Data Exchange beyond Complete Data

Marcelo Arenas

Department of Computer Science
Pontificia Universidad Católica de Chile

Joint work with Jorge Pérez (U. de Chile) and Juan Reutter (U. Edinburgh)

Outline

- The need for a more general data exchange framework
  - Two important scenarios: Incomplete databases and knowledge bases
- Formalism for exchanging representations systems
- Applications to incomplete databases
- Applications to metadata management
- Concluding remarks

Outline

- The need for a more general data exchange framework
  - Two important scenarios: Incomplete databases and knowledge bases
- Formalism for exchanging representations systems
- Applications to incomplete databases
- Applications to metadata management
- Concluding remarks

Data exchange: Some lessons learned

Key steps in the development of the area:

- Definition of schema mapping: Precise syntax and semantics
- Definition of the notion of solution
- Identification of good solutions
  - Universal solutions
- Polynomial time algorithms for materializing good solutions
  - Based on the chase procedure

Data exchange: Some lessons learned

Key steps in the development of the area:

- Definition of schema mapping: Precise syntax and semantics
- Definition of the notion of solution
- Identification of good solutions
  - Universal solutions
- Polynomial time algorithms for materializing good solutions
  - Based on the chase procedure

Creating schema mappings is a time consuming and expensive process

- Manual or semi-automatic process in general

Ongoing project: Reusing schema mappings

$$\mathbf{S} \xrightarrow[\Sigma_{\mathbf{ST}}]{} \mathbf{T}$$

Ongoing project: Reusing schema mappings

$$\mathbf{S} \xrightarrow[\Sigma_{\mathbf{ST}}]{} \mathbf{T} \xrightarrow[\Sigma_{\mathbf{TU}}]{} \mathbf{U}$$

Ongoing project: Reusing schema mappings

$$
\begin{tikzcd}
\mathbf{S} \arrow[r, "\Sigma_{\mathbf{ST}}"'] \arrow[rr, bend right, "\Sigma_{\mathbf{SU}}\,?"'] & \mathbf{T} \arrow[r, "\Sigma_{\mathbf{TU}}"'] & \mathbf{U}
\end{tikzcd}
$$

Ongoing project: Reusing schema mappings

$$
\begin{tikzcd}[column sep=huge]
\mathbf{S} \arrow[r, "\Sigma_{\mathbf{ST}}"'] \arrow[rr, bend right, "\Sigma_{\mathbf{SU}}\,?"'] & \mathbf{T} \arrow[r, "\Sigma_{\mathbf{TU}}"'] & \mathbf{U}
\end{tikzcd}
$$

We need some operators for schema mappings

Ongoing project: Reusing schema mappings

$$
\begin{tikzcd}
\mathbf{S} \arrow[r, "\Sigma_{\mathbf{ST}}"'] \arrow[rr, bend right] & \mathbf{T} \arrow[r, "\Sigma_{\mathbf{TU}}"'] & \mathbf{U}
\end{tikzcd}
$$

$$\Sigma_{\mathbf{SU}} = \Sigma_{\mathbf{ST}} \circ \Sigma_{\mathbf{TU}}$$

We need some operators for schema mappings

- **Composition** in the above case

Metadata management

This has motivated the need for the development of a general infrastructure for managing schema mappings.

The problem of managing schema mappings is called **metadata management.**

High-level algebraic operators, such as compose, are used to manipulate mappings.

- What other operators are needed?

An inverse operator is also needed

$$\mathbf{S} \xrightarrow[\Sigma_{\mathbf{ST}}]{} \mathbf{T}$$

An inverse operator is also needed

$$
\begin{tikzcd}
\mathbf{S} \arrow[r, "\Sigma_{\mathbf{ST}}"'] \arrow[d, "\Sigma_{\mathbf{SV}}"] & \mathbf{T} \\
\mathbf{V} &
\end{tikzcd}
$$

An inverse operator is also needed

$$
\begin{tikzcd}
\mathbf{S} \arrow[r, "\Sigma_{\mathbf{ST}}"'] \arrow[d, "\Sigma_{\mathbf{SV}}"] & \mathbf{T} \\
\mathbf{V} \arrow[u, bend left, "\Sigma_{\mathbf{VS}}?"]
\end{tikzcd}
$$

An inverse operator is also needed

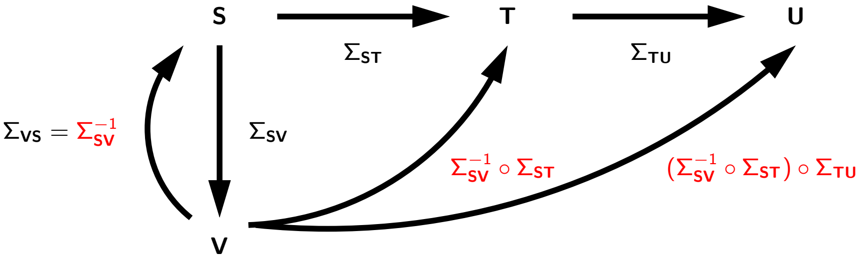
$$\begin{tikzcd}[column sep=huge, row sep=huge]
\mathbf{S} \arrow[r, "\Sigma_{\mathbf{ST}}"'] \arrow[d, "\Sigma_{\mathbf{SV}}"] & \mathbf{T} \\
\mathbf{V} \arrow[u, bend left, "\Sigma_{\mathbf{VS}} = \Sigma_{\mathbf{SV}}^{-1}"]
\end{tikzcd}$$

An inverse operator is also needed

$$
\begin{tikzcd}
\mathbf{S} \arrow[r, "\Sigma_{\mathbf{ST}}"'] \arrow[d, "\Sigma_{\mathbf{SV}}"] & \mathbf{T} \\
\mathbf{V} \arrow[u, bend left, "\Sigma_{\mathbf{VS}} = \Sigma_{\mathbf{SV}}^{-1}"] \arrow[ur, bend right, "\Sigma_{\mathbf{SV}}^{-1} \circ \Sigma_{\mathbf{ST}}"'] &
\end{tikzcd}
$$

Composition and inverse operators have to be combined

An inverse operator is also needed



Composition and inverse operators have to be combined

Metadata management: A more general data exchange framework is needed

Composition and inverse operators have been extensively studied in the relational world.

- Semantics, computation, ...

Combining these operators is an open issue.

Metadata management: A more general data exchange framework is needed

Composition and inverse operators have been extensively studied in the relational world.

- Semantics, computation, . . .

Combining these operators is an open issue.

- Key observation: A target instance of a mapping can be the source instance of another mapping

Metadata management: A more general data exchange framework is needed

Composition and inverse operators have been extensively studied in the relational world.

- Semantics, computation, ...

Combining these operators is an open issue.

- Key observation: A target instance of a mapping can be the source instance of another mapping
- Sources instances may contain null values

Metadata management: A more general data exchange framework is needed

Composition and inverse operators have been extensively studied in the relational world.

- Semantics, computation, ...

Combining these operators is an open issue.

- Key observation: A target instance of a mapping can be the source instance of another mapping
- Sources instances may contain null values

There is a need for a data exchange framework that can handle databases with incomplete information.

Data exchange in the RDF world

There is an increasing interest in publishing relational data as RDF

- Resulted in the creation of the W3C RDB2RDF Working Group

The problem of translating relational data into RDF can be seen as a data exchange problem

- Schema mappings can be used to describe how the relational data is to be mapped into RDF

Data exchange in the RDF world

There is an increasing interest in publishing relational data as RDF

- Resulted in the creation of the W3C RDB2RDF Working Group

The problem of translating relational data into RDF can be seen as a data exchange problem

- Schema mappings can be used to describe how the relational data is to be mapped into RDF

But there is a mismatch here: A relational database under a closed-world semantics is to be translated into an RDF graph under an open-world semantics

- There is a need for a data exchange framework that can handle both databases with complete and incomplete information

Data exchange in the RDF world

An issue discussed at the W3C RDB2RDF Working Group: Is a mapping information preserving?

