ and $r' \cup A \models \Sigma$. There there is a set A_t of facts such that

- $t \in A_t$
- $A_t \subseteq r \setminus r'$
- $|A_t| \leq c$
- $r' \cup A_t \models \Sigma$.

Weak Acyclicity and Subset Repairs: Tractability

Algorithm for subset-repair checking w.r.t. a set Σ that is the union of a weakly acyclic set of LAV tgds and a set of egds.

Given r and r' with $r' \subset r$, $r \not\models \Sigma$, $r' \models \Sigma$:

Test whether there is a set A^* such that

1. A^* is non-empty
2. $|A^*| \leq c$
3. $A^* \subseteq r \setminus r'$
4. $r' \cup A^* \models \Sigma$.

- If such a set A^* exists, then r' is not a subset repair of r w.r.t. Σ;
- Otherwise, r' is a subset repair of r w.r.t. Σ.

Full Tgds and Subset Repairs: PTIME-completeness

Theorem (Staworko – 2007)

If Σ is a set of full tgds, then the subset repair checking problem w.r.t. Σ is in PTIME.

Full Tgds and Subset Repairs: PTIME-completeness

Theorem (Staworko – 2007)

If Σ is a set of full tgds, then the subset repair checking problem w.r.t. Σ is in PTIME.

Theorem

There is a set Σ of full tgds such that the subset-repair problem w.r.t. Σ is PTIME-complete.

Full Tgds and Subset Repairs: PTIME-completeness

Theorem (Staworko – 2007)

If Σ is a set of full tgds, then the subset repair checking problem w.r.t. Σ is in PTIME.

Theorem

There is a set Σ of full tgds such that the subset-repair problem w.r.t. Σ is PTIME-complete.

Proof *(Hint).*

- Logspace Reduction from HORN 3-SAT.
- Use full tgds to encode the unit propagation algorithm for HORN 3-SAT.

□

Complexity of Subset Repair Checking

Constraints	Subset-Repair Checking
Denial	LOGSPACE
Acyclic set of IND & egds	LOGSPACE
Weakly acyclic LAV tgds & egds	LOGSPACE
Full tgds & egds	PTIME-complete
IND & egds	coNP-complete
Weakly acyclic tgds	coNP-complete
Weakly acyclic tgds & egds	coNP-complete

Complexity of Subset Repair Checking

Constraints	Subset-Repair Checking
Denial	LOGSPACE
Acyclic set of IND & egds	LOGSPACE
Weakly acyclic LAV tgds & egds	LOGSPACE
Full tgds & egds	PTIME-complete
IND & egds	coNP-complete
Weakly acyclic tgds	coNP-complete
Weakly acyclic tgds & egds	coNP-complete

Note

- The presence of egds does not increase complexity.
- Good algorithmic behavior of acyclic sets of inclusion dependencies and sets of full tgds for subset-repair checking extends to weakly acyclic sets of LAV tgds, but not to arbitrary weakly acyclic sets of tgds.

Subset Repairs vs. $\oplus$-Repairs

Question: What is the complexity of the $\oplus$-repair checking problem w.r.t. to the classes of constraints considered thus far?

Subset Repairs vs. $\oplus$-Repairs

Question: What is the complexity of the $\oplus$-repair checking problem w.r.t. to the classes of constraints considered thus far?

Theorem (Staworko – 2007)

If Σ is a set of full tgds, then the $\oplus$-repair checking problem w.r.t. Σ is in PTIME.

Complexity of the $\oplus$-Repair Checking Problem

Good News:

Theorem

If Σ is a weakly acyclic set of LAV tgds, then the $\oplus$-repair problem w.r.t. Σ is in PTIME; in fact, it is in LOGSPACE.

Bad News:

Complexity of the $\oplus$-Repair Checking Problem

Good News:

Theorem

If Σ is a weakly acyclic set of LAV tgds, then the $\oplus$-repair problem w.r.t. Σ is in PTIME; in fact, it is in LOGSPACE.

