Integrity Constraints in Data Exchange

Víctor Gutiérrez-Basulto

Universität Bremen

Universität Bremen

Basic Notions

Embedded Dependencies: Definition and sub-classes

FOL sentences of the form:

$$\varphi(\mathbf{x}) \to \exists \mathbf{y}\, \psi(\mathbf{x}, \mathbf{y})$$

- φ is a conjunction (possibly empty) of relational atoms;
- ψ is a conjunction of relational atoms and equality atoms.

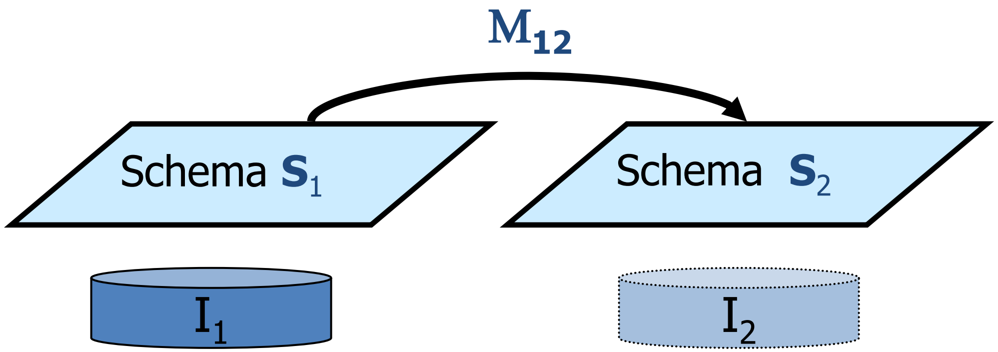
Three important sub-classes:

Full Dependency is a dependency that has no existential quantifiers.

Equality-Generating Dependency (EGD) allows only for equality atoms in ψ.

Tuple-Generating Dependency (TGD) allows only for relational atoms in ψ.

Schema Mappings



Provide:

High-Level & Declarative relationship between two schemas

Trade-Off:

Expressive vs Simple

Specification Language:

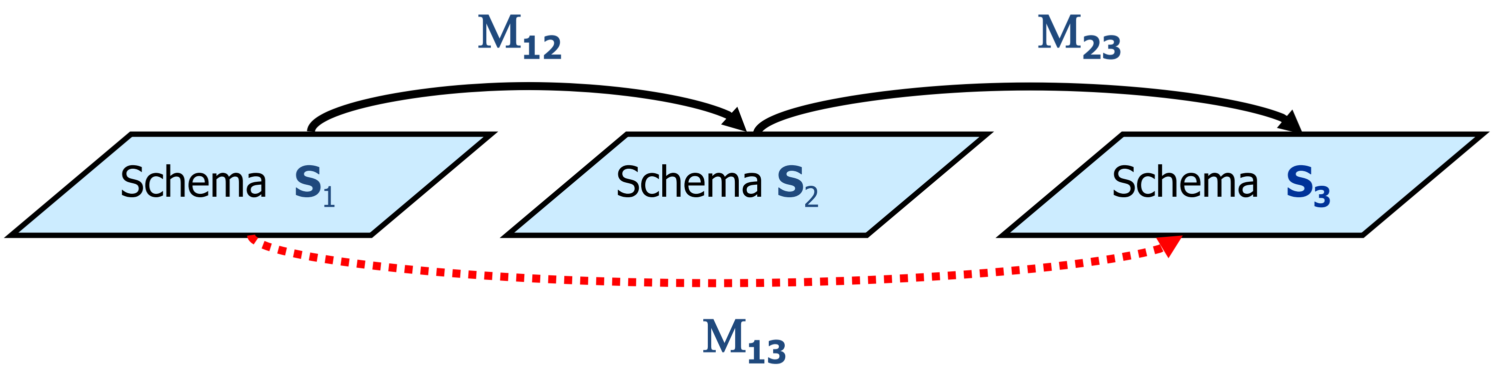
Use a well-behaved fragment of FOL

Data Exchange Setting with tgds and egds [FKMP 03]

Schema mapping $M = (\mathbf{S}, \mathbf{T}, \Sigma_{st} \cup \Sigma_t)$ such that