- In particular: For the default mapping defined by this group

How can we address this issue?

- Metadata management can help us

Data exchange in the RDF world

An issue discussed at the W3C RDB2RDF Working Group: Is a mapping information preserving?

- In particular: For the default mapping defined by this group

How can we address this issue?

- Metadata management can help us

Question to answer: Is a mapping invertible?

Data exchange in the RDF world

An issue discussed at the W3C RDB2RDF Working Group: Is a mapping information preserving?

- In particular: For the default mapping defined by this group

How can we address this issue?

- Metadata management can help us

Question to answer: Is a mapping invertible?

- This time an RDF graph is to be translated into a relational database!

Data exchange in the RDF world

An issue discussed at the W3C RDB2RDF Working Group: Is a mapping information preserving?

- In particular: For the default mapping defined by this group

How can we address this issue?

- Metadata management can help us

Question to answer: Is a mapping invertible?

- This time an RDF graph is to be translated into a relational database!
- We want to have a unifying framework for all these cases

But these are not the only reasons . . .

Nowadays several applications use knowledge bases to represent data.

- A knowledge base has not only data but also rules that allows to infer new data
- In the Semantics Web: RDFS and OWL ontologies

But these are not the only reasons . . .

Nowadays several applications use knowledge bases to represent data.

- A knowledge base has not only data but also rules that allows to infer new data
- In the Semantics Web: RDFS and OWL ontologies

In a data exchange application over the Semantics Web:

> The input is a mapping and a source specification including data and rules, and the output is a target specification also including data and rules

But these are not the only reasons . . .

Nowadays several applications use knowledge bases to represent data.

- A knowledge base has not only data but also rules that allows to infer new data
- In the Semantics Web: RDFS and OWL ontologies

In a data exchange application over the Semantics Web:

> The input is a mapping and a source specification including data and rules, and the output is a target specification also including data and rules

There is a need for a data exchange framework that can handle knowledge bases.

One can exchange more than complete data

- In data exchange one starts with a database instance (with complete information).
- What if we have an initial object that has several interpretations?
  - A representation of a set of possible instances
- We propose a new general formalism to exchange representations of possible instances
  - We apply it to the problems of exchanging instances with incomplete information and exchanging knowledge bases

Outline

- The need for a more general data exchange framework
  - Two important scenarios: Incomplete databases and knowledge bases
- Formalism for exchanging representations systems
- Applications to incomplete databases
- Applications to metadata management
- Concluding remarks

Representation systems

A representation system $\mathcal{R} = (\mathbf{W}, \mathrm{rep})$ consists of:

- a set $\mathbf{W}$ of *representatives*
- a function rep that assigns a set of instances to every element in $\mathbf{W}$

$$\mathrm{rep}(\mathcal{V}) = \{I_1, I_2, I_3, \ldots\} \text{ for every } \mathcal{V} \in \mathbf{W}$$

Uniformity assumption: For every $\mathcal{V} \in \mathbf{W}$, there exists a relational schema $\mathbf{S}$ (the type of $\mathcal{V}$) such that $\mathrm{rep}(\mathcal{V}) \subseteq \mathrm{Inst}(\mathbf{S})$

Representation systems

A representation system $\mathcal{R} = (\mathbf{W}, \text{rep})$ consists of:

- a set **W** of *representatives*
- a function rep that assigns a set of instances to every element in **W**

$$\text{rep}(\mathcal{V}) = \{I_1, I_2, I_3, \ldots\} \text{ for every } \mathcal{V} \in \mathbf{W}$$

Uniformity assumption: For every $\mathcal{V} \in \mathbf{W}$, there exists a relational schema **S** (the type of $\mathcal{V}$) such that $\text{rep}(\mathcal{V}) \subseteq \text{Inst}(\mathbf{S})$

Incomplete instances and knowledge bases are representation systems

In classical data exchange we consider only *complete* data

$\mathcal{M}$ is a mapping from **S** to **T** if $\mathcal{M} \subseteq \text{Inst}(\mathbf{S}) \times \text{Inst}(\mathbf{T})$

- Given instances I of **S** and J of **T**: J is a solution for I under $\mathcal{M}$ if **S** if $(I, J) \in \mathcal{M}$

In classical data exchange we consider only *complete* data

$\mathcal{M}$ is a mapping from $\mathbf{S}$ to $\mathbf{T}$ if $\mathcal{M} \subseteq \text{Inst}(\mathbf{S}) \times \text{Inst}(\mathbf{T})$

- Given instances I of $\mathbf{S}$ and J of $\mathbf{T}$: J is a solution for I under $\mathcal{M}$ if $\mathbf{S}$ if $(I, J) \in \mathcal{M}$

$\mathcal{M}$ is defined by a set Σ of dependencies (e.g., st-tgds) if: $(I, J) \in \mathcal{M}$ iff $(I, J) \models \Sigma$.

- Notation: $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$

Extending the definition to representation systems

$\text{Sol}_{\mathcal{M}}(I)$: Set of solutions for I under $\mathcal{M}$

Extending the definition to representation systems

$\text{Sol}_{\mathcal{M}}(I)$: Set of solutions for I under $\mathcal{M}$

This can be extended to set of instances. Given $\mathcal{X} \subseteq \text{Inst}(\mathbf{S})$:

$$\text{Sol}_{\mathcal{M}}(\mathcal{X}) \;=\; \bigcup_{I \in \mathcal{X}} \text{Sol}_{\mathcal{M}}(I)$$

Extending the definition to representation systems

Given:

- a mapping $\mathcal{M}$ from $\mathbf{S}$ to $\mathbf{T}$
- a representation system $\mathcal{R} = (\mathbf{W}, \text{rep})$
- $\mathcal{U}, \mathcal{V} \in \mathbf{W}$ of types $\mathbf{S}$ and $\mathbf{T}$, respectively

Extending the definition to representation systems

Given:

- a mapping $\mathcal{M}$ from **S** to **T**
- a representation system $\mathcal{R} = (\mathbf{W}, \text{rep})$
- $\mathcal{U}, \mathcal{V} \in$ **W** of types **S** and **T**, respectively

Definition (APR11)

$\mathcal{V}$ is an $\mathcal{R}$*-solution* of $\mathcal{U}$ under $\mathcal{M}$ if

$$\text{rep}(\mathcal{V}) \quad \subseteq \quad \text{Sol}_{\mathcal{M}}(\text{rep}(\mathcal{U}))$$

Extending the definition to representation systems

Given:

- a mapping $\mathcal{M}$ from **S** to **T**
- a representation system $\mathcal{R} = (\mathbf{W}, \text{rep})$
- $\mathcal{U}, \mathcal{V} \in \mathbf{W}$ of types **S** and **T**, respectively

Definition (APR11)

$\mathcal{V}$ is an $\mathcal{R}$-*solution* of $\mathcal{U}$ under $\mathcal{M}$ if

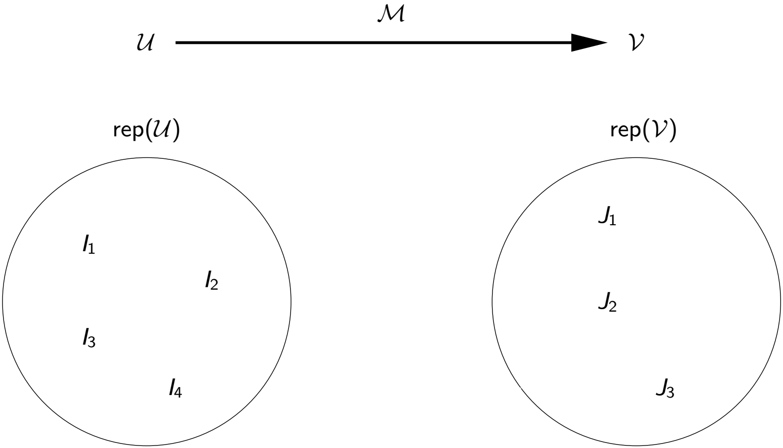
$$\text{rep}(\mathcal{V}) \quad \subseteq \quad \text{Sol}_{\mathcal{M}}(\text{rep}(\mathcal{U}))$$

Or equivalently: $\mathcal{V}$ is an $\mathcal{R}$-*solution* of $\mathcal{U}$ if for every $J \in \text{rep}(\mathcal{V})$, there exists $I \in \text{rep}(\mathcal{U})$ such that $J \in \text{Sol}_{\mathcal{M}}(I)$.

