Complexity and Approximation of Fixing Numerical Attributes in Databases Under Integrity Constraints

Leopoldo Bertossi
Carleton University
Ottawa, Canada

Join work with:
Loreto Bravo (Carleton U.)
Enrico Franconi, **Andrei Lopatenko** (U. Bolzano-Bozen)

Motivation I

Databases may become inconsistent wrt a given set of integrity constraints (ICs)

- DBMS has no mechanism to maintain certain classes of ICs
- Data of different sources are being integrated

  Even if the independent data sources are consistent, integrated data may not
- New constraints are imposed on pre-existing, legacy data
- Soft, user, or informational constraints, to be considered at query answering, but without being enforced

Considerable amount of research on consistent query answering has been carried out in the last 5 years

In (Arenas, Bertossi, Chomicki; PODS 99):

- Characterization of consistent answers to queries as those that are invariant under minimal repairs of the original database

- Mechanism for computing them (for certain classes of queries and ICs)

Most of the research has concentrated on repairs of databases that minimize under set inclusion the set of insertions/deletions of whole database tuples (no matter what type of data they contain)

No much interest in the repairs *per se* or their computation (unless absolutely necessary), but in consistent answers to queries

In some scenarios or applications these assumptions or semantic commitments may not be the most natural ones

Motivation II

We were motivated in this research by census-like data, where other ontological assumptions, fixes, and computational needs appear (Franconi,Laureti Palma, Leone, Perri, Scarcello; LPAR 01)

- Many attributes take numerical values
- Most natural form of fix is to change some of the values of attributes in tuples (Wijsen; ICDT 03)
- The tuples are identified by a key (e.g. household identifier) that is fixed and not subject to changes

  Identifiers are kept in any repaired version of the database (census form)
- Constraints are expressed as prohibitions on positive data, that can be captured by denial constraints

- Few relations in the database, actually not uncommon to have one single relation
- Computation of fixes (repairs) becomes crucial and a common task in census like applications
- Aggregate queries are most important, rather than queries about individual properties

Check out:

United Nations Economic Commission for Europe, Work Session on Statistical Data Editing (Ottawa, 16-18 May 2005)

`http://www.unece.org/stats/documents/2005.05.sde.htm`

For a glossary of terms and computational tasks related to data editing

`http://www.unece.org/stats/publications/editingglossary.pdf`

The Problem

We propose a notion of fix (repair) of a database containing numerical, integer values that:

- Allows, as basic fixing actions, changes of values in (changeable or flexible) attributes
- For the first time takes into account the occurrence of numerical values and their nature when distances between databases are measured in numerical terms
- Considers the presence of non-contradictable key constraints

In this scenario, completely new issues and problems appear:

- Computing database fixes
- Determining existence of fixes, e.g. not beyond a certain distance from the original database
- Consistent query answering to aggregate queries wrt to denial constraints
- ...

Example 1: A denial constraint DC: *Pay at most* 6000 *to employees with less than 5 years of experience*

Database D, with Name as the key, is inconsistent wrt DC

Employee	Name	Experience	Salary
	Sarah	6	12000
	Robert	4	7000
	Daniel	5	8000

Under tuple and set oriented semantics for repairs, the only minimal repair corresponds to deleting the tuple Employee(Robert, 4, 7000)

Other options may make more sense that deleting employee Robert (a value for the key):

- Change the violating tuple to Employee(Robert,5, 7000)
- Change it to Employee(Robert, 4, 6000)

They satisfy the implicit requirements that:

- Key values are kept
- Numerical values associated to them do not change too much

Minimum Numerical Fixes

We concentrate on linear denial constraints (LDCs) of the form

$$\forall \bar{x} \neg (A_1 \wedge \ldots \wedge A_m)$$

- A_i are database atoms, or built-in atoms of the form $X \theta c$, with $\theta \in \{=, \neq, <, >, \leq, \geq\}$, or $X = Y$

  (the latter can be replaced by different occurrences of the same variable)

- If we allow atoms of the form $X \neq Y$, we call them extended linear denial constraints (ELDs)

Examples: *Pay at most* 6000 *to employees with less than 5 years of experience*

$$\forall Name, Experience, Salary \neg (Employee(Name, Experience, Salary), Experience < 5, Salary > 6000)$$

Distance between Instances:

- When numerical values are updated to restore consistency, it is desirable to make the smallest overall variation of the original values
- A database and a fix share the same key values
- Key values can be used to compute associated numerical variations