- Σ_{st} is a set of source to target tgds
- Σ_t is a set of target tgds and target egds

Composing Schema Mappings



Given $M_{12} = (\mathbf{S}_1, \mathbf{S}_2, \Sigma_{12})$ and $M_{23} = (\mathbf{S}_2, \mathbf{S}_3, \Sigma_{23})$ derive

a schema mapping $M_{13} = (\mathbf{S}_1, \mathbf{S}_3, \Sigma_{13})$

that is **equivalent** to the successive application of M_{12} and M_{23}

M_{13} is a **composition** of M_{12} and M_{23}

$$M_{13} = M_{12} \circ M_{23}$$

Universität Bremen

Semantics of Composition

A relationship between instances:

Every schema mapping $M = (\mathbf{S}, \mathbf{T}, \Sigma)$ defines

$$\mathbf{Inst}(M) = \{\langle I, J \rangle \mid \langle I, J \rangle \models \Sigma\}$$

A Formal Definition [FKPT05]

A schema mapping M_{13} is a **composition** of M_{12} and M_{23} if

$$\mathbf{Inst}(M_{13}) = \mathbf{Inst}(M_{12}) \circ \mathbf{Inst}(M_{23}), \text{ i.e.,}$$

$$\langle I_1, I_3 \rangle \models \Sigma_{13}$$

if and only if

there exists I_2 s.t. $\langle I_1, I_2 \rangle \models \Sigma_{12}$ and $\langle I_2, I_3 \rangle \models \Sigma_{13}$.

Issues in Composition of Schema Mappings

Closure of Schemma Mapping Language under Composition:

M_{12} and M_{23} are specified by sets of formulas of some logic $\mathcal{L}$.

Is $M_{12} \circ M_{23}$ definable in $\mathcal{L}$?

s-t tgds [FKPT05]

The language of s-t tgds is not closed under composition

SO tgds [FKPT05]

well-behaved fragment of second-order logic that extends s-t tgds with Skolem functions.

SO-tgds: Definition

Let $\mathbf{S}$ be a source schema and $\mathbf{T}$ be a target schema

A **second-order tuple-generating dependency (SO-tgd)** is a formula of the form

$$\exists \mathbf{f_1} \ldots \exists \mathbf{f_m} (\forall \mathbf{x_1} (\varphi_1 \to \psi_1)) \wedge \ldots \wedge (\forall \mathbf{x_n} (\varphi_n \to \psi_n)), \text{ where}$$

- Each $\mathbf{f_i}$ is a function symbol
- Each φ_i is a conjunction of atoms from $\mathbf{S}$ and equalities over terms
- Each ψ_i is a conjunction of atoms from $\mathbf{T}$

Some Results [FKPT05]

Closed under Composition:

- The composition of two SO-tgds is definable by a SO-tgd
- Every SO tgd is the

  composition of finitely many finite sets of s-t tgds.
- Hence, SO tgds are the **“right”** language for the composition of s-t tgds

Example [FKPT05]

Σ_{12} :

$$\forall e\,(\texttt{Emp}(e) \rightarrow \exists m\,\texttt{Mgr}_1(e,m)))$$

Σ_{23} :

$$\forall e \forall m\,(\texttt{Mgr}_1(e,m) \rightarrow \texttt{Mgr}(e,m))$$
$$\forall e\,(\texttt{Mgr}_1(e,e) \rightarrow \texttt{SelfMgr}(e)))$$

$$\exists \mathbf{f}\,(\forall e(\texttt{Emp}(e) \ \rightarrow \ \texttt{Mgr}(e,\mathbf{f}(\mathbf{e}))) \wedge$$
$$\forall e\,(\texttt{Emp}(e) \ \wedge \ (\mathbf{e} = \mathbf{f}(\mathbf{e})) \rightarrow \texttt{SelfMgr}(e)))$$

Beyond

source to target

&

Back to FO

[Nash, Bernstein & Melnik 05]

Main Features

Prev. Work [FKPT 05]:

tgds & SO tgds.