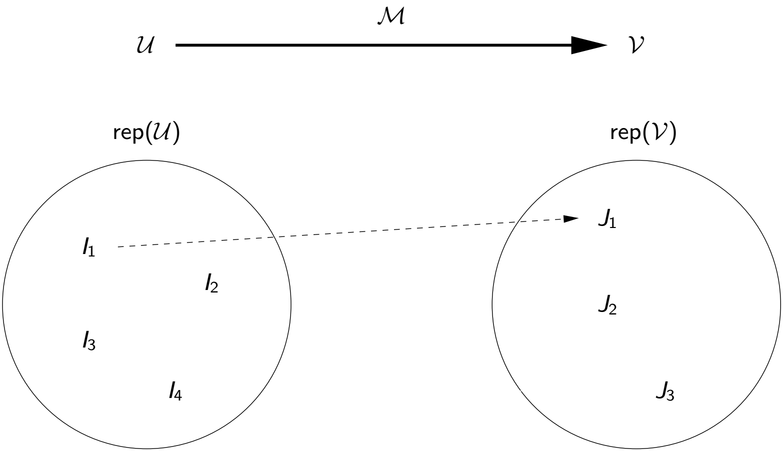
Extending the definition to representation systems

$$\mathcal{U} \xrightarrow{\mathcal{M}} \mathcal{V}$$

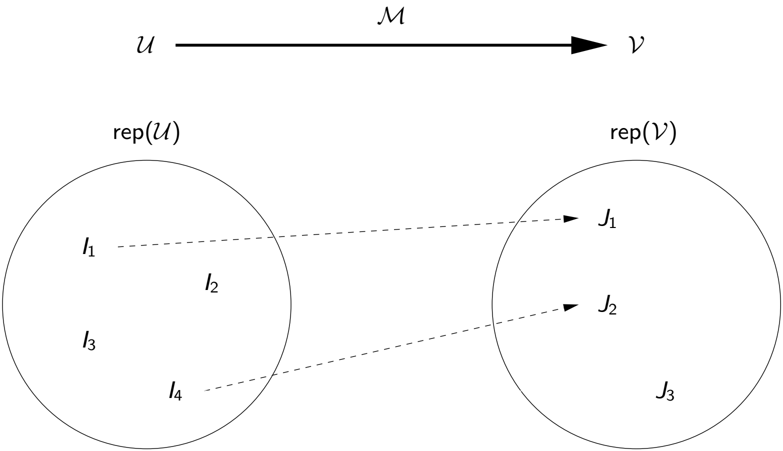
Extending the definition to representation systems



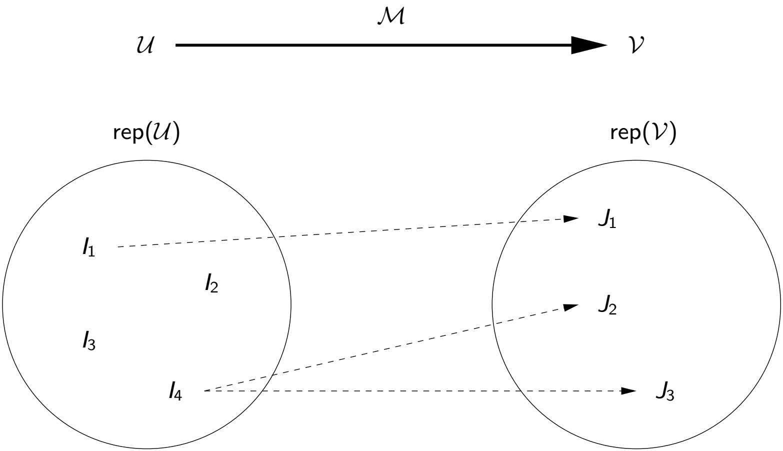
Extending the definition to representation systems



Extending the definition to representation systems



Extending the definition to representation systems



Universal solutions

What is a good solution in this framework?

Universal solutions

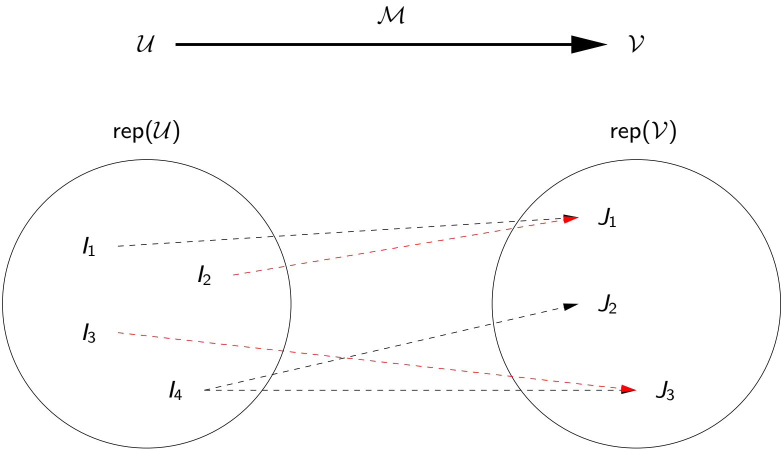
What is a good solution in this framework?

Definition (APR11)

$\mathcal{V}$ is an *universal $\mathcal{R}$-solution* of $\mathcal{U}$ under $\mathcal{M}$ if

$$\text{rep}(\mathcal{V}) \quad = \quad \text{Sol}_{\mathcal{M}}(\text{rep}(\mathcal{U}))$$

Universal solutions in a figure



Strong representation systems

Let $\mathcal{C}$ be a class of mappings.

Strong representation systems

Let $\mathcal{C}$ be a class of mappings.

Definition (APR11)

$\mathcal{R} = (\mathbf{W}, \text{rep})$ is a *strong representation system* for $\mathcal{C}$ if for every $\mathcal{M} \in \mathcal{C}$ and for every $\mathcal{U} \in \mathbf{W}$, there exists a $\mathcal{V} \in \mathbf{W}$:

$$\text{rep}(\mathcal{V}) \quad = \quad \text{Sol}_{\mathcal{M}}(\text{rep}(\mathcal{U}))$$

Strong representation systems

Let $\mathcal{C}$ be a class of mappings.

Definition (APR11)

$\mathcal{R} = (\mathbf{W}, \text{rep})$ is a *strong representation system* for $\mathcal{C}$ if for every $\mathcal{M} \in \mathcal{C}$ from **S** to **T**, and for every $\mathcal{U} \in \mathbf{W}$, there exists a $\mathcal{V} \in \mathbf{W}$:

$$\text{rep}(\mathcal{V}) \quad = \quad \text{Sol}_{\mathcal{M}}(\text{rep}(\mathcal{U}))$$

Strong representation systems

Let $\mathcal{C}$ be a class of mappings.

