Information integration

Maurizio Lenzerini

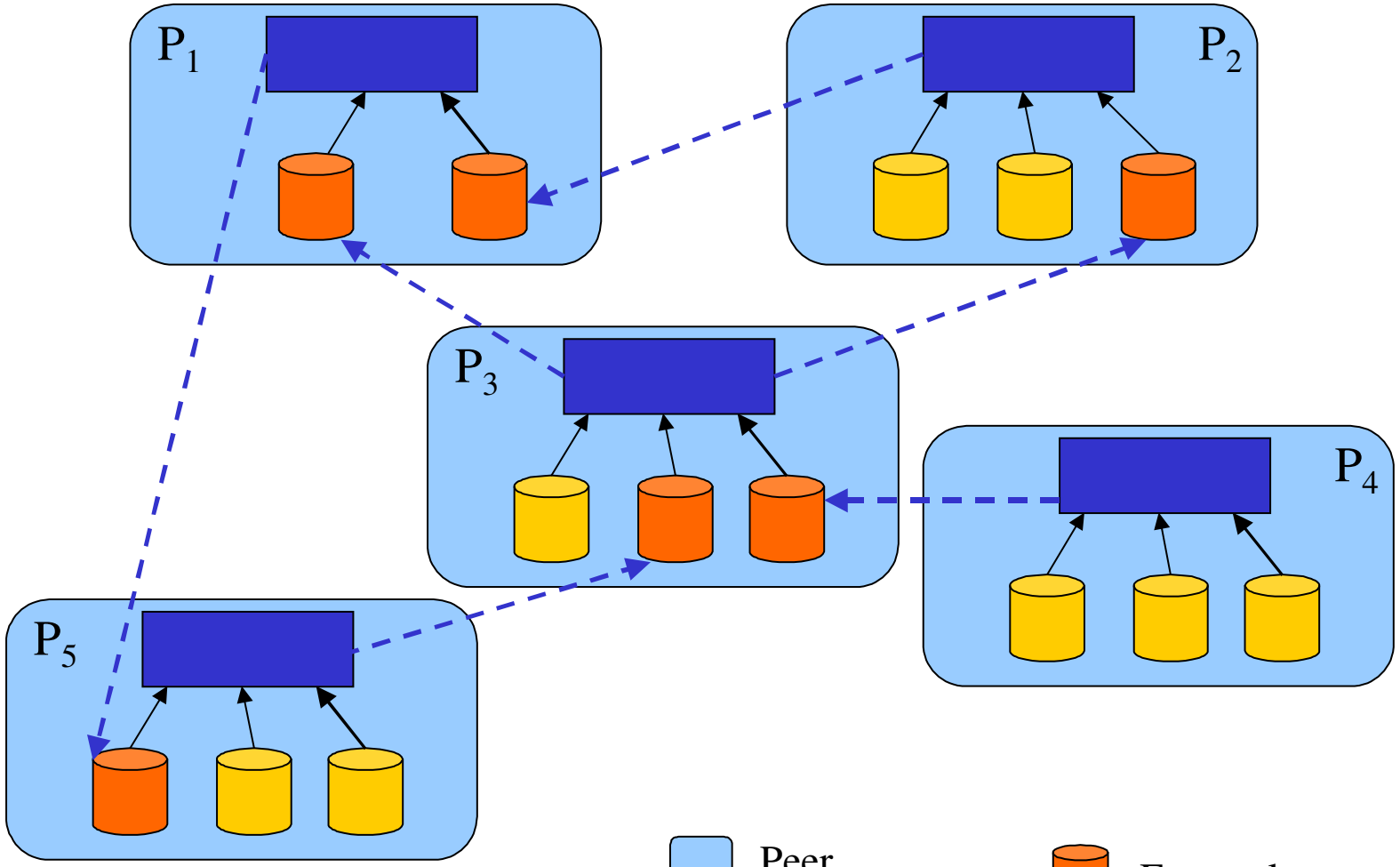
Dipartimento di Informatica e Sistemistica "Antonio Ruberti"

Università di Roma "La Sapienza"

Tutorial at IJCAI 2003

Acapulco, Mexico – August 2003

Peer-based architecture for information integration

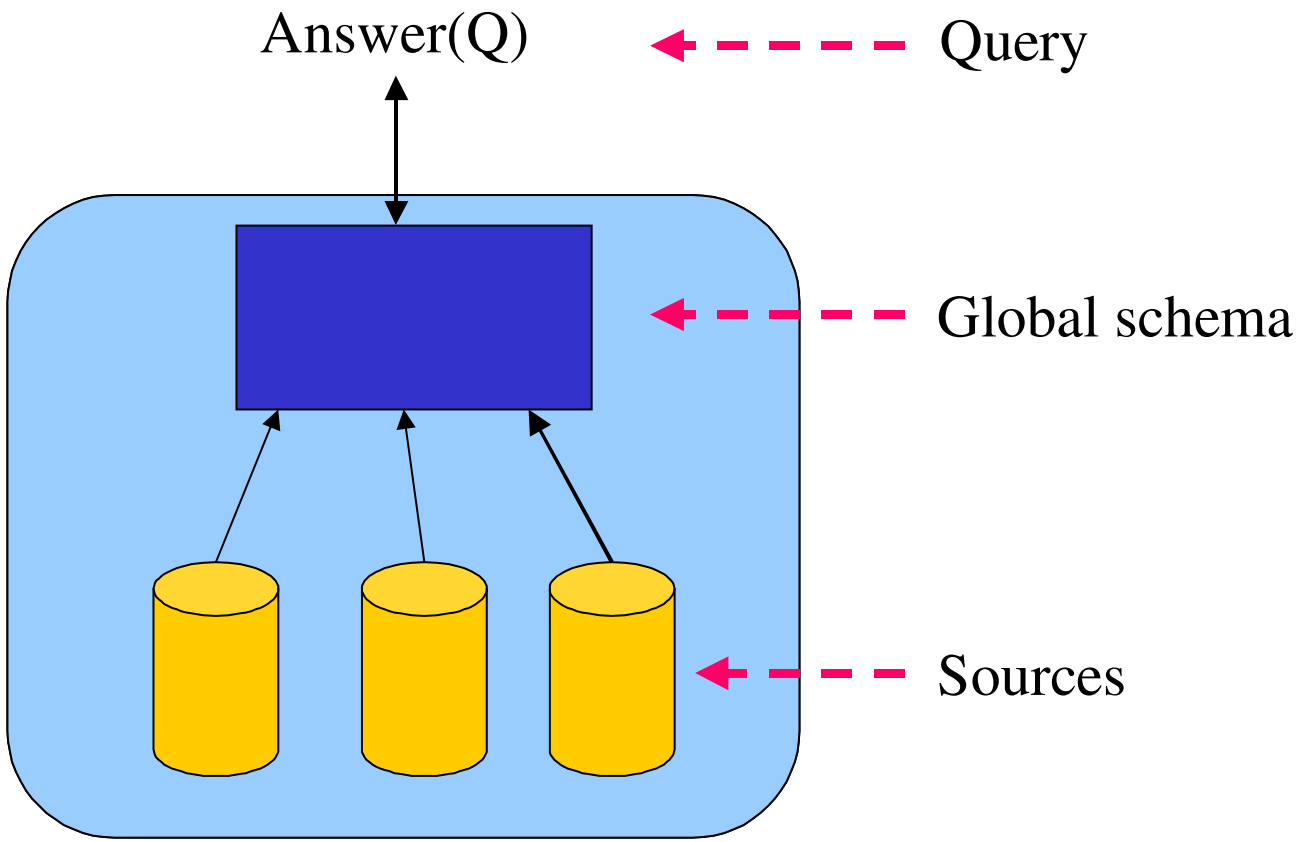


Operations:

- Answer(Q, P_i)
- Materialize(P_i)

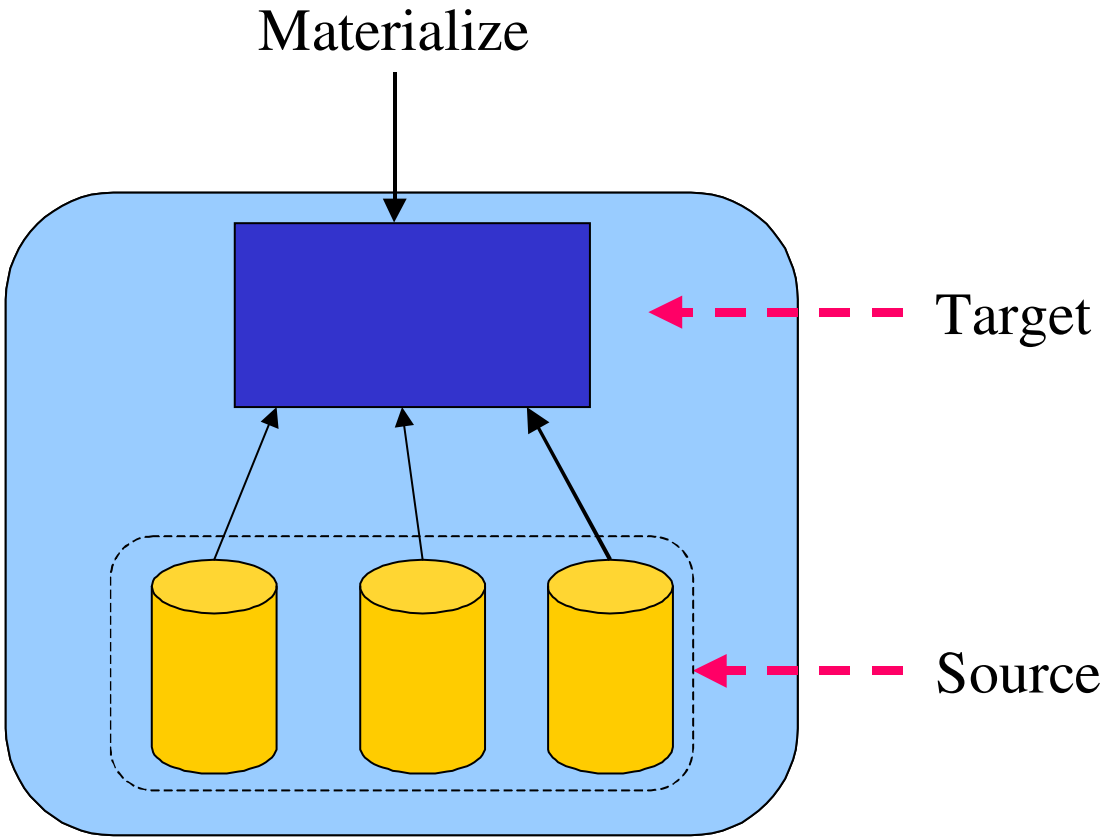
Special case 1: Mediator-based data integration