Both source to target

SOtgds as a result of the composition

Motivation

Allow Schema Constraints

Deployment of composition in current DB systems

Mapping Languages

($\forall CQ_0^=$) **FullD-mappings**

Given by Full Dependencies

($\forall CQ^=$) **ED-mappings**

Given by Embedded Dependencies

($Sk\forall CQ^=$) **SkED-mappings**

Given by Second-Order Constraints

Without equality:

($\forall CQ_0$) **FullTGD**

($\forall CQ$) **TGD**

Composing Embedded Dependencies

1. Skolemize ED-mappings to get SkED-mappings;
2. SKED-axiomatization of all the SkED constraints that hold in the composition;
3. de-Skolemize the SKED-axiomatization to get a **ED-mapping**

A difference:

The composition in [FKPT 05] is given by second-order constraints

Basic Questions

1. Is $\mathcal{L}$ closed under composition?
2. **If not:**

   Is there a decision procedure to determine whether

   the composition of two $\mathcal{L}$-mappings is a $\mathcal{L}$-mapping?

Note:

Whenever a result holds for a class without equality
it also holds for the corresponding class with equality

Full Dependencies

Definability & Closure:

There are $\forall$CQ$_0$-mappings whose composition is not an FO-mapping.
In particular, $\forall$CQ$_0$ is not closed under composition

$$\begin{array}{lr} \Sigma_{12} \text{ is} & R(x,y) \rightarrow S(x,y) \\ & S(x,y), S(y,z) \rightarrow S(x,y) \\ \Sigma_{23} \text{ is} & S(x,y) \rightarrow T(x,y) \end{array}$$

$$R(x,v_1), R(v_1,v_2), \ldots, R(v_{i-1},v_i), R(v_i,y) \rightarrow T(x,y)$$

No finite set expresses: $tc(R) \subseteq T$

Full Dependencies

Undecidability:

Checking whether the composition of two $\forall CQ_0$-mappings is a $\forall CQ_0$-mapping is undecidable. In fact, coRE-hard

Reduction from the Post Correspondence Problem

Full Dependencies: Other Results

1. Necessary and sufficient (but uncomputable) conditions for composition of FullTGDs (the same for $\forall CQ^{=}$).
2. Algorithms that compute the composition of FullTGD-mappings when these conditions are satisfied.
3. Definition of sub-classes of $\forall CQ_0$ and $\forall CQ_0^{=}$ that are closed under composition.

Full Dependencies: A Main Theorem

Theorem 1: If the $\forall\mathsf{CQ}_0^=$-mappings M_{12}, M_{13} are given by $(\mathbf{S}_1, \mathbf{S}_2, \Sigma_{12})$ and $(\mathbf{S}_2, \mathbf{S}_3, \Sigma_{23})$ with $\Sigma_{123} := \Sigma_{12} \cup \Sigma_{23}$ and $\mathbf{S}_{13} = \mathbf{S}_1 \cup \mathbf{S}_3$, then the following are equivalent:

1. There is a finite set of constraints $\Sigma_{13} \subseteq \forall\mathsf{CQ}_0^=$ over the signature $\mathbf{S}_{13}$ s.t. $M := M_{12} \circ M_{13}$ is given by $(\mathbf{S}_1, \mathbf{S}_3, \Sigma_{13})$.

2. There is a finite set of constraints $\Sigma_{13} \subseteq \forall\mathsf{CQ}_0^=$ over the signature $\mathbf{S}_{13}$ s.t.

$$\mathsf{DC}(\forall\mathsf{CQ}_0^=, \Sigma_{123})|_{\mathbf{S}_{13}} = \mathsf{DC}(\forall\mathsf{CQ}_0^=, \Sigma_{13})$$

3. There is a k s.t. for every ξ over $\mathbf{S}_{13}$ satisfying $\Sigma_{123} \vdash \xi$ there is a deduction of ξ from Σ_{123} using at most k $\mathbf{S}_2$-resolutions.

Full Dependencies: Composition

Procedure: FullD-COMPOSE $(\Sigma_{12}, \Sigma_{23})$, when it terminates, computes the deductive closure of $\Sigma_{12} \cup \Sigma_{23}$ then, restrict to constraints not refering to $\mathbf{S}_2$

Correctness:

Under the hypotheses of Theorem 1, FULLD-COMPOSE $(\Sigma_{12}, \Sigma_{23})$, whenever it terminates, yields Σ_{13} s.t $M_{12} \circ M_{23}$ is given by

$$(\mathbf{S}_1, \mathbf{S}_3, \Sigma_{13})$$

Size?