Definition (APR11)

$\mathcal{R} = (\mathbf{W}, \mathrm{rep})$ is a *strong representation system* for $\mathcal{C}$ if for every $\mathcal{M} \in \mathcal{C}$ from $\mathbf{S}$ to $\mathbf{T}$, and for every $\mathcal{U} \in \mathbf{W}$ of type $\mathbf{S}$, there exists a $\mathcal{V} \in \mathbf{W}$:

$$\mathrm{rep}(\mathcal{V}) \quad = \quad \mathrm{Sol}_{\mathcal{M}}(\mathrm{rep}(\mathcal{U}))$$

Strong representation systems

Let $\mathcal{C}$ be a class of mappings.

Definition (APR11)

$\mathcal{R} = (\mathbf{W}, \text{rep})$ is a *strong representation system* for $\mathcal{C}$ if for every $\mathcal{M} \in \mathcal{C}$ from $\mathbf{S}$ to $\mathbf{T}$, and for every $\mathcal{U} \in \mathbf{W}$ of type $\mathbf{S}$, there exists a $\mathcal{V} \in \mathbf{W}$ of type $\mathbf{T}$:

$$\text{rep}(\mathcal{V}) \quad = \quad \text{Sol}_{\mathcal{M}}(\text{rep}(\mathcal{U}))$$

Strong representation systems

Let $\mathcal{C}$ be a class of mappings.

Definition (APR11)

$\mathcal{R} = (\mathbf{W}, \text{rep})$ is a *strong representation system* for $\mathcal{C}$ if for every $\mathcal{M} \in \mathcal{C}$ from **S** to **T**, and for every $\mathcal{U} \in \mathbf{W}$ of type **S**, there exists a $\mathcal{V} \in \mathbf{W}$ of type **T**:

$$\text{rep}(\mathcal{V}) \quad = \quad \text{Sol}_{\mathcal{M}}(\text{rep}(\mathcal{U}))$$

If $\mathcal{R} = (\mathbf{W}, \text{rep})$ is a strong representation system, then the universal solutions for the representatives in **W** can be represented in the same system.

Outline

- The need for a more general data exchange framework
  - Two important scenarios: Incomplete databases and knowledge bases
- Formalism for exchanging representations systems
- Applications to incomplete databases
- Applications to metadata management
- Concluding remarks

Motivating questions

What is a strong representation system for the class of mappings specified by st-tgds?

- Are instances including nulls enough?

Can the fundamental data exchange problems be solved in polynomial time in this setting?

- Computing (universal) solutions

Naive instances

We have already considered naive instances: Instances with null values

- Example: Universal solutions

A naive instance $\mathcal{I}$ has labeled nulls:

$$
\begin{array}{c}
R(1, n_1) \\
R(n_1, 2) \\
R(1, n_2)
\end{array}
$$

Naive instances

We have already considered naive instances: Instances with null values

- Example: Universal solutions

A naive instance $\mathcal{I}$ has labeled nulls:

$$R(1, n_1)$$
$$R(n_1, 2)$$
$$R(1, n_2)$$

The interpretations of $\mathcal{I}$ are constructed by replacing nulls by constants:

$$\text{rep}(\mathcal{I}) \quad = \quad \{K \mid \mu(\mathcal{I}) \subseteq K \text{ for some valuation } \mu\}$$

Are naive instances expressive enough?

Naive instances have been extensively used in data exchange:

Proposition (FKMP03)

Let $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$, where Σ is a set of st-tgds. Then for every instance I of $\mathbf{S}$, there exists a naive instance $\mathcal{J}$ of $\mathbf{T}$ such that:

$$rep(\mathcal{J}) \quad = \quad \mathrm{Sol}_{\mathcal{M}}(I)$$

In fact, every universal solution satisfies the property mentioned above.

Are naive instances expressive enough?

But naive instances are not expressive enough to deal with incomplete information in the source instances:

Proposition (APR11)

Naive instances are not a strong representation system for the class of mappings specified by st-tgds

Are naive instances expressive enough?

Example

Consider a mapping $\mathcal{M}$ specified by:

$$\begin{array}{rcl} \texttt{Manager}(x, y) & \rightarrow & \texttt{Reports}(x, y) \\ \texttt{Manager}(x, x) & \rightarrow & \texttt{SelfManager}(x) \end{array}$$

The *canonical* universal solution for $\mathcal{I} = \{\texttt{Manager}(n, \text{Peter})\}$ under $\mathcal{M}$:

$$\mathcal{J} \quad = \quad \{\texttt{Reports}(n, \text{Peter})\}$$

But $\mathcal{J}$ is not a *good* solution for $\mathcal{I}$.

- It cannot represent the fact that if n is given value Peter, then $\texttt{SelfManager}(\text{Peter})$ should hold in the target.

Conditional instances

What should be added to naive instances to obtain a strong representation system?

Conditional instances

What should be added to naive instances to obtain a strong representation system?

- Answer from database theory: Conditions on the nulls

Conditional instances

What should be added to naive instances to obtain a strong representation system?

- Answer from database theory: Conditions on the nulls

Conditional instances: Naive instances plus *tuple conditions*

A tuple condition is a positive Boolean combinations of:

- equalities and inequalities between nulls, and between nulls and constants

Conditional instances

Example

$$\begin{array}{l|l} R(1, n_1) & n_1 = n_2 \\ R(n_1, n_2) & n_1 \neq n_2 \ \vee \ n_2 = 2 \end{array}$$

Conditional instances

Example

$$\begin{array}{l|l} R(1, n_1) & n_1 = n_2 \\ R(n_1, n_2) & n_1 \neq n_2 \ \vee \ n_2 = 2 \end{array}$$

Semantics:

Conditional instances

Example

$$
\begin{array}{l|l}
R(1, n_1) & n_1 = n_2 \\
R(n_1, n_2) & n_1 \neq n_2 \ \vee \ n_2 = 2
\end{array}
$$

Semantics:

$\underline{\mu(n_1) = \mu(n_2) = 2}$ $\quad$ $\underline{\mu(n_1) = \mu(n_2) = 3}$ $\quad$ $\underline{\mu(n_1) = 2, \mu(n_2) = 3}$

Conditional instances

Example

$$\begin{array}{l|l} R(1, n_1) & n_1 = n_2 \\ R(n_1, n_2) & n_1 \neq n_2 \;\vee\; n_2 = 2 \end{array}$$

Semantics:

$\mu(n_1) = \mu(n_2) = 2$	$\mu(n_1) = \mu(n_2) = 3$	$\mu(n_1) = 2, \mu(n_2) = 3$
$R(1,2)$		
$R(2,2)$		

Conditional instances

Example

$$\begin{array}{l|l} R(1, n_1) & n_1 = n_2 \\ R(n_1, n_2) & n_1 \neq n_2 \ \vee \ n_2 = 2 \end{array}$$

Semantics:

$$\frac{\mu(n_1) = \mu(n_2) = 2}{\begin{array}{c} R(1,2) \\ R(2,2) \end{array}} \qquad \frac{\mu(n_1) = \mu(n_2) = 3}{R(1,3)} \qquad \frac{\mu(n_1) = 2, \mu(n_2) = 3}{}$$

Conditional instances

Example

$$\begin{array}{l|l} R(1, n_1) & n_1 = n_2 \\ R(n_1, n_2) & n_1 \neq n_2 \ \vee \ n_2 = 2 \end{array}$$

Semantics:

$$\frac{\mu(n_1) = \mu(n_2) = 2}{\begin{array}{c} R(1,2) \\ R(2,2) \end{array}} \qquad \frac{\mu(n_1) = \mu(n_2) = 3}{R(1,3)} \qquad \frac{\mu(n_1) = 2, \mu(n_2) = 3}{\begin{array}{c} \\ R(2,3) \end{array}}$$

Conditional instances

Example

$$\begin{array}{l|l} R(1, n_1) & n_1 = n_2 \\ R(n_1, n_2) & n_1 \neq n_2 \ \vee \ n_2 = 2 \end{array}$$

Semantics:

$$\frac{\mu(n_1) = \mu(n_2) = 2}{\begin{array}{c} R(1,2) \\ R(2,2) \end{array}} \qquad \frac{\mu(n_1) = \mu(n_2) = 3}{R(1,3)} \qquad \frac{\mu(n_1) = 2, \mu(n_2) = 3}{\begin{array}{c} \\ R(2,3) \end{array}}$$

Interpretations of a conditional instance $\mathcal{I}$:

$$\text{rep}(\mathcal{I}) \quad = \quad \{K \mid \mu(\mathcal{I}) \subseteq K \text{ for some valuation } \mu\}$$

Positive conditional instances

Many problems are intractable over conditional instances.