To compare instances, we need a common relational schema $\mathcal{R}$, a set of key constraints $\mathcal{K}$, assumed to be satisfied by all the instances, the set of attributes $\mathcal{A}$, a subset of fixable, numerical attributes $\mathcal{F}$ (not containing any attributes in the key)

For a tuple $\bar{k}$ of key values in relation R in instance D, $\bar{t}(\bar{k}, R, D)$ denotes the unique tuple $\bar{t}$ in relation R in instance D whose key value is $\bar{k}$

Example 1: (cont.) $\mathcal{R} = \{Employee\}$, $\mathcal{A} = \{Name, Experience, Salary\}$, $\mathcal{F} = \{Experience, Salary\}$ $\quad Key(Employee) = \{Name\}$

Employee	Name	Experience	Salary
	Sarah	6	12000
	Robert	4	7000
	Daniel	5	8000

The (kept) key values are: $Sarah$, $Robert$, $Daniel$

$\bar{t}(Sarah, Employee, D) = (Sarah, 6, 12000)$, etc.

Example 2: D has tables $Client(\underline{ID}, A, M)$, A is age, M is money; and $Buy(\underline{ID, I}, P)$, I is items, P is price

Denials: *People younger than 18 cannot spend more than 25 on one item nor spend more than 50 in the store*

$IC_1: \forall ID, P, A, M \neg(Buy(ID, I, P), Client(ID, A, M), A < 18, P > 25)$

$IC_2: \forall ID, A, M \neg(Client(ID, A, M), A < 18, M > 50)$

D:

Client	**ID**	**A**	**M**	
	1	15	52	t_1
	2	16	51	t_2
	3	60	900	t_3
Buy	**ID**	**I**	**P**	
	1	CD	27	t_4
	1	DVD	26	t_5
	3	DVD	40	t_6

IC_1 is violated by $\{t_1, t_4\}$ and $\{t_1, t_5\}$; IC_2 by $\{t_1\}$ and $\{t_2\}$

$\bar{t}((1, CD), Buy, D) = Buy(1, CD, 27)$, etc. $\mathcal{F} = \{A, M, P\}$

For instances D, D' over the same schema and same key values for each relation $R \in \mathcal{R}$, their square distance is

$$\Delta(D, D') = \sum \alpha_A [\pi_A(\bar{t}(\bar{k}, R, D)) - \pi_A(\bar{t}(\bar{k}, R, D'))]^2$$

- Sum is over all relations R, fixable numerical attributes $A \in \mathcal{F}$ and tuples of key values $\bar{k}$
- π_A is the projection on attribute A
- α_A is the -application dependent- weight of attribute A

Other numerical distance functions can be considered, e.g. "city distance", obtaining results similar to those that follow

Given an instance D, with $D \models \mathcal{K}$, and a set of possibly violated ELDCs IC, a least-squares fix (LS-fix) for D wrt IC is an instance D' with:

1. Same schema and domain as D
2. Same values as D in the non-fixable attributes $\mathcal{A} \setminus \mathcal{F}$ (in particular in key attributes)
3. $D' \models \mathcal{K} \cup IC$;
4. $\Delta(D, D')$ is minimum over all the instances that satisfy 1. - 3.

$LS\text{–}Fix(D, IC) := \{D' \mid D' \text{ is an LS-fix of } D \text{ wrt } IC\}$

$Fix(D, IC)$ defined as $LS\text{–}Fix(D, IC)$, but without minimality condition

Example 1: (cont.) Candidates to fixes were

$$D_1 = \{(Sarah, 6,\ 12000), (Robert, 5,\ 7000),\ (Daniel, 5,\ 8000)\}$$

$$D_2 = \{(Sarah, 6, 12000), (Robert, 4,\ 6000),\ (Daniel, 5,\ 8000)\}$$

With $\alpha_{Salary} = 10^{-6}$, $\alpha_{Experience} = 1$:

Square distances to D: $\Delta(D, D_1) = 1$, $\Delta(D, D_2) = 4$

Only D_1 is an LS-fix

Example 2: (cont.)