FullDCOMPOSE may produce a result that is exponential in the size of the input

$$\Sigma_{12} \text{ is } \quad \begin{array}{r} R(x,y), R(y,x) \rightarrow S(x,y) \\ R(x,y), R(x,x), \rightarrow S(x,y) \end{array}$$

$$\Sigma_{23}^{k} \text{ is } S(x,u_1),\ldots,S(u_{k-1},y) \rightarrow T(x,y)$$

For each $S(u,v)$, we can substitute either

$$R(u,v), R(v,u) \text{ or } R(u,v), R(v,v)$$

Then, 2^k constraints in the composition $M_{12} \circ M_{23}^{k}$.

FULLDCOMPOSE:Termination

Σ_{12} is
$$R(x,y) \rightarrow S(x,y)$$
$$S(x,y), S(y,z) \rightarrow S(x,y)$$
$$R(x,y), R(y,z) \rightarrow R(x,z)$$

Σ_{23} is
$$S(x,y) \rightarrow T(x,y)$$

FULL Dependency:
$$R(x,y), R(y,z) \rightarrow R(x,z)$$
$$R(x,y) \rightarrow T(x,y)$$

Termination: If "non-trivial" recursion over atoms in $\mathbf{S}_2$ is disallowed, then FULLD-COMPOSE$(\Sigma_{12}, \Sigma_{23})$ terminates and therefore $M_{12} \circ M_{23}$ is a FULLD-mapping

Second-Order Dependencies

Why?:

Handle existential quantifiers in a ED-dependency,
first convert ED constraints into SKED constraints

Composition:

Necessary and sufficient (but uncomputable) conditions
similar to the ones for FULLD-mappings

SKCOMPOSE:

Analogous to FULLDCOMPOSE but operating on SkED constraints

Embedded Dependencies

Procedure ED-COMPOSE $(\Sigma_{12}, \Sigma_{23})$

1. Σ'_{12} := SKOLEMIZE (Σ_{12})
   Σ'_{23} := SKOLEMIZE (Σ_{23})
2. Σ'_{13} := **SkED** -COMPOSE$(\Sigma'_{12}, \Sigma'_{23})$
3. Return DE-SKOLEMIZE (Σ'_{13})

DE-SKOLEMIZE

Intuition:

1. Put constraints in the input in to a for where they are the obvious result of Skolemization. Some steps:
    - Check for cycles
    - Check for repeated function symbols
    - Align variables
2. Reverse the Skolemization in the obvious way.
    - Combine Dependencies
    - Function symbols are actually replaced by existentially variables

Some Results

Theorem: If DE-SKOLEMIZE (Σ) succeeds on input $\Sigma \subseteq$ SkED giving Σ', then

$$\Sigma' \subseteq \text{ ED and } \Sigma' \equiv \Sigma$$

Theorem: **DE-SKOLEMIZE** may produce a result that is exponential in the size of the input

Why?

Combine Dependencies

Combine Dependencies

Input: $R(x,y) \to S(x, f(x,y)), S(f(x,y), y)$

$$R(x,y) \to \exists u\, S(x,u), S(u,v)$$

$$\begin{array}{rcl} R(x,y), u = f(x,y) & \to & S(x,u) \\ R(x,y), u = f(x,y) & \to & S(u,y) \end{array}$$

$1st$ **attemp** $\begin{array}{rcl} R(x,y) & \to & \exists u S(x,u) \\ R(x,y) & \to & \exists u S(u,y) \end{array}$

Combine Dependencies

Input: $R(x,y) \to S(x, f(x,y)), S(f(x,y), y)$

$$R(x,y) \to \exists u\, S(x,u), S(u,v)$$

$$\begin{array}{lcl} R(x,y), \mathbf{u} = \mathbf{f}(\mathbf{x},\mathbf{y}) & \to & S(x,u) \\ R(x,y), \mathbf{u} = \mathbf{f}(\mathbf{x},\mathbf{y}) & \to & S(u,y) \end{array}$$

$2nd$ **attemp**

$$\begin{array}{lcl} R(x,y), \mathbf{u} = \mathbf{f}(\mathbf{x},\mathbf{y}) & \to & S(x,u) \\ R(x,y), \mathbf{u} = \mathbf{f}(\mathbf{x},\mathbf{y}) & \to & S(u,y) \\ R(x,y), \mathbf{u} = \mathbf{f}(\mathbf{x},\mathbf{y}) & \to & S(x,u), S(u,y) \end{array}$$

Exponential unavoidable

Theorem $\exists$ sequences of TGD-mappings M^k_{12} and M^k_{23} given Σ^k_{12} and Σ^k_{23} s.t.