- We also consider a restricted class of conditional instances

Positive conditional instances: Conditional instances without inequalities

(Positive) conditional instances are enough

Theorem (APR11)

Both conditional instances and positive conditional instances are strong representation systems for the class of mappings specified by st-tgds.

Example

Consider again the mapping $\mathcal{M}$ specified by:

$$\begin{array}{rcl} \texttt{Manager}(x, y) & \rightarrow & \texttt{Reports}(x, y) \\ \texttt{Manager}(x, x) & \rightarrow & \texttt{SelfManager}(x) \end{array}$$

The following is a universal solution for $\mathcal{I} = \{\texttt{Manager}(n, \text{Peter})\}$

$$\begin{array}{l|l} \texttt{Reports}(n, \text{Peter}) & \textit{true} \\ \texttt{SelfManager}(\text{Peter}) & n = \text{Peter} \end{array}$$

Positive conditional instances are *exactly* the needed representation system

Theorem (APR11)

All the following are needed to obtain a strong representation system for the class of mappings specified by st-tgds:

Positive conditional instances are *exactly* the needed representation system

Theorem (APR11)

All the following are needed to obtain a strong representation system for the class of mappings specified by st-tgds:

- *equalities between nulls*

Positive conditional instances are *exactly* the needed representation system

Theorem (APR11)

All the following are needed to obtain a strong representation system for the class of mappings specified by st-tgds:

- *equalities between nulls*
  - *There exists a mapping $\mathcal{M}$ given by st-tgds and a source naive instance $\mathcal{I}$ such that for every target positive conditional $\mathcal{J}$ not mentioning equalities between nulls:* $rep(\mathcal{J}) \neq \mathrm{Sol}_{\mathcal{M}}(rep(\mathcal{I}))$

Positive conditional instances are *exactly* the needed representation system

Theorem (APR11)

All the following are needed to obtain a strong representation system for the class of mappings specified by st-tgds:

- *equalities between nulls*
  - *There exists a mapping $\mathcal{M}$ given by st-tgds and a source naive instance $\mathcal{I}$ such that for every target positive conditional $\mathcal{J}$ not mentioning equalities between nulls:* $rep(\mathcal{J}) \neq \mathrm{Sol}_{\mathcal{M}}(rep(\mathcal{I}))$
- *equalities between constant and nulls*

Positive conditional instances are *exactly* the needed representation system

Theorem (APR11)

All the following are needed to obtain a strong representation system for the class of mappings specified by st-tgds:

- *equalities between nulls*
  - *There exists a mapping $\mathcal{M}$ given by st-tgds and a source naive instance $\mathcal{I}$ such that for every target positive conditional $\mathcal{J}$ not mentioning equalities between nulls:* $rep(\mathcal{J}) \neq \mathrm{Sol}_{\mathcal{M}}(rep(\mathcal{I}))$
- *equalities between constant and nulls*
- *conjunctions and disjunctions*

Positive conditional instances are *exactly* the needed representation system

Theorem (APR11)

All the following are needed to obtain a strong representation system for the class of mappings specified by st-tgds:

- *equalities between nulls*
  - *There exists a mapping $\mathcal{M}$ given by st-tgds and a source naive instance $\mathcal{I}$ such that for every target positive conditional $\mathcal{J}$ not mentioning equalities between nulls: $rep(\mathcal{J}) \neq \mathrm{Sol}_{\mathcal{M}}(rep(\mathcal{I}))$*
- *equalities between constant and nulls*
- *conjunctions and disjunctions*

Conditional instances are enough but not minimal.

Positive conditional instance can be used in practice!

Let $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$, where Σ is a set of st-tgds.

Positive conditional instance can be used in practice!

Let $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$, where Σ is a set of st-tgds.

Theorem (APR11)

There exists a polynomial time algorithm that, given a positive conditional instance $\mathcal{I}$ over $\mathbf{S}$, computes a positive conditional instance $\mathcal{J}$ over $\mathbf{T}$ that is a universal solution for $\mathcal{I}$ under $\mathcal{M}$.

Positive conditional instance can be used in practice!

Let $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$, where Σ is a set of st-tgds.

Theorem (APR11)

There exists a polynomial time algorithm that, given a positive conditional instance $\mathcal{I}$ over $\mathbf{S}$, computes a positive conditional instance $\mathcal{J}$ over $\mathbf{T}$ that is a universal solution for $\mathcal{I}$ under $\mathcal{M}$.

Remark

They are also appropriate for query answering in data exchange.

- Same polynomial-time cases as in the usual setting

Outline

- The need for a more general data exchange framework
  - Two important scenarios: Incomplete databases and knowledge bases
- Formalism for exchanging representations systems
- Applications to incomplete databases
- Applications to metadata management
- Concluding remarks

The composition operator

Definition (FKPT04)

Let $\mathcal{M}_{12}$ be a mapping from $\mathbf{S}_1$ to $\mathbf{S}_2$, and $\mathcal{M}_{23}$ a mapping from $\mathbf{S}_2$ to $\mathbf{S}_3$:

$$\mathcal{M}_{12} \circ \mathcal{M}_{23} = \{(I_1, I_3) \mid \exists I_2 : (I_1, I_2) \in \mathcal{M}_{12} \text{ and } (I_2, I_3) \in \mathcal{M}_{23}\}$$

Expressing the composition of mappings

Question

What is the right language for expressing the composition?

- st-tgds?

Example (FKPT04)

Consider the mappings $\mathcal{M}_{12}$:

$$\begin{aligned} node(x) &\rightarrow \exists y\, coloring(x, y) \\ edge(x, y) &\rightarrow edge'(x, y) \end{aligned}$$

and $\mathcal{M}_{23}$:

$$\begin{aligned} edge'(x, y) \wedge coloring(x, u) \wedge coloring(y, u) &\rightarrow error(x, y) \\ coloring(x, y) &\rightarrow color(y) \end{aligned}$$

SO tgds: The right language for expressing the composition of mappings

Example (Cont'd)

The following dependency defines the composition:

$$\exists f \Big[\forall x (node(x) \rightarrow color(f(x))) \wedge \\ \forall x \forall y (edge(x,y) \wedge f(x) = f(y) \rightarrow error(x,y)) \Big]$$

SO tgds: The right language for expressing the composition of mappings

Example (Cont'd)

The following dependency defines the composition:

$$\exists f \left[\forall x (node(x) \rightarrow color(f(x))) \wedge \right.$$
$$\left. \forall x \forall y (edge(x,y) \wedge f(x) = f(y) \rightarrow error(x,y)) \right]$$

This example shows the main ingredients of SO tgds:

- Predicates including terms: $color(f(x))$
- Equality between terms: $f(x) = f(y)$

SO tgds: The right language for expressing the composition of mappings

SO tgds were introduced in [FKPT04]

- They have good properties regarding composition

SO tgds: The right language for expressing the composition of mappings

SO tgds were introduced in [FKPT04]

- They have good properties regarding composition

Theorem (FKPT04)

If $\mathcal{M}_{12}$ and $\mathcal{M}_{23}$ are specified by SO tgds, then $\mathcal{M}_{12} \circ \mathcal{M}_{23}$ can be specified by an SO tgd

SO tgds: The right language for expressing the composition of mappings

SO tgds were introduced in [FKPT04]

- They have good properties regarding composition

Theorem (FKPT04)