- One peer
- No external sources
- Queries over the peer schema



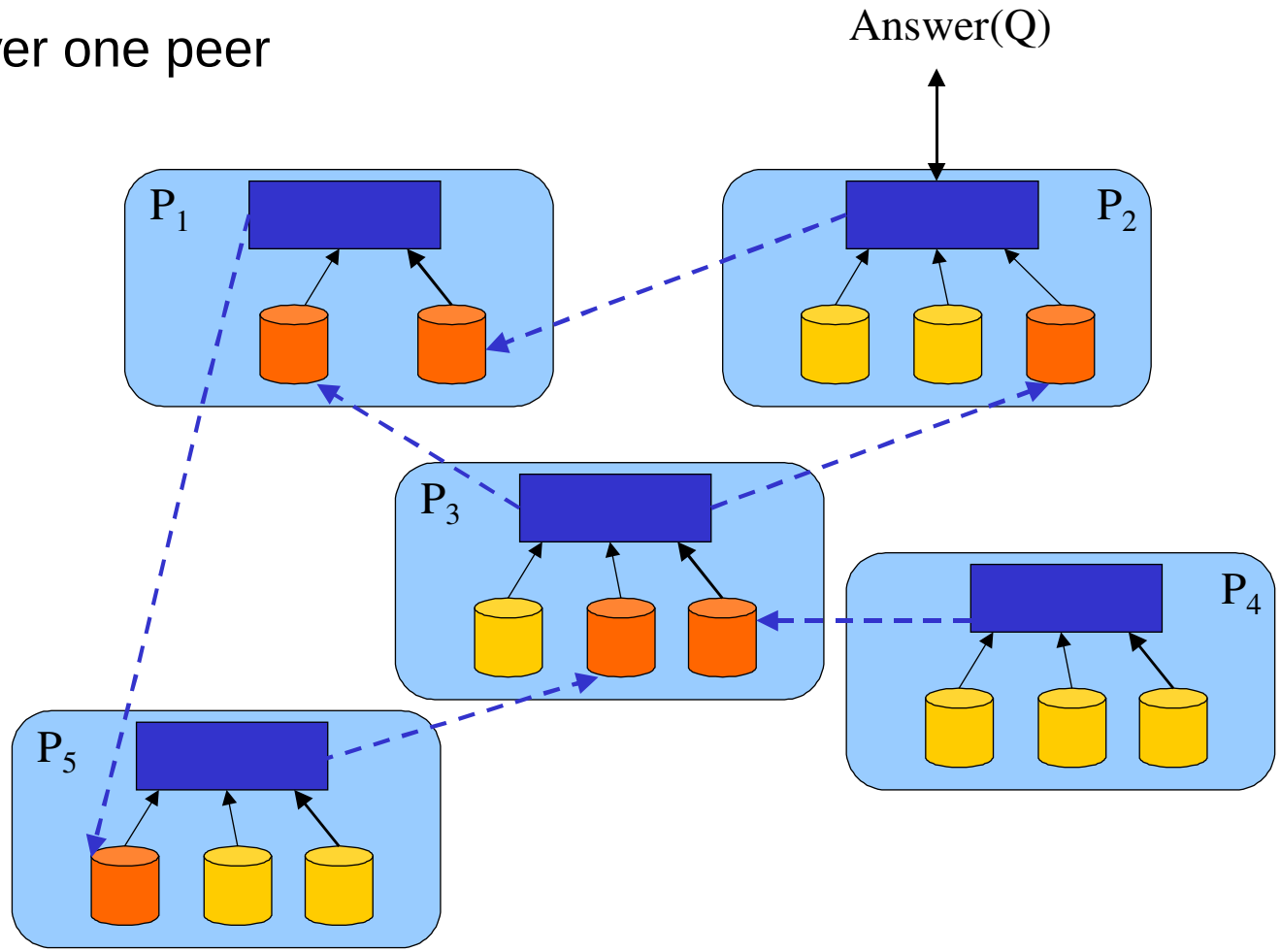
Special case 2: Data exchange

- One peer
- No external sources
- Materialization



Special case 3: P2P data integration

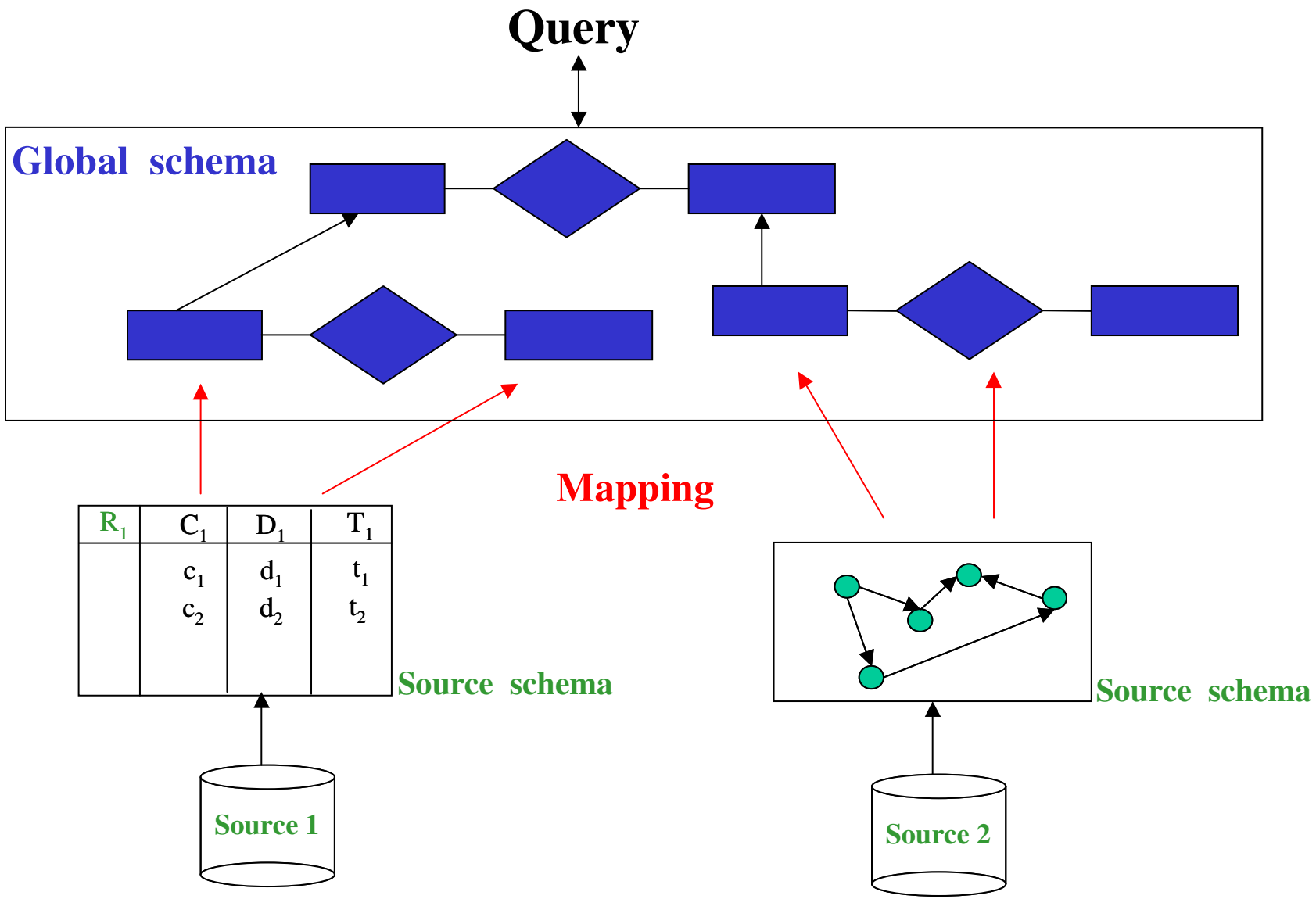
- Several peers
- Peer with local and external sources
- Queries over one peer



Outline

- Peer-based Distributed Information Systems
- Data integration
  - Approaches to data integration
  - Query answering in different approaches
  - Dealing with inconsistency
- Data exchange
- P2P data integration
- Conclusions

Data integration



Main problems in data integration

1. How to construct the global schema
2. (Automatic) source wrapping
3. How to discover mappings between the sources and the global schema
4. Limitations in the mechanisms for accessing the sources
5. Data extraction, cleaning and reconciliation
6. How to process updates expressed on the global schema, and updates expressed on the sources
7. The modeling problem: How to model the mappings between the sources and the global schema
8. The querying problem: How to answer queries expressed on the global schema
9. Query optimization

Formal framework for data integration

A **data integration system** $\mathcal{I}$ is a triple $\langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$, where

- $\mathcal{G}$ is the global schema

  The global schema is a logical theory over an alphabet $\mathcal{A}_\mathcal{G}$

- $\mathcal{S}$ is the source schema

  The source schema is constituted simply by an alphabet $\mathcal{A}_\mathcal{S}$ disjoint from $\mathcal{A}_\mathcal{G}$

- $\mathcal{M}$ is the mapping between $\mathcal{S}$ and $\mathcal{G}$

  Different approaches to the specification of mapping

Semantics of a data integration system

Which are the databases that satisfy $\mathcal{I}$, i.e., which are the logical models of $\mathcal{I}$?

The databases that satisfy $\mathcal{I}$ are logical interpretations for $\mathcal{A}_\mathcal{G}$ (called global databases). We refer only to databases over a fixed infinite domain Γ of constants.

Let $\mathcal{C}$ be **a source database** over Γ (also called source model), fixing the extension of the predicates of $\mathcal{A}_\mathcal{S}$ (thus modeling the data present in the sources).

The set of models of (i.e., databases for $\mathcal{A}_\mathcal{G}$ that satisfy) $\mathcal{I}$ relative to $\mathcal{C}$ is:

$$sem^{\mathcal{C}}(\mathcal{I}) = \{\ \mathcal{B} \quad | \quad \mathcal{B} \text{ is a } \mathcal{G}\text{-model} \quad \text{(i.e., a global database that is legal wrt } \mathcal{G}\text{)}$$
$$\text{and is an } \mathcal{M}\text{-model wrt } \mathcal{C} \quad \text{(i.e., satisfies } \mathcal{M} \text{ wrt } \mathcal{C}\text{)} \ \}$$

What it means to satisfy $\mathcal{M}$ wrt $\mathcal{C}$ depends on the nature of the mapping $\mathcal{M}$.

Semantics of queries to $\mathcal{I}$

A **query** q of arity n is a formula with n free variables.

If $\mathcal{D}$ is a database, then $q^{\mathcal{D}}$ denotes the extension of q in $\mathcal{D}$ (i.e., the set of n-tuples that are valuations in Γ for the free variables of q that make q true in $\mathcal{D}$).

If q is a query of arity n posed to a data integration system $\mathcal{I}$ (i.e., a formula over $\mathcal{A}_{\mathcal{G}}$ with n free variables), then the set of **certain answers to q wrt $\mathcal{I}$ and $\mathcal{C}$** is

$$cert(q, \mathcal{I}, \mathcal{C}) = \{(c_1, \ldots, c_n) \in q^{\mathcal{B}} \mid \forall \mathcal{B} \in sem^{\mathcal{C}}(\mathcal{I})\}.$$

Note: query answering is logical implication.