- TGD-composition $M^k_{12} \circ M^k_{23}$ grows exponentially
- SkTGD-composition $M^k_{12} \circ M^k_{23}$ grows linearly

in the size of $\Sigma^k_{12} \cup \Sigma^k_{13}$

Proof

Σ_{12} is $\quad R_0(x) \to \exists y\, S_0(x,y)$

$\quad R_i(x) \to S_i(x)$

Σ_{23} is $\quad S_0(x,y), S_i(x) \to T_i(y)$

SKTGD-composition $M_{13}^k := M_{12}^k \circ M_{23}^k$. Given by Σ_{13}^k

$$R_0(x), \mathbf{y} = \mathbf{f}(\mathbf{x}), R_i(x) \to T_i(y)$$

TGD-composition: DESKOLEMIZE(Σ_{13}^k). Given by Σ'^k_{13}

$$R_0(x), R_Z(x) \to \exists y T_Z(y)$$

where $R_Z(x) := \wedge_{i \in Z} R_i(x)$

We can not do better

M_{13}^k cannot be expressed by any $(\mathbf{S}_1, \mathbf{S}_3, \Sigma)$ $\Sigma \subseteq TGD$ with $\mid \Sigma \mid < 2^{k-1}$

Inexpressibility tool

Characterize constraints in terms of monotonicity

- Consider Σ over σ and A_0 over σ. $A_0 \models \Sigma$
- Add more tuples to some relation in $A_0 \rightsquigarrow A_1$
- Truth value flips or stay the same
- Keep adding tuples $A_0, \ldots, A_n, \ldots$
- The truth values of Σ form segments: Positive and Negative

- Example: (true, true, false, false, true) for a chain of structure $(A_0, A_1, A_2, A_3, A_4)$
- To **Characterize** Σ, count the maximal number of negativ segments in any chain.
- If the number is finite, Σ is n *monotonic* and *nonmonotonic* othw.

Charaterize a class of constraints, we study the monotonocity properties of its constituent sentences

Example: $\Sigma = \{R(x) \rightarrow \exists y\, S(y)\}$ is 1-monotonic

- $R \neq \emptyset \rightarrow S \rightarrow S \neq \emptyset$
- $(\emptyset, \emptyset), (R, \emptyset), (R_1, S)$
- $(R, \emptyset)$ belongs to the only negative segment

source to target

but

with Target Constraints

[Arenas, Fagin & Nash 10]

Composition: Back to the standard setting?

Back to Standard Mappings:
Schema mapping $M = (\mathbf{S}, \mathbf{T}, \Sigma_{st} \cup \Sigma_t)$ such that

- Σ_{st} is a set of s-t tgds
- Σ_t is a set of target tgds and target egds

Target tgds:
In particular, weakly acyclic t-tgds

What is the right language to express the composition of standard schemma mappings?

SO Tgds:
Is the language of SO tgds the right one to compose standard schema mappings?

SO tgds are NOT enough

Let $M_{12} = (\mathbf{S}_1, \mathbf{S}_2, \Sigma_{12}, \Sigma_2)$ and $M_{23} = (\mathbf{S}_2, \mathbf{S}_3, \Sigma_{23})$,

$$\begin{aligned}\Sigma_{12} &= \{P(x,y) \to R(x,y)\} \\ \Sigma_2 &= \{R(x,y) \wedge R(x,z) \to y = z\} \\ \Sigma_{23} &= \{R(x,y) \to T(x,y)\}\end{aligned}$$

$P^{I_1} = \{(1,2),(1,3)\}$, $\nexists\, I_3$ of $\mathbf{S}_3$ s.t. $(I_1, I_3) \in M_{12} \circ M_{23}$,

I_1 does not have any solutions under M_{12}.