If $\mathcal{M}_{12}$ and $\mathcal{M}_{23}$ are specified by SO tgds, then $\mathcal{M}_{12} \circ \mathcal{M}_{23}$ can be specified by an SO tgd

- *There exists an exponential time algorithm that computes such SO tgds*

SO tgds: The right language for expressing the composition of mappings

Corollary (FKPT04)

The composition of a finite number of mappings, each defined by a finite set of st-tgds, is defined by an SO tgd

SO tgds: The right language for expressing the composition of mappings

Corollary (FKPT04)

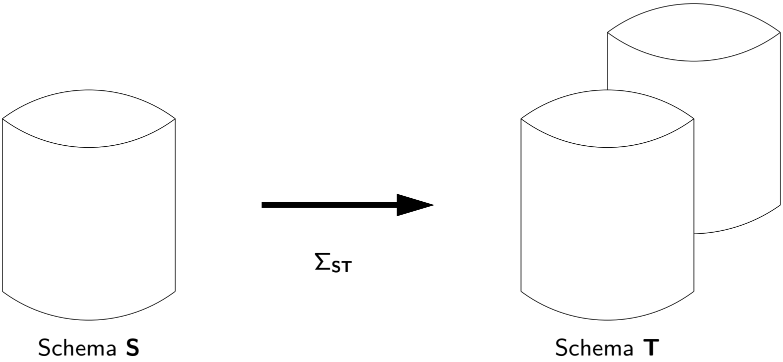
The composition of a finite number of mappings, each defined by a finite set of st-tgds, is defined by an SO tgd

But not only that, SO tgds are *exactly* the right language:

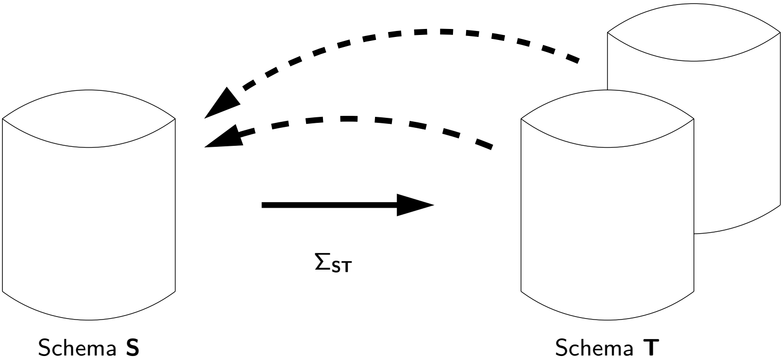
Theorem (FKPT05)

Every SO tgd defines the composition of a finite number of mappings, each defined by a finite set of st-tgds.

The inverse operator



The inverse operator



The inverse operator

Question

What is the semantics of the inverse operator?

This turns out to be a very difficult question.

Several notions of inverse have been considered:

- Fagin-inverse [F06]
- Quasi-inverse [FKPT07]
- Maximum recovery [APR08]
- Maximum extended recovery [FKPT09]
- $\mathcal{C}$-maximum recovery [APRR09]

The inverse operator

Question

What is the semantics of the inverse operator?

This turns out to be a very difficult question.

Several notions of inverse have been considered:

- Fagin-inverse [F06]
- Quasi-inverse [FKPT07]
- Maximum recovery [APR08]
- Maximum extended recovery [FKPT09]
- $\mathcal{C}$-maximum recovery [APRR09]

Recovery: specifies how to recover sound information

Data may be lost in the exchange through a mapping $\mathcal{M}$

- We would like to find a mapping $\mathcal{M}^\star$ that at least recovers sound data w.r.t. $\mathcal{M}$
  - $\mathcal{M}^\star$ is called a recovery of $\mathcal{M}$

Recovery: specifies how to recover sound information

Data may be lost in the exchange through a mapping $\mathcal{M}$

- We would like to find a mapping $\mathcal{M}^\star$ that at least recovers sound data w.r.t. $\mathcal{M}$
  - $\mathcal{M}^\star$ is called a recovery of $\mathcal{M}$

Example

Consider a mapping $\mathcal{M}$ specified by:

$$emp(x, y, z) \wedge y \neq z \quad \rightarrow \quad shuttle(x, z)$$

What mappings are recoveries of $\mathcal{M}$?

Recovery: specifies how to recover sound information

Data may be lost in the exchange through a mapping $\mathcal{M}$

- We would like to find a mapping $\mathcal{M}^\star$ that at least recovers sound data w.r.t. $\mathcal{M}$
  - $\mathcal{M}^\star$ is called a recovery of $\mathcal{M}$

Example

Consider a mapping $\mathcal{M}$ specified by:

$$emp(x, y, z) \wedge y \neq z \quad \rightarrow \quad shuttle(x, z)$$

What mappings are recoveries of $\mathcal{M}$?

$$\mathcal{M}_1^\star: \quad shuttle(x, z) \quad \rightarrow \quad \exists u \exists v \, emp(x, u, v)$$

Recovery: specifies how to recover sound information

Data may be lost in the exchange through a mapping $\mathcal{M}$

- We would like to find a mapping $\mathcal{M}^\star$ that at least recovers sound data w.r.t. $\mathcal{M}$
  - $\mathcal{M}^\star$ is called a recovery of $\mathcal{M}$

Example

Consider a mapping $\mathcal{M}$ specified by:

$$emp(x, y, z) \wedge y \neq z \quad \rightarrow \quad shuttle(x, z)$$

What mappings are recoveries of $\mathcal{M}$?

$$\mathcal{M}_1^\star: \quad shuttle(x, z) \quad \rightarrow \quad \exists u \exists v \, emp(x, u, v) \quad \checkmark$$

Recovery: specifies how to recover sound information

Data may be lost in the exchange through a mapping $\mathcal{M}$

- We would like to find a mapping $\mathcal{M}^\star$ that at least recovers sound data w.r.t. $\mathcal{M}$
  - $\mathcal{M}^\star$ is called a recovery of $\mathcal{M}$

Example

Consider a mapping $\mathcal{M}$ specified by:

$$emp(x, y, z) \wedge y \neq z \quad \rightarrow \quad shuttle(x, z)$$

What mappings are recoveries of $\mathcal{M}$?

$$\begin{array}{llll} \mathcal{M}_1^\star: & shuttle(x, z) & \rightarrow & \exists u \exists v\, emp(x, u, v) \quad \checkmark \\ \mathcal{M}_2^\star: & shuttle(x, z) & \rightarrow & \exists u\, emp(x, u, z) \end{array}$$

Recovery: specifies how to recover sound information

Data may be lost in the exchange through a mapping $\mathcal{M}$

- We would like to find a mapping $\mathcal{M}^\star$ that at least recovers sound data w.r.t. $\mathcal{M}$
  - $\mathcal{M}^\star$ is called a recovery of $\mathcal{M}$

Example

Consider a mapping $\mathcal{M}$ specified by:

$$emp(x, y, z) \wedge y \neq z \quad \rightarrow \quad shuttle(x, z)$$

What mappings are recoveries of $\mathcal{M}$?

$$\begin{array}{llcll} \mathcal{M}_1^\star: & shuttle(x, z) & \rightarrow & \exists u \exists v\, emp(x, u, v) & \checkmark \\ \mathcal{M}_2^\star: & shuttle(x, z) & \rightarrow & \exists u\, emp(x, u, z) & \checkmark \end{array}$$

Recovery: specifies how to recover sound information

Data may be lost in the exchange through a mapping $\mathcal{M}$

- We would like to find a mapping $\mathcal{M}^\star$ that at least recovers sound data w.r.t. $\mathcal{M}$
  - $\mathcal{M}^\star$ is called a recovery of $\mathcal{M}$