IC_1: $\forall ID, P, A, M \neg(Buy(ID, I, P), Client(ID, A, M), A < 18, P > 25)$

IC_2: $\forall ID, A, M \neg(Client(ID, A, M), A < 18, M > 50)$

D:

Client	**ID**	**A**	**M**	
	1	15	52	t_1
	2	16	51	t_2
	3	60	900	t_3
Buy	**ID**	**I**	**P**	
	1	CD	27	t_4
	1	DVD	26	t_5
	3	DVD	40	t_6

Assume $\alpha_A = \alpha_M = \alpha_P = 1$

A (minimal) fix D' with $cost = 1^2 + 2^2 + 1^2 + 2^2 = 10$

Client'	**ID**	**A**	**M**	
	1	15	~~52~~ 50	t_1'
	2	16	~~51~~ 50	t_2'
	3	60	900	t_3
Buy'	**ID**	**I**	**P**	
	1	CD	~~27~~ 25	t_4'
	1	DVD	~~26~~ 25	t_5'
	3	DVD	40	t_6

A fix D'' with $cost = 1^2 + 3^2 = 10$

Client"	**ID**	**A**	**M**	
	1	~~15~~ 18	52	t_1''
	2	16	~~51~~ 50	t_2''
	3	60	900	t_3
Buy"	**ID**	**I**	**P**	
	1	CD	27	t_4
	1	DVD	26	t_5
	3	DVD	40	t_6

In this example, it was possible to obtain LS-fixes by performing direct, local changes in the original conflictive tuples alone

No new, intermediate inconsistencies introduced in the repair process

This may not be always the case ...

Decidability and Complexity of LS-fixes

It is relevant to ask about the existence of fixes

In contrast to the "classical" case of CQA, there may be no fixes

We concentrate on data complexity and denial constraints

The number of fixes can be exponential, actually

For (E)LDCs:

$NE(IC) := \{D \mid Fix(D, IC) \neq \emptyset\}$ is NP-complete

The Database Fix Problem

$DFP(IC) := \{(D,k) |$ there is $D' \in Fix(D, IC)$ with $\Delta(D, D') \leq k\}$, the *database fix problem*

It is important for transformation of inconsistent database into the closest consistent state

- What is the distance to the closest consistent state?
- Make computations more efficient by cutting off incorrect (too expensive) branches during computation or materialization of a consistent state

Complexity of DFP provides lower bounds for CQA under some assumptions

Theorem: $DFP(IC)$, the problem of deciding whether there exists a fix wrt IC at a distance $\leq k$, is NP-complete

Hardness: By encoding "Vertex Cover"

One LDC with three database atoms plus two built-ins is good enough

Membership:

- Possible values in repaired tuples are determined by the constants in the DCs
- $PTIME$ computation of the distance function

Approximate Solutions to DFP

Given that $DFP(IC)$ is NP-complete, can we get a good and efficient approximate solution?

$DFOP$ denotes the optimization problem of finding the minimum distance to a fix

Theorem: For a fixed set of LDCs, $DFOP$ is $MAXSNP$-hard

Proof by reduction from "Minimum Vertex Cover Problem for Graphs of Bounded Degree" which is $MAXSNP$-complete

$MAXSNP$-hardness implies (unless $P = NP$) that there exists constant δ such that $DFOP$ for LDCs cannot be approximated within an approximation factor less than δ

Can we provide an approximation within a constant factor, possibly with restriction to a still useful but hard class of LDCs?

Definition: A set of LDCs IC is local if:

- Equalities between attributes and joins involve only non fixable attributes
- There is a built-in with a fixable attribute in each IC
- No attribute A appears in IC both in comparisons of the form $A < c_1$ and $A > c_2$

Example 2: (cont.) The LDCs there are local

$IC_1 : \forall ID, P, A, M \neg(Buy(ID, I, P), Client(ID, A, M), A < 18, P > 25)$

$IC_2 : \forall ID, A, M \neg(Client(ID, A, M), A < 18, M > 50)$

Why "local" and why interesting?

- Basically, inconsistencies can be fixed tuple by tuple, without introducing new violations
- LS-fixes always exist
- Local LDCs are the most common in census-like applications (Franconi,Laureti Palma, Leone, Perri, Scarcello; LPAR 01)
- DFP still NP-complete for local LDCs
- $DFOP$ still $MAXSNP$-hard for local LDCs
  (the non-local LDCs can be eliminated from the proof of the general case)

We will reduce $DFOP$ to the "Weighted Set Cover Problem"

A Weighted Set Cover Problem

The WSCP is defined as follows

Instance: $(U, \mathcal{S}, w)$

- $\mathcal{S}$ is a collection of subsets of set U
- $\bigcup \mathcal{S} = U$ (a cover)
- w assigns numerical weights to elements of $\mathcal{S}$

Question: Find a sub-collection of $\mathcal{S}$ with minimum weight that covers U

$WSCP$ is $MAXSNP$-hard

We create a "good" instance of $WSCP$ ...