Note: complexity will be mainly measured wrt the size of the source database $\mathcal{C}$, and will refer to the problem of deciding whether $\vec{\mathbf{c}} \in cert(q, \mathcal{I}, \mathcal{C})$, for a given $\vec{\mathbf{c}}$.

Databases with incomplete information, or Knowledge Bases

- Traditional database: one model of a first-order theory

  Query answering means evaluating a formula in the model

- Database with incomplete information, or Knowledge Base: set of models (specified, for example, as a restricted first-order theory)

  Query answering means computing the tuples that satisfy the query in all the models in the set

There is a strong connection between query answering in data integration and query answering in databases with incomplete information under constraints (or, query answering in knowledge bases).

Outline

- Peer-based Distributed Information Systems
- Data integration
  - Approaches to data integration
  - Query answering in different approaches
  - Dealing with inconsistency
- Data exchange
- P2P data integration
- Conclusions

The mapping

How is the mapping $\mathcal{M}$ between $\mathcal{S}$ and $\mathcal{G}$ specified?

- Are the sources defined in terms of the global schema?

  Approach called **source-centric**, or **local-as-view**, or **LAV**

- Is the global schema defined in terms of the sources?

  Approach called **global-schema-centric**, or **global-as-view**, or **GAV**

- A mixed approach?

  Approach called **GLAV**

GAV vs LAV – example

Global schema: movie$(Title, Year, Director)$

european$(Director)$

review$(Title, Critique)$

Source 1: $r_1(Title, Year, Director)$ since 1960, european directors

Source 2: $r_2(Title, Critique)$ since 1990

Query: Title and critique of movies in 1998

$\exists D.\ \text{movie}(T, 1998, D) \wedge \text{review}(T, R)$, written

$\{\ (T, R) \mid \text{movie}(T, 1998, D) \wedge \text{review}(T, R)\ \}$

Formalization of LAV

In LAV (with sound sources), the mapping $\mathcal{M}$ is constituted by a set of assertions:

$$s \quad \rightsquigarrow \quad \phi_{\mathcal{G}}$$

one for each source element s in $\mathcal{A}_{\mathcal{S}}$, where $\phi_{\mathcal{G}}$ is a **query** over $\mathcal{G}$ of the arity of s.

Given source database $\mathcal{C}$, a database $\mathcal{B}$ for $\mathcal{G}$ satisfies $\mathcal{M}$ wrt $\mathcal{C}$ if for each $s \in \mathcal{S}$:

$$s^{\mathcal{C}} \quad \subseteq \quad {\phi_{\mathcal{G}}}^{\mathcal{B}}$$

In other words, the assertion means $\quad \forall \vec{\mathbf{x}} \, (s(\vec{\mathbf{x}}) \rightarrow \phi_{\mathcal{G}}(\vec{\mathbf{x}}))$.

The mapping $\mathcal{M}$ and the source database $\mathcal{C}$ do **not** provide direct information about which data satisfy the global schema. Sources are views, and we have to answer queries on the basis of the available data in the views.

LAV – example

Global schema: movie$(Title, Year, Director)$
european$(Director)$
review$(Title, Critique)$

LAV: associated to source relations we have views over the global schema

$$\mathsf{r}_1(T, Y, D) \rightsquigarrow \{ (T, Y, D) \mid \mathsf{movie}(T, Y, D) \wedge \mathsf{european}(D) \wedge Y \geq 1960 \}$$

$$\mathsf{r}_2(T, R) \rightsquigarrow \{ (T, R) \mid \mathsf{movie}(T, Y, D) \wedge \mathsf{review}(T, R) \wedge Y \geq 1990 \}$$

The query $\{ (T, R) \mid \mathsf{movie}(T, 1998, D) \wedge \mathsf{review}(T, R) \}$ is processed by means of an inference mechanism that aims at re-expressing the atoms of the global schema in terms of atoms at the sources. In this case:

$$\{ (T, R) \mid \mathsf{r}_2(T, R) \wedge \mathsf{r}_1(T, 1998, D) \}$$

Formalization of GAV

In GAV (with sound sources), the mapping $\mathcal{M}$ is constituted by a set of assertions:

$$g \quad \leadsto \quad \phi_{\mathcal{S}}$$

one for each element g in $\mathcal{A}_{\mathcal{G}}$, where $\phi_{\mathcal{S}}$ is a **query** over $\mathcal{S}$ of the arity of g.

Given source database $\mathcal{C}$, a database $\mathcal{B}$ for $\mathcal{G}$ satisfies $\mathcal{M}$ wrt $\mathcal{C}$ if for each $g \in \mathcal{G}$:

$$g^{\mathcal{B}} \quad \supseteq \quad {\phi_{\mathcal{S}}}^{\mathcal{C}}$$

In other words, the assertion means $\quad \forall \vec{\mathbf{x}} \, (\phi_{\mathcal{S}}(\vec{\mathbf{x}}) \rightarrow g(\vec{\mathbf{x}}))$.

Given a source database, $\mathcal{M}$ provides direct information about which data satisfy the elements of the global schema. Relations in $\mathcal{G}$ are views, and queries are expressed over the views. Thus, it seems that we can simply evaluate the query over the data satisfying the global relations (as if we had a single database at hand).

GAV – example

Global schema: movie$(Title, Year, Director)$

european$(Director)$

review$(Title, Critique)$

GAV: associated to relations in the global schema we have views over the sources

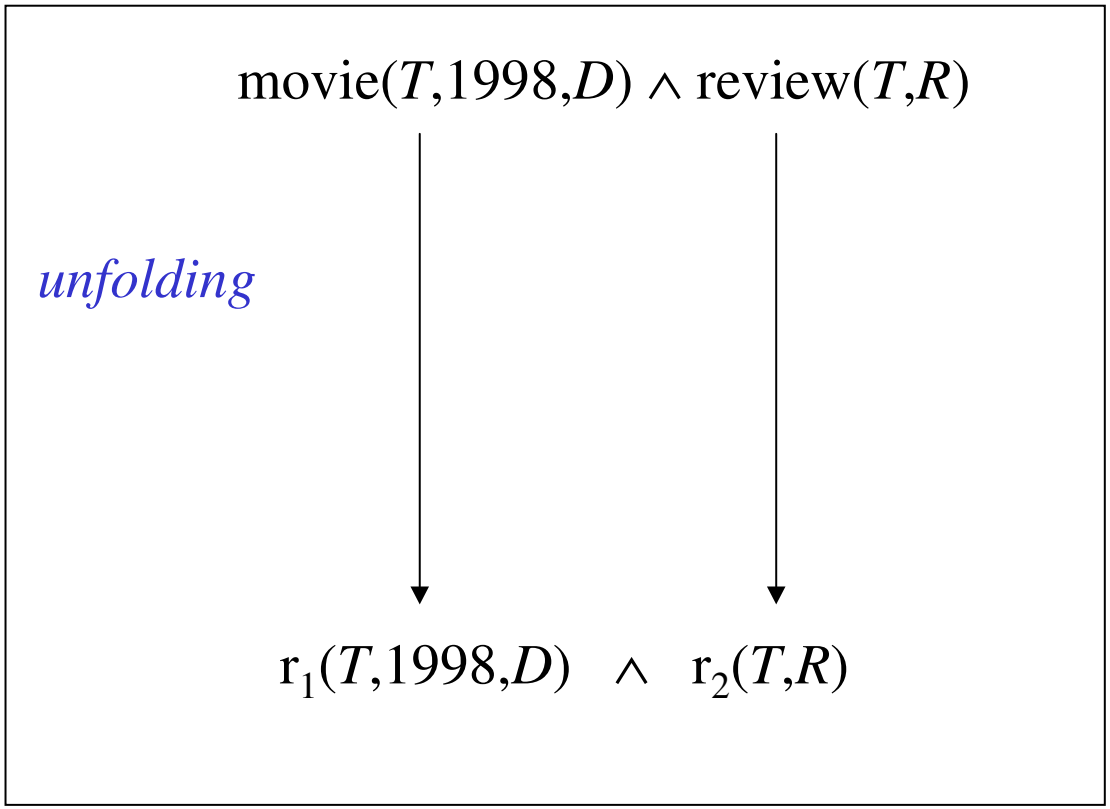
$$\text{movie}(T, Y, D) \rightsquigarrow \{ (T, Y, D) \mid \mathsf{r}_1(T, Y, D) \}$$