Extra help

source & target constraints

$$\Sigma_1 = \{P(x,y) \wedge P(x,z) \rightarrow y = z\}$$
$$\Sigma_{13} = \{P(x,y) \rightarrow T(x,y)\}$$

Is the language of SO tgds + s & t-constraints is the right language?

Theorem: There are standard mappings M_{12} and M_{23} s.t $M_{12} \circ M_{23}$ cannot by specified by an SO tgd, an arbitrary set of target constraints and an arbitrary set of source constraints

Proof: Notion of Locality

Reminder:

- Notions of locality have been used to prove inexpressibility results for FO.
- FO logic cannot express properties that involve no trivial recursive computations

Standard Steps

- Provide a Notion of Locality:
  Notion of Locality for Data Transformation [ABFL 04]
- For every st-gd mapping, the canonical transformation is local [ABFL 04]
- The composition is not local

source-to-target SO schema mappings

An extension SO tgds:

st SO dependency extend SO tgds by allowing equalities in the conclusions

SO standard Mapping:

A schema mapping where the constraints consists of

- A st SO tgd
- A set of target tgds and target egds

SO standard schema mappings is the right language

Theorem 2:

1. The composition of two standard SO schema mappings is equivalent a standard schema mapping
2. The composition of a finite number standard SO schema mappings is equivalent a standard schema mapping
3. Every standard SO schema mappings is equivalent to the composition of finete number of standard schema mappings

Key for 1. To simulate the atomic formula $C(x, y)$ introduce the equality $f_C(x, y) = g_C(x, y)$

Nested Terms

SO tgds and st SO dependencies can have nested terms.
These can be difficult to work with and understand

Example:

$$f(g(x), h(f(x,y))) = g(f(x, h(y)))$$

premise of a SO tgd or in the premise/conclusion of a st SO dependecy

Unnested

It is better to work with unnested SO tgds and unnested st SO dependencies

Obvious way to DENEST doesn't work

Nested SO tgd

$$\exists f \exists g \forall (x) \forall (y) (P(x,y)) \wedge (f(g(x) = y)) \rightarrow Q(f(x), g(y)))$$

Obvious way to Denest

$$\exists f \exists g \forall (x) \forall (y) \forall (z) ((P(x,y)) \wedge (g(x) = z)) \wedge (f(z) = y) \rightarrow Q(f(x), g(y)))$$

Unsafe

The variable z does not appear in an atomic formula in the premise

Denesting Results

Theorem:
Every st-SO dependency is equivalent to an unnested st-SO dependency

Theorem:
Every SO tgd is equivalent to an unnested SO tgd

Collapsing Results: The composition of a finite number of st tgd mappings is equivalent to the composition of two st tgd mappings

- The composition is specified by an SO tgd
- Such SO tgd is equivalent to an unnested one
- Lemma [FKPT 05]

  Every schema mapping specified by an SO tgd of depth r is equivalent to a composition of $r+1$ st tgds

CHASE for ST-SO Dependencies

Chasable:

st SO schema mappings have a chase that terminates in Polynomial time

Challenge: While computing the solution this chase needs to keep track of constantly changing values of functions

Previous Work

Two terms are treated as equal if they are sintactically identical.

Example: A premise containing the atom $f(x) = g(y)$

Now: SO egd part may force $f(0)$ and $g(1)$ to be equal

References

[**ABFL 04**] M. Arenas, P. Barceló, R. Fagin, and L. Libkin. Locally Consistent Transformations and Query Answering in Data Exchange. *In Proceedings of the 23rd ACM Symposium on Principles of Database Systems, PODS04*, pages 229-240, 2004.

[**FKPT 05**] R. Fagin, P. G. Kolaitis, L. Popa, and W. C. Tan. Composing schema mappings: Second-order dependencies to the rescue. *ACM Trans. Database Syst.*, 30(4):994-1055, 2005.

[**NBM 05**] A. Nash, P. A. Bernstein, and S. Melnik. Composition of Mappings Given by Embedded Dependencies. In *Proceedings of the 24th ACM Symposium on Principles of Database Systems, PODS05*, pages 172-183, 2005.

Thank You!