Bad News:

Theorem

There is an acyclic set Σ_1 of inclusion dependencies and a set Σ_2 of egds such that the $\oplus$-repair checking problem w.r.t. $\Sigma_1 \cup \Sigma_2$ is coNP-complete.

Complexity of the $\oplus$-Repair Checking Problem

Theorem

If Σ is a weakly acyclic set of LAV tgds, then the $\oplus$-repair problem w.r.t. Σ is in PTIME; in fact, it is in LOGSPACE.

Complexity of the $\oplus$-Repair Checking Problem

Theorem

If Σ is a weakly acyclic set of LAV tgds, then the $\oplus$-repair problem w.r.t. Σ is in PTIME; in fact, it is in LOGSPACE.

Proof.

- Non-trivial extension of the LOGSPACE-algorithm for subset-repair checking for weakly acyclic sets of LAV tgds and egds.
- Additional structural lemmas have to be proved first.

$\square$

Complexity of the $\oplus$-Repair Checking Problem

Theorem

There is an acyclic set Σ_1 of inclusion dependencies tgds and a set Σ_2 of egds such that the $\oplus$-repair checking problem w.r.t. $\Sigma_1 \cup \Sigma_2$ is coNP-complete.

Proof. By reduction from POSITIVE 1-IN-3-SAT.

$$P_1(x,y,z) \rightarrow \exists w, w' P_{23}(w,w',x,y,z)$$
$$P_{23}(w,w,x,y,z) \rightarrow \exists u, v, w R(x,u,y,v,z,w)$$
$$R(x,u,y,v,z,w) \rightarrow S(u,v,w)$$
$$R(x,u,y,v,z,w) \wedge P_{23}(w,w',x',y',z') \rightarrow w = w'$$
$$R(x,u,y,v,z,w) \wedge R(x,u',y,v',z',w') \rightarrow u = u'$$
$$R(x,u,y,v,z,w) \wedge R(x',u',y,v',z',w') \rightarrow v = v'$$
$$R(x,u,y,v,z,w) \wedge R(x',u',y',v',z,w') \rightarrow w = w'$$

Complexity of Subset- and ⊕-Repair Checking

Constraints	Subset-Repairs	⊕-Repairs
Denial	LOGSPACE	LOGSPACE
Acyc. set of IND & egds	LOGSPACE	coNP-comp.
Weak. acyc. LAV tgds	LOGSPACE	LOGSPACE
Weak. acyc. LAV tgds & egds	LOGSPACE	coNP-comp.
Full tgds & egds	PTIME-comp.	PTIME-comp.
IND & egds	coNP-comp.	coNP-comp.
Weak. acyc. tgds & egds	coNP-comp.	coNP-comp.

Complexity of Subset- and ⊕-Repair Checking

Constraints	Subset-Repairs	⊕-Repairs
Denial	LOGSPACE	LOGSPACE
Acyc. set of IND & egds	LOGSPACE	coNP-comp.
Weak. acyc. LAV tgds	LOGSPACE	LOGSPACE
Weak. acyc. LAV tgds & egds	LOGSPACE	coNP-comp.
Full tgds & egds	PTIME-comp.	PTIME-comp.
IND & egds	coNP-comp.	coNP-comp.
Weak. acyc. tgds & egds	coNP-comp.	coNP-comp.

Note

Unlike subset repair checking, the presence of egds increases the complexity of the ⊕-repair checking problem even for acyclic sets of inclusion dependencies.

Synopsis

- Subset repair checking is in PTIME for weakly acyclic sets of LAV tgds and egds.
- $\oplus$-repair checking is in PTIME for weakly acyclic sets of LAV tgds.
- $\oplus$-repair checking can be coNP-complete for weakly acyclic sets of LAV tgds and egds.
- $\oplus$-repair checking can be coNP-complete for weakly acyclic sets of tgds.

Directions and Problems

- Open Problem: Prove or disprove that a *dichotomy theorem holds for the complexity of the $\oplus$-repair checking problem w.r.t. sets of tgds and egds.*
- Investigate the complexity of repair checking for other types of repairs (attribute-based repairs).
  Work in this direction has already been carried out by J. Wisjen.
- Are there criteria for differentiating between repairs of the same type?