Example

Consider a mapping $\mathcal{M}$ specified by:

$$emp(x, y, z) \wedge y \neq z \quad \rightarrow \quad shuttle(x, z)$$

What mappings are recoveries of $\mathcal{M}$?

$$\begin{array}{llcll}
\mathcal{M}_1^\star: & shuttle(x, z) & \rightarrow & \exists u \exists v\, emp(x, u, v) & \checkmark \\
\mathcal{M}_2^\star: & shuttle(x, z) & \rightarrow & \exists u\, emp(x, u, z) & \checkmark \\
\mathcal{M}_3^\star: & shuttle(x, z) & \rightarrow & \exists u\, emp(x, z, u) &
\end{array}$$

Recovery: specifies how to recover sound information

Data may be lost in the exchange through a mapping $\mathcal{M}$

- We would like to find a mapping $\mathcal{M}^\star$ that at least recovers sound data w.r.t. $\mathcal{M}$
  - $\mathcal{M}^\star$ is called a recovery of $\mathcal{M}$

Example

Consider a mapping $\mathcal{M}$ specified by:

$$emp(x, y, z) \wedge y \neq z \quad \rightarrow \quad shuttle(x, z)$$

What mappings are recoveries of $\mathcal{M}$?

$$\begin{array}{llcll}
\mathcal{M}_1^\star: & shuttle(x, z) & \rightarrow & \exists u \exists v\, emp(x, u, v) & \checkmark \\
\mathcal{M}_2^\star: & shuttle(x, z) & \rightarrow & \exists u\, emp(x, u, z) & \checkmark \\
\mathcal{M}_3^\star: & shuttle(x, z) & \rightarrow & \exists u\, emp(x, z, u) & \times
\end{array}$$

Maximum recovery: The *most informative* recovery

Example

Consider again mapping $\mathcal{M}$ specified by:

$$emp(x, y, z) \wedge y \neq z \quad \rightarrow \quad shuttle(x, z)$$

These mappings are recoveries of $\mathcal{M}$:

$$\begin{array}{llcl} \mathcal{M}_1^\star: & shuttle(x, z) & \rightarrow & \exists u \exists v \, emp(x, u, v) \\ \mathcal{M}_2^\star: & shuttle(x, z) & \rightarrow & \exists u \, emp(x, u, z) \end{array}$$

Maximum recovery: The *most informative* recovery

Example

Consider again mapping $\mathcal{M}$ specified by:

$$emp(x, y, z) \wedge y \neq z \quad \rightarrow \quad shuttle(x, z)$$

These mappings are recoveries of $\mathcal{M}$:

$$\begin{array}{llll} \mathcal{M}_1^\star: & shuttle(x, z) & \rightarrow & \exists u \exists v \, emp(x, u, v) \\ \mathcal{M}_2^\star: & shuttle(x, z) & \rightarrow & \exists u \, emp(x, u, z) \end{array}$$

Intuitively: $\mathcal{M}_2^\star$ is better than $\mathcal{M}_1^\star$

Maximum recovery: The *most informative* recovery

Example

Consider again mapping $\mathcal{M}$ specified by:

$$emp(x, y, z) \wedge y \neq z \quad \rightarrow \quad shuttle(x, z)$$

These mappings are recoveries of $\mathcal{M}$:

$$\begin{array}{lccc} \mathcal{M}_1^\star: & shuttle(x, z) & \rightarrow & \exists u \exists v\, emp(x, u, v) \\ \mathcal{M}_2^\star: & shuttle(x, z) & \rightarrow & \exists u\, emp(x, u, z) \\ \mathcal{M}_4^\star: & shuttle(x, z) & \rightarrow & \exists u\, emp(x, u, z) \wedge u \neq z \end{array}$$

Intuitively: $\mathcal{M}_2^\star$ is better than $\mathcal{M}_1^\star$

Maximum recovery: The *most informative* recovery

Example

Consider again mapping $\mathcal{M}$ specified by:

$$emp(x, y, z) \wedge y \neq z \quad \rightarrow \quad shuttle(x, z)$$

These mappings are recoveries of $\mathcal{M}$:

$$\begin{array}{llcl}
\mathcal{M}_1^\star: & shuttle(x, z) & \rightarrow & \exists u \exists v\, emp(x, u, v) \\
\mathcal{M}_2^\star: & shuttle(x, z) & \rightarrow & \exists u\, emp(x, u, z) \\
\mathcal{M}_4^\star: & shuttle(x, z) & \rightarrow & \exists u\, emp(x, u, z) \wedge u \neq z
\end{array}$$

Intuitively: $\mathcal{M}_2^\star$ is better than $\mathcal{M}_1^\star$
$\mathcal{M}_4^\star$ is better than $\mathcal{M}_2^\star$ and $\mathcal{M}_1^\star$

Maximum recovery: The *most informative* recovery

Example

Consider again mapping $\mathcal{M}$ specified by:

$$emp(x, y, z) \wedge y \neq z \quad \rightarrow \quad shuttle(x, z)$$

These mappings are recoveries of $\mathcal{M}$:

$$\begin{array}{llcl}
\mathcal{M}_1^\star: & shuttle(x, z) & \rightarrow & \exists u \exists v\, emp(x, u, v) \\
\mathcal{M}_2^\star: & shuttle(x, z) & \rightarrow & \exists u\, emp(x, u, z) \\
\mathcal{M}_4^\star: & shuttle(x, z) & \rightarrow & \exists u\, emp(x, u, z) \wedge u \neq z
\end{array}$$

Intuitively: $\mathcal{M}_2^\star$ is better than $\mathcal{M}_1^\star$
$\mathcal{M}_4^\star$ is better than $\mathcal{M}_2^\star$ and $\mathcal{M}_1^\star$

We would like to find a recovery of $\mathcal{M}$ that is better than any other recovery: Maximum recovery

The notion of recovery: Formalization

Definition (APR08)

Let $\mathcal{M}$ be a mapping from $\mathbf{S}_1$ to $\mathbf{S}_2$ and $\mathcal{M}^\star$ a mapping from $\mathbf{S}_2$ to $\mathbf{S}_1$. Then $\mathcal{M}^\star$ is a recovery of $\mathcal{M}$ if:

for every instance I of $\mathbf{S}_1$: $(I, I) \in \mathcal{M} \circ \mathcal{M}^\star$

The notion of recovery: Formalization

Definition (APR08)

Let $\mathcal{M}$ be a mapping from $\mathbf{S}_1$ to $\mathbf{S}_2$ and $\mathcal{M}^\star$ a mapping from $\mathbf{S}_2$ to $\mathbf{S}_1$. Then $\mathcal{M}^\star$ is a recovery of $\mathcal{M}$ if:

for every instance I of $\mathbf{S}_1$: $(I, I) \in \mathcal{M} \circ \mathcal{M}^\star$

Example

Consider again mapping $\mathcal{M}$ specified by:

$$emp(x, y, z) \wedge y \neq z \quad \rightarrow \quad shuttle(x, z)$$

This mapping is not a recovery of $\mathcal{M}$:

$$\mathcal{M}_3^\star: \quad shuttle(x, z) \quad \rightarrow \quad \exists u \, emp(x, z, u)$$

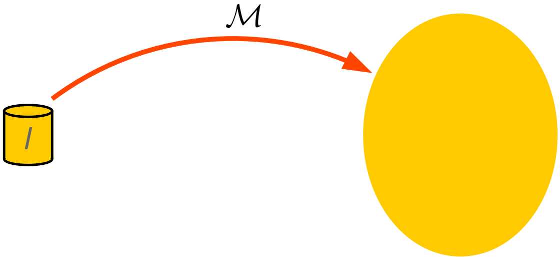
The notion of recovery: Formalization

Example (Cont'd)