$$\text{european}(D) \rightsquigarrow \{ (D) \mid \mathsf{r}_1(T, Y, D) \}$$

$$\text{review}(T, R) \rightsquigarrow \{ (T, R) \mid \mathsf{r}_2(T, R) \}$$

GAV – example of query processing

The query $\{ (T, R) \mid \text{movie}(T, 1998, D) \wedge \text{review}(T, R) \}$ is processed by means of unfolding, i.e., by expanding each atom according to its associated definition in $\mathcal{M}$, so as to come up with source relations. In this case:



GAV and LAV – comparison

LAV: (Information Manifold, DWQ, Picsel)

- Quality depends on how well we have characterized the sources
- High modularity and extensibility (if the global schema is well designed, when a source changes, only its definition is affected)
- Query processing needs reasoning (query reformulation complex)

GAV: (Carnot, SIMS, Tsimmis, IBIS, Picsel, Momis, DisAtDis, . . .)

- Quality depends on how well we have compiled the sources into the global schema through the mapping
- Whenever a source changes or a new one is added, the global schema needs to be reconsidered
- Query processing can be based on some sort of unfolding (query reformulation looks easier)

For more details, see [Ullman TCS'00], [Halevy VLDBJ'01], [Lenzerini PODS'02].

Beyond GAV and LAV: GLAV

In GLAV (with sound sources), the mapping $\mathcal{M}$ is constituted by a set of assertions:

$$\phi_{\mathcal{S}} \quad \rightsquigarrow \quad \phi_{\mathcal{G}}$$

where $\phi_{\mathcal{S}}$ is a **query** over $\mathcal{S}$, and $\phi_{\mathcal{G}}$ is a **query** over $\mathcal{G}$ of the arity $\phi_{\mathcal{S}}$.

Given source database $\mathcal{C}$, a database $\mathcal{B}$ that is legal wrt $\mathcal{G}$ satisfies $\mathcal{M}$ wrt $\mathcal{C}$ if for each assertion in $\mathcal{M}$:

$$\phi_{\mathcal{S}}{}^{\mathcal{C}} \quad \subseteq \quad \phi_{\mathcal{G}}{}^{\mathcal{B}}$$

In other words, the assertion means $\forall \vec{\mathbf{x}} \, (\phi_{\mathcal{S}}(\vec{\mathbf{x}}) \rightarrow \phi_{\mathcal{G}}(\vec{\mathbf{x}}))$.

As for LAV, the mapping $\mathcal{M}$ does **not** provide direct information about which data satisfy the global schema: to answer a query q over $\mathcal{G}$, we have to **infer** how to use $\mathcal{M}$ in order to access the source database $\mathcal{C}$.

Example of GLAV

Global schema: $Work(Person, Project), \quad Area(Project, Field)$

Source 1: $HasJob(Person, Field)$

Source 2: $Teach(Professor, Course), \ In(Course, Field)$

Source 3: $Get(Researcher, Grant), \ For(Grant, Project)$

GLAV mapping:

$$\{ (r, f) \mid HasJob(r, f) \} \rightsquigarrow \{ (r, f) \mid Work(r, p) \wedge Area(p, f) \}$$

$$\{ (r, f) \mid Teach(r, c) \wedge In(c, f) \} \rightsquigarrow \{ (r, f) \mid Work(r, p) \wedge Area(p, f) \}$$

$$\{ (r, p) \mid Get(r, g) \wedge For(g, p) \} \rightsquigarrow \{ (r, p) \mid Work(r, p) \}$$

Outline

- Peer-based Distributed Information Systems
- Data integration
  - Approaches to data integration
  - Query answering in different approaches
  - Dealing with inconsistency
- Data exchange
- P2P data integration
- Conclusions

Query answering in different approaches

The problem of query answering comes in different forms, depending on several parameters:

- **Global schema**
  - without constraints (i.e., empty theory)
  - with constraints
- **Mapping**
  - GAV
  - LAV
- **Queries**
  - user queries
  - queries in the mapping

Two observations

- Unless otherwise specified, we consider **conjunctive queries** (or, unions thereof) as both user queries and queries in the mapping. A conjunctive query has the form

$$\{ (\vec{\mathbf{x}}) \mid \exists \vec{\mathbf{y}} \ p_1(\vec{\mathbf{x}}, \vec{\mathbf{y}}) \wedge \cdots \wedge p_m(\vec{\mathbf{x}}, \vec{\mathbf{y}}) \}$$

- Given a source database $\mathcal{C}$, we call **retrieved global database**, denoted $\mathcal{M}(\mathcal{C})$, the global database obtained by "applying" the queries in the mapping, and "transferring" to the elements of $\mathcal{G}$ the corresponding retrieved tuples.

Incompleteness and inconsistency

Query answering heavily depends upon whether incompleteness/inconsistency shows up.

Constraints in $\mathcal{G}$	Type of mapping	Incompleteness	Inconsistency
no	GAV	**yes**/no	no
no	GLAV	**yes**	no
yes	GAV	**yes**	**yes**
yes	GLAV	**yes**	**yes**

Incompleteness and inconsistency

Constraints in $\mathcal{G}$	Type of mapping	Incompleteness	Inconsistency
no	*GAV*	**yes**/no	no
no	GLAV	**yes**	no
yes	GAV	**yes**	**yes**
yes	GLAV	**yes**	**yes**

INT[noconstr, GAV]: example

Consider $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$, with

Global schema $\mathcal{G}$:

$$\text{student}(code, name, city)$$

$$\text{university}(code, name)$$

$$\text{enrolled}(Scode, Ucode)$$

Source schema $\mathcal{S}$: relations $\mathsf{s}_1(X, Y, W, Z),\ \mathsf{s}_2(X, Y),\ \mathsf{s}_3(X, Y)$

Mapping $\mathcal{M}$:

$$\begin{aligned}
\text{student}(X, Y, Z) &\rightsquigarrow \{\, (X, Y, Z) \mid \mathsf{s}_1(X, Y, Z, W) \,\} \\
\text{university}(X, Y) &\rightsquigarrow \{\, (X, Y) \mid \mathsf{s}_2(X, Y) \,\} \\
\text{enrolled}(X, W) &\rightsquigarrow \{\, (X, W) \mid \mathsf{s}_3(X, W) \,\}
\end{aligned}$$

INT[noconstr, GAV]: example

university

code	name
AF	*bocconi*
BN	*ucla*

student

code	name	city
15	*bill*	*oslo*
12	*anne*	*florence*

enrolled

Scode	Ucode
12	*AF*
16	*BN*

$s_1^{\mathcal{C}}$

12	*anne*	*florence*	21
15	*bill*	*oslo*	24

$s_2^{\mathcal{C}}$

AF	*bocconi*
BN	*ucla*

$s_3^{\mathcal{C}}$

12	AF
16	BN

Example of source database $\mathcal{C}$ and corresponding retrieved global database $\mathcal{M}(\mathcal{C})$