1. A set I of database atoms (tuples) from D is a violation set for $ic \in IC$ if $I \not\models ic$, and for every $I' \subsetneq I$, $I' \models ic$, i.e. a minimal set of tuples that participate in the violation of an IC

   $$U := \text{set of violation sets for } D, IC$$

2. $\mathcal{I}(D, ic, t)$ denotes the set of violation sets for ic in D that contain tuple t

   $$S(t, t') := \{I \mid \text{ exists } ic \in IC,\ I \in \mathcal{I}(D, ic, t) \text{ and } ((I \setminus \{t\}) \cup \{t'\}) \models ic\},$$

   i.e. the violations sets containing tuple t that are solved by changing t to t'

$\mathcal{S}$:= collection of $S(t, t')$'s such that t' is a *local fix* of t

I.e.

(a) t, t' share same values in non-fixable attributes

(b) $S(t, t') \neq \emptyset$ (some inconsistency is solved)

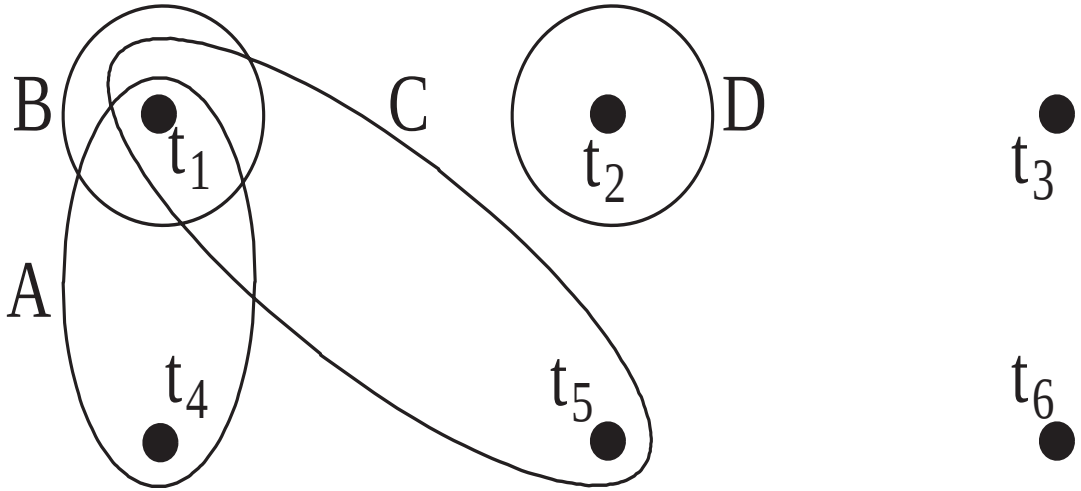
(c) $\Delta(\{t\}, \{t'\})$ is minimum (relative to (a), (b))

3. Finally, $w(S(t, t')) := \Delta(\{t\}, \{t'\})$

Properties of the Reduction:

- Local fixes can be obtained in polynomial time
- One-to-one correspondence between solutions to $DFOP$ and solutions to $WSCP$ that keeps the optimum values
- Each set cover corresponds to a consistent instance
- Transformation is in polynomial time
- Correspondence may be lost if applied to non-local LDCs (LDCs may not be satisfied by resulting instances)

Example 2: (cont.)



Client	ID	A	M	
	1	15	52	t_1
	2	16	51	t_2
	3	60	900	t_3
Buy	**ID**	**I**	**P**	
	1	CD	27	t_4
	1	DVD	26	t_5
	3	DVD	40	t_6

Violation sets: $\{t_1, t_4\}$, $\{t_1, t_5\}$ for IC_1; $\{t_1\}$, $\{t_2\}$ for IC_2

Tuple t_1: One local fix wrt IC_1, one wrt IC_2

Tuple t_2: One local fix Tuples t_4, t_5: One local fix each

Conflict (Hyper)Graphs: (Arenas, Bertossi, Chomicki; ICDT 01),
(Chomicki, Marcinkowski; Information and Computation 05)

Tuples, their local fixes, and weights ...