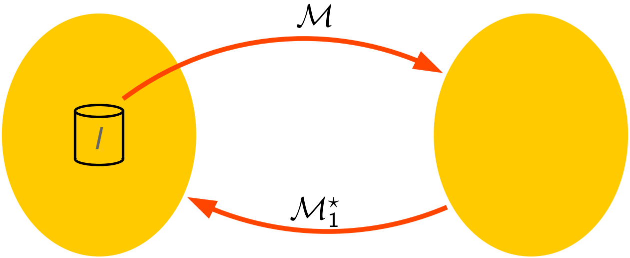
On the other hand, these mappings are recoveries of $\mathcal{M}$:

$$\begin{array}{llcl}
\mathcal{M}_1^\star: & \mathit{shuttle}(x,z) & \rightarrow & \exists u \exists v\, \mathit{emp}(x,u,v) \\
\mathcal{M}_2^\star: & \mathit{shuttle}(x,z) & \rightarrow & \exists u\, \mathit{emp}(x,u,z) \\
\mathcal{M}_4^\star: & \mathit{shuttle}(x,z) & \rightarrow & \exists u\, \mathit{emp}(x,u,z) \wedge u \neq z
\end{array}$$

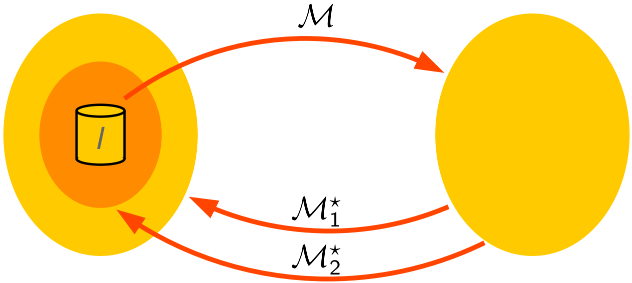
The notion of maximum recovery



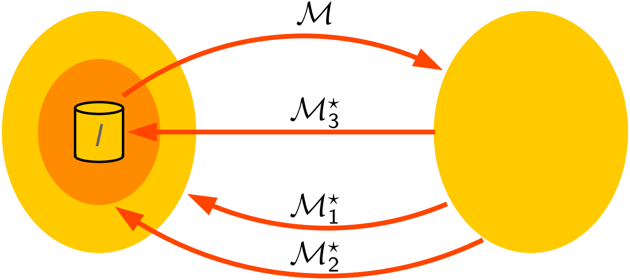
The notion of maximum recovery



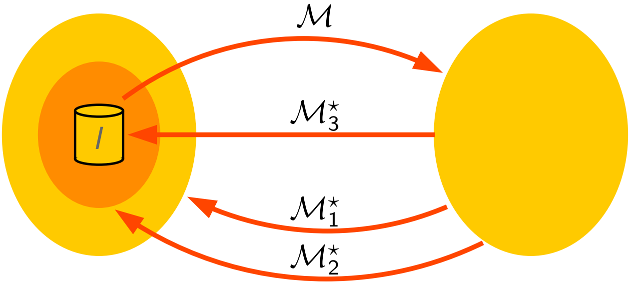
The notion of maximum recovery



The notion of maximum recovery



The notion of maximum recovery



Definition (APR08)

$\mathcal{M}^\star$ is a maximum recovery of $\mathcal{M}$ if:

- $\mathcal{M}^\star$ is a recovery of $\mathcal{M}$
- for every recovery $\mathcal{M}'$ of $\mathcal{M}$: $\mathcal{M} \circ \mathcal{M}^\star \subseteq \mathcal{M} \circ \mathcal{M}'$

On the existence of maximum recoveries

Theorem (APR08)

Every mapping specified by a finite set of st-tgds has a maximum recovery.

On the existence of maximum recoveries

Theorem (APR08)

Every mapping specified by a finite set of st-tgds has a maximum recovery.

But this does not hold if one also considers naive instances in the source.

- Maximum extended recovery was introduced to overcome this limitation

We need to combine the operators

Can we combine the composition and inverse operators?

- Is there a good language for both operators?

We need to combine the operators

Can we combine the composition and inverse operators?

- Is there a good language for both operators?

Bad news:

Theorem (APR11)

There exists a mapping specified by an SO tgd that does not have a maximum recovery.

We need to combine the operators

Even worse:

- Previous mapping has neither a Fagin-inverse nor a quasi-inverse nor a $\mathcal{C}$-maximum recovery ($\mathbf{CQ} \subseteq \mathcal{C}$)
- Semantics of maximum extended recovery is appropriate for st-tgds.

We need to combine the operators

Even worse:

- Previous mapping has neither a Fagin-inverse nor a quasi-inverse nor a $\mathcal{C}$-maximum recovery ($\mathbf{CQ} \subseteq \mathcal{C}$)
- Semantics of maximum extended recovery is appropriate for st-tgds.

Do we need yet another notion of inverse?

We need to combine the operators

Even worse:

- Previous mapping has neither a Fagin-inverse nor a quasi-inverse nor a $\mathcal{C}$-maximum recovery ($\mathbf{CQ} \subseteq \mathcal{C}$)
- Semantics of maximum extended recovery is appropriate for st-tgds.

Do we need yet another notion of inverse?

- No, we need to revisit the semantics of mappings

What went wrong?

Key observation: A target instance of a mapping can be the source instance of another mapping.

- Sources instances may contain null values

What went wrong?

Key observation: A target instance of a mapping can be the source instance of another mapping.

- Sources instances may contain null values

Theorem (APR11)

Positive conditional instances are a strong representation system for the class of mappings specified by SO tgds.

A solution to the problem

Theorem (APR11)

If (usual) instances are replaced by positive conditional instances:

- *SO tgds are still the right language for the composition of mappings given by st-tgds*
- *Every mapping specified by an SO tgd admits a maximum recovery*

Outline

- The need for a more general data exchange framework
  - Two important scenarios: Incomplete databases and knowledge bases
- Formalism for exchanging representations systems
- Applications to incomplete databases
- Applications to metadata management
- Concluding remarks

We can exchange more than complete data

We propose a general formalism to exchange *representation systems*

- Applications to incomplete instances
- Applications to metadata management
- Applications to knowledge bases

We can exchange more than complete data

We propose a general formalism to exchange *representation systems*

- Applications to incomplete instances
- Applications to metadata management
- Applications to knowledge bases

Next step: Apply our general setting to the Semantic Web

- Semantic Web data has *nulls* (blank nodes)
- Semantic Web specifications have rules (RDFS, OWL)

Thank you!

Bibliography

[FKMP03] R. Fagin, P. G. Kolaitis, R. J. Miller, L. Popa. Data Exchange: Semantics and Query Answering. ICDT 2003: 207-224

[FKPT04] R. Fagin, P. G. Kolaitis, L. Popa, W.-C. Tan. Composing Schema Mappings: Second-Order Dependencies to the Rescue. PODS 2004: 83-94

[FKPT05] R. Fagin, P. G. Kolaitis, L. Popa, W.-C. Tan. Composing schema mappings: Second-order dependencies to the rescue. TODS 30(4): 994-1055, 2005

[F06] R. Fagin. Inverting schema mappings. PODS 2006: 50-59

Bibliography

[FKPT07] R. Fagin, P. G. Kolaitis, L. Popa, W.-C. Tan. Quasi-inverses of schema mappings. PODS 2007: 123-132

[FKPT09] R. Fagin, P. G. Kolaitis, L. Popa, W.-C. Tan. Reverse data exchange: coping with nulls. PODS 2009: 23-32

[APR08] M. Arenas, J. Pérez, C. Riveros. The recovery of a schema mapping: bringing exchanged data back. PODS 2008: 13-22

[APRR09] M. Arenas, J. Pérez, J. Reutter, C. Riveros. Inverting Schema Mappings: Bridging the Gap between Theory and Practice. PVLDB 2(1): 1018-1029, 2009

[APR11] M. Arenas, J. Pérez, J. Reutter. Data Exchange beyond Complete Data. PODS 2011: 83-94