INT[noconstr, GAV]: minimal model

GAV mapping assertions $g \rightsquigarrow \phi_{\mathcal{S}}$ have the logical form:

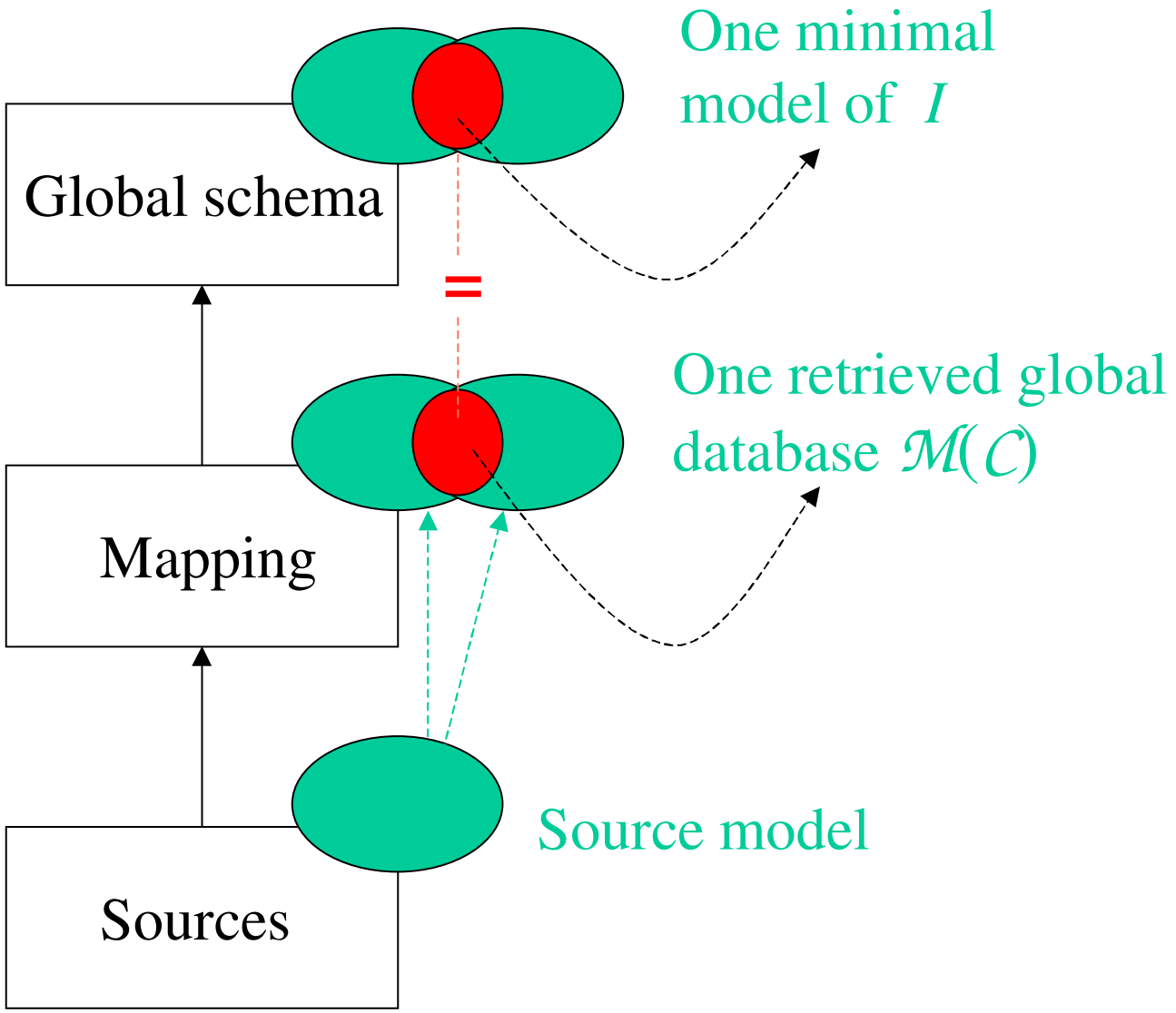
$$\forall \vec{\mathbf{x}} \;\; \phi_{\mathcal{S}}(\vec{\mathbf{x}}) \rightarrow g(\vec{\mathbf{x}})$$

where ϕ_S is a conjunctive query, and g is an element of $\mathcal{G}$.

In general, given a source database $\mathcal{C}$ there are several databases that are legal wrt $\mathcal{G}$ that satisfies $\mathcal{M}$ wrt $\mathcal{C}$.

However, it is easy to see that $\mathcal{M}(\mathcal{C})$ is the intersection of all such databases, and therefore, is the **only** "minimal" model of $\mathcal{I}$.

INT[noconstr, GAV]



INT[noconstr, GAV]: query answering

- If q is a conjunctive query, then $\vec{\mathbf{t}} \in cert(q, \mathcal{I}, \mathcal{C})$ if and only if $\vec{\mathbf{t}} \in q^{\mathcal{M}(\mathcal{C})}$
- If q is query over $\mathcal{G}$, then the unfolding of q wrt $\mathcal{M}$, $unf_{\mathcal{M}}(q)$, is the query over $\mathcal{S}$ obtained from q by substituting every symbol g in q with the query $\phi_{\mathcal{S}}$ that $\mathcal{M}$ associates to g
- It can be shown that evaluating a query q over $\mathcal{M}(\mathcal{C})$ is equivalent to evaluating $unf_{\mathcal{M}}(q)$ over $\mathcal{C}$. It follows that, if q is a conjunctive query, then $\vec{\mathbf{t}} \in cert(q, \mathcal{I}, \mathcal{C})$ if and only if $\vec{\mathbf{t}} \in unf_{\mathcal{M}}(q)^{\mathcal{C}}$

  Unfolding is therefore sufficient
- (Data) complexity of query answering is **polynomial** ($|\mathcal{M}(\mathcal{C})|$ is polynomial wrt $|\mathcal{C}|$)

INT[noconstr, GAV]: example

university

code	name
AF	*bocconi*
BN	*ucla*

student

code	name	city
15	*bill*	*oslo*
12	*anne*	*florence*

{ x | student(15,x,y) }

unfolding

$s_1^{\mathcal{C}}$

12	*anne*	*florence*	21
15	*bill*	*oslo*	24

$s_2^{\mathcal{C}}$

AF	*bocconi*
BN	*ucla*

$\{ x \mid s_1(15, x, y, z) \}$

INT[noconstr, GAV]: more expressive queries?

- More expressive queries in the mapping?
  - Same results hold if we use any computable query in the mapping
- More expressive user queries?
  - Same results hold if we use Datalog queries as user queries
  - Same results hold if we use union of conjunctive queries with inequalities as user queries

INT[noconstr, GAV]: another view

Let B_1 and B_2 be two global databases with values in $\Gamma \cup$ Var.