Set cover els.	$S(t_1, t_1')$	$S(t_1, t_1'')$	$S(t_2, t_2')$	$S(t_4, t_4')$	$S(t_5, t_5')$
Local Fix	t_1'	t_1''	t_2'	t_4'	t_5'
Weight	4	9	1	4	1
Violation set A	0	1	0	1	0
Violation set B	1	1	0	0	0
Violation set C	0	1	0	0	1
Violation set D	0	0	1	0	0

1 and 0 at the bottom indicate if the violation set belongs or not to $S(t, t')$

Approximating $DFOP$

We approximate a solution to $DFOP$ by approximating $WSCP$

In general, $WSCP$ can be approximated within a factor $O(log(n))$ by greedy algorithms (Chvatal)

In our case, the frequency (number of elements of $\mathcal{S}$ covering an element of U) is bounded by the maximum number of atoms in the ICs (small in general)

In this case $WSCP$ can be efficiently approximated within a constant factor given by the maximum frequency
(Hochbaum; 1997)

The approximation algorithm always returns a cover, from which we can compute an instance, that turns out to satisfy the LD-Cs, but may not be optimal (in example 2 we get D')

Consistent Query Answering

We have several results on data complexity for CQA, a decision or an optimization problem depending on the semantics

Some involving 1-atom denial constraints (1AD), a common case in census-like applications (one DB atom plus built-ins)

Usual semantics for CQA:

- Certain: Answer true in every LS-fix (default semantics)
- Possible: True in some LS-fix
- Range: Shortest numerical interval where all answers from LS-fixes can be found (for numerical, mainly aggregate queries) (Arenas, Bertossi, Chomicki; ICDT 2001)

  Two optimization problems: find extremes of the interval

Some results for CQA:

- Non aggregate queries, certain semantics:
  - 1ADs; atomic ground query (and a wide class of boolean conjunctive queries): $PTIME$
  - Arbitrary extended LDCs; atomic query: P^{NP}-hard and in Π_2^P
- Aggregate queries (with one of *sum*, *count distinct*, *average*) and possible semantics (boolean query true in some fix: comparison of the aggregation to a constant) or range semantics
  - 1ADs; acyclic conjunctive query: $coNP$-hard

E.g. 1AD: $\forall u, c_1, c_2 \neg(R(u, c_1, c_2) \wedge c_1 < 1 \wedge c_2 < 1)$
Query: $q(count(distinct\ z)) \leftarrow R(u, x, y), S(u, z, x)$

The Gist: COUNT DISTINCT Aggregation

CQA under range semantics for COUNT DISTINCT acyclic conjunctive queries and one 1AD is $coNP$-complete

By reduction from MAX-SAT with instance $P = \langle U, C, K \rangle$; U a set of variables, C collection of clauses over U, K a positive integer

- Introduce relation $Var(\underline{u}, C_1, C_2)$ with tuple $(u, 0, 0)$ for every variable in $u \in U$; C_1, C_2 fixable, taking values 0 or 1
- Introduce non-fixable relation $Clause(u, c, s)$, with tuple (u, c, s) for every occurrence of $u \in U$ in clause $c \in C$, s is assignment (0 or 1) for u satisfying clause c

- 1AD: $\forall u, c_1, c_2 \neg(Var(u, c_1, c_2) \wedge c_1 < 1 \wedge c_2 < 1)$

Query: $q(count(distinct\ c)) \leftarrow Var(u, c_1, c_2), Clause(u, c, s), c_1 = s$
asks for how many clauses are satisfied in a given fix

Max value in a fix is max number of clauses that can be satisfied for P

Approximating CQA for Aggregate Queries

Even for simple constraints, and simple aggregate queries defined on top of also simple conjunctive queries, CQA becomes hard

Finding the minimum/maximum values for an aggregate query among the LS-fixes become optimization problems

They are $MAXSNP$-hard for *sum*

For 1ADs and conjunctive aggregate query with *sum*, the maximum value for CQA (i.e. range semantics) can be efficiently approximated within a constant factor

Ongoing and Future Work

- Implementation issues and experimentation
- Applications in census-like scenarios
- Cases of polynomial complexity for LDCs with more that one database atoms
- Applications to data integration with preferences
- Approximation algorithm for DFP in more general cases
- Approximation algorithms for CQA to aggregate queries
- Fixes that preserve statistical validity; distribution preserving DFP and CQA
- Other numerical domains? Real numbers?

Summary of Results

DFP	1AD	Local LDC	LDC	ELDC + Agg. ICs
complexity	$PTIME$	NP-complete	NP-complete	undecidable
approximation	-	$MAXSNP$-hard const. factor	$MAXSNP$-hard	-

CQA	1AD	LDC
FO conjunctive query	$PTIME$ for $\mathcal{C}_{Tree}$	atomic $P^{NP(log(n))}$-complete general, in P^{NP}
aggregate query	Complex.: NP-complete possible and range **SUM**, **AVG**, **COUNT DISTINCT** Approx.: $MAXSNP$-hard const. factor, **SUM**	Complex.: P^{NP}-complete Approx.: n^{ϵ}-hard