- A homomorphism $h : B_1 \rightarrow B_2$ is a mapping from ($\Gamma \cup$ Var(B_1)) to ($\Gamma \cup$ Var(B_2)) such that
  1. $h(c) = c$, for every $c \in \Gamma$
  2. for every fact $R_i(t)$ of B_1, we have that $R_i(h(t))$ is a fact in B_2 (where, if $t = (a_1, \ldots, a_n)$, then $h(t) = (h(a_1), \ldots, h(a_n))$
- B_1 is homomorphically equivalent to B_2 if there is a homomorphism $h : B_1 \rightarrow B_2$ and a homomorphism $h' : B_2 \rightarrow B_1$

Let $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$ be a data integration system. If $\mathcal{C}$ is a source database, then a universal solution for $\mathcal{I}$ relative to $\mathcal{C}$ is a model J of $\mathcal{I}$ relative to $\mathcal{C}$ such that for every model J' of $\mathcal{I}$ relative to $\mathcal{C}$, there exists a homomorphism $h : J \rightarrow J'$ (see [Fagin&al. ICDT'03]).

INT[noconstr, GAV]: another view

- Homomorphism preserves satisfaction of conjunctive queries: if there exists a homomorphism $h : J \longrightarrow J'$, and q is a conjunctive query, then $\vec{\mathbf{t}} \in q^J$ implies $\vec{\mathbf{t}} \in q^{J'}$
- Let $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$ be a GAV data integration system without constraints in the global schema. If $\mathcal{C}$ is a source database, then $\mathcal{M}(\mathcal{C})$ is the minimal universal solution for $\mathcal{I}$ relative to $\mathcal{C}$
- We derive again the following results
  - if q is a conjunctive query, then $\vec{\mathbf{t}} \in cert(q, \mathcal{I}, \mathcal{C})$ if and only if $\vec{\mathbf{t}} \in q^{\mathcal{M}(\mathcal{C})}$
  - complexity of query answering is polynomial

Incompleteness and inconsistency

Constraints in $\mathcal{G}$	Type of mapping	Incompleteness	Inconsistency
no	GAV	**yes**/no	no
no	*GLAV*	**yes**	no
yes	GAV	**yes**	**yes**
yes	GLAV	**yes**	**yes**

INT[noconstr, GLAV]: example

Consider $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$, with

Global schema $\mathcal{G}$:

$$\text{student}(code, name, city)$$

$$\text{enrolled}(Scode, Ucode)$$

Source schema $\mathcal{S}$: relation $\text{s}_1(X, Y, W, Z)$

Mapping $\mathcal{M}$:

$$\{ (X, Y, Z) \mid \text{s}_1(X, Y, Z, W) \} \rightsquigarrow \{ (X, Y, Z) \mid \text{student}(X, Y, Z) \wedge \text{enrolled}(X, W) \}$$

INT[noconstr, GLAV]: example

$\{ (X, Y, Z) \mid s_1(X, Y, Z, W) \} \quad \rightsquigarrow \quad \{ (X, Y, Z) \mid \text{student}(X, Y, Z) \wedge \text{enrolled}(X, W) \}$

student

code	name	city
15	*bill*	*oslo*
12	*anne*	*florence*

enrolled

Scode	Ucode
15	***x***
12	***y***

$s_1{}^{\mathcal{C}}$

15	*bill*	*oslo*	24
12	*anne*	*florence*	21

Example of source database $\mathcal{C}$ and corresponding retrieved global database $\mathcal{M}(\mathcal{C})$

INT[noconstr, GLAV]: example

$\{ (X, Y, Z) \mid s_1(X, Y, Z, W) \} \quad \rightsquigarrow \quad \{ (X, Y, Z) \mid \text{student}(X, Y, Z) \wedge \text{enrolled}(X, W) \}$

student

code	name	city
15	*bill*	*oslo*
12	*anne*	*florence*

enrolled

Scode	Ucode
15	*x*
12	*x*

$s_1{}^{\mathcal{C}}$

15	*bill*	*oslo*	24
12	*anne*	*florence*	21

Example of source database $\mathcal{C}$ and corresponding retrieved global database $\mathcal{M}(\mathcal{C})$

INT[noconstr, GLAV]: incompleteness

GLAV mapping assertions $\phi_{\mathcal{S}} \rightsquigarrow \phi_{\mathcal{G}}$ have the logical form:

$$\forall \vec{\mathbf{x}} \ \phi_{\mathcal{S}}(\vec{\mathbf{x}}) \longrightarrow \exists \vec{\mathbf{y}} \phi_{\mathcal{G}}(\vec{\mathbf{x}}, \vec{\mathbf{y}})$$

where $\phi_{\mathcal{S}}$ and $\phi_{\mathcal{G}}$ are conjunctions of atoms.

In general, given a source database $\mathcal{C}$ there are several solutions for a set of assertions of the above form (i.e., different databases that are legal wrt $\mathcal{G}$ that satisfies $\mathcal{M}$ wrt $\mathcal{C}$): **incompleteness comes from the mapping**.

This holds even for the case of very simple queries $\phi_{\mathcal{G}}$:

$$s_1(x) \quad \rightsquigarrow \quad \{ \ (x) \mid \exists y \ g(x, y) \ \}$$

INT[noconstr, GLAV]: canonical retrieved global database

What is a retrieved global database in this case?

We build what we call the **canonical retrieved global database** for $\mathcal{I}$ relative to $\mathcal{C}$, denoted $\mathcal{M}(\mathcal{C})\downarrow$, as follows:

- let all predicates be empty in $\mathcal{M}(\mathcal{C})\downarrow$
- for each mapping assertion $\phi_{\mathcal{S}} \rightsquigarrow \phi_{\mathcal{G}}$ in $\mathcal{M}$
  - for each tuple $\vec{\mathbf{t}} \in \phi_{\mathcal{S}}^{\mathcal{C}}$ such that $\vec{\mathbf{t}} \notin \phi_{\mathcal{G}}^{\mathcal{M}(\mathcal{C})\downarrow}$, add $\vec{\mathbf{t}}$ to $\phi_{\mathcal{G}}^{\mathcal{M}(\mathcal{C})\downarrow}$ by inventing fresh variables (Skolem terms) in order to satisfy the existentially quantified variables in $\phi_{\mathcal{G}}$

There is a unique (up to variable renaming) canonical retrieved global database for $\mathcal{I}$ relative to $\mathcal{C}$, that can be computed in polynomial time wrt the size of $\mathcal{C}$. $\mathcal{M}(\mathcal{C})\downarrow$ obviously satisfies $\mathcal{G}$, and is also called the **canonical model of $\mathcal{I}$ relative to $\mathcal{C}$**.

INT[noconstr, GLAV]: example of canonical model

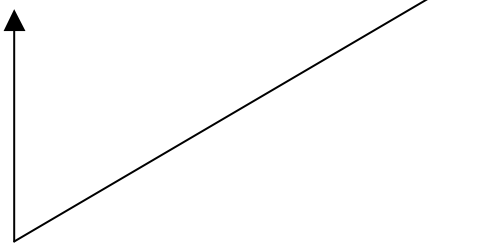
$\{ (X, Y, Z) \mid s_1(X, Y, Z, W) \} \quad \rightsquigarrow \quad \{ (X, Y, Z) \mid \text{student}(X, Y, Z) \wedge \text{enrolled}(X, W) \}$

student

code	name	city
15	*bill*	*oslo*
12	*anne*	*florence*

enrolled

Scode	Ucode
15	***x***
12	***y***

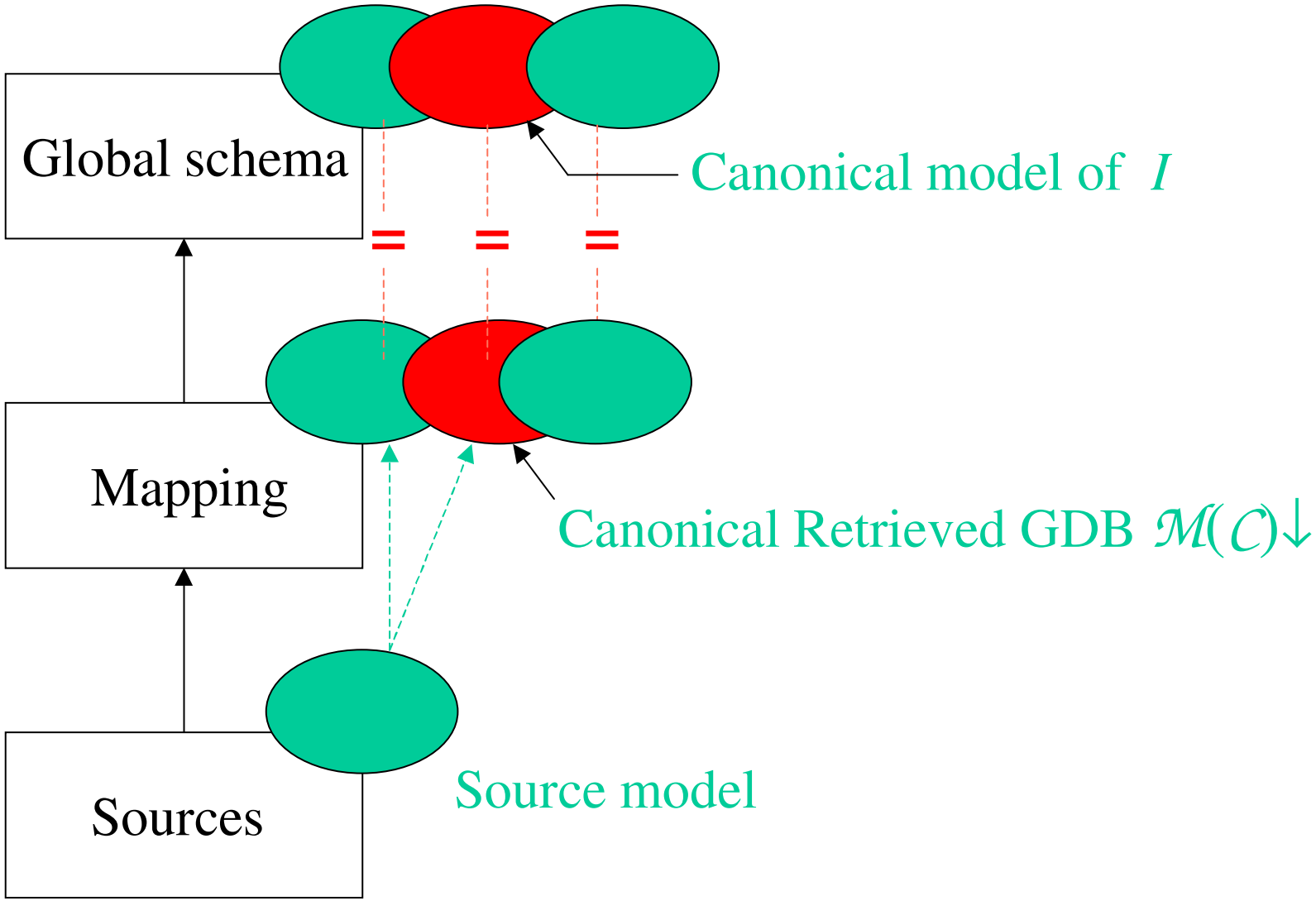


$s_1{}^{\mathcal{C}}$

15	*bill*	*oslo*	24
12	*anne*	*florence*	21

Example of source database $\mathcal{C}$ and corresponding canonical model $\mathcal{M}(\mathcal{C})\downarrow$

INT[noconstr, GLAV]: universal solution



INT[noconstr, GLAV]: universal solution

Let $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$ be a GLAV data integration system without constraints in the global schema. If $\mathcal{C}$ is a source database, then $\mathcal{M}(\mathcal{C})\downarrow$ is a **universal solution** for $\mathcal{I}$ relative to $\mathcal{C}$ (follows from [Fagin&al. ICDT'03]).

It follows that:

- if q is a conjunctive query, then $\vec{\mathbf{t}} \in cert(q, \mathcal{I}, \mathcal{C})$ if and only if $\vec{\mathbf{t}} \in q^{\mathcal{M}(\mathcal{C})\downarrow}$
- complexity of query answering is **polynomial**

INT[noconstr, GLAV]: algorithms

- Inverse rules [Duschka&Genesereth PODS'97]
- Bucket algorithm [Levy&al. AAAI'96]
- MiniCon algorithm [Pottinger&Levy VLDB'00]

Other works on LAV:

- Conjunctive queries using conjunctive views [Levy&al. PODS'95]
- Recursive queries (datalog programs) using conjunctive views [Duschka&Genesereth PODS'97], [Afrati&al. ICDT'99]
- Conjunctive queries with arithmetic comparison [Afrati&al. PODS'01]
- Complexity analysis [Abiteboul&Duschka PODS'98] [Grahne&Mendelzon ICDT'99]
- Variants of Regular Path Queries [Calvanese&al. ICDE'00, PODS'00] [Deutsch&Tannen DBPL'01], [Calvanese&al. DBPL'01]

INT[noconstr, GLAV]: basic technique

From [Duschka&Genesereth PODS'97]:

$$\mathsf{r}_1(T) \quad \leadsto \quad \{\ (T) \mid \mathsf{movie}(T, Y, D) \wedge \mathsf{european}(D)\ \}$$

$$\mathsf{r}_2(T, V) \quad \leadsto \quad \{\ (T, V) \mid \mathsf{movie}(T, Y, D) \wedge \mathsf{review}(T, V)\ \}$$

$$\forall T\ \mathsf{r}_1(T) \quad \rightarrow \quad \exists Y \exists D\ \mathsf{movie}(T, Y, D) \wedge \mathsf{european}(D)$$

$$\forall T\ \forall V\ \mathsf{r}_2(T, V) \quad \rightarrow \quad \exists Y \exists D\ \mathsf{movie}(T, Y, D) \wedge \mathsf{review}(T, V)$$

$$\mathsf{movie}(T, f_1(T), f_2(T)) \quad \leftarrow \quad \mathsf{r}_1(T)$$

$$\mathsf{european}(f_2(T)) \quad \leftarrow \quad \mathsf{r}_1(T)$$

$$\mathsf{movie}(T, f_4(T, V), f_5(T, V)) \quad \leftarrow \quad \mathsf{r}_2(T, V)$$

$$\mathsf{review}(T, V) \quad \leftarrow \quad \mathsf{r}_2(T, V)$$

- Answering a query means evaluating a goal wrt to this nonrecursive logic program (that can be transformed into a union of conjunctive query)
- PTIME data complexity

INT[noconstr, GLAV]: more expressive queries?

- More expressive source queries in the mapping?
  - Same results hold if we use any computable query as source query in the mapping assertions
- More expressive queries over the global schema in the mapping?
  - Already positive queries lead to intractability
- More expressive user queries?
  - Same results hold if we use Datalog queries as user queries
  - Even the simplest form of negation (inequalities) leads to intractability

INT[noconstr, GLAV]: data complexity

From [Abiteboul&Duschka PODS'98]:

Sound sources	CQ	$CQ^{\neq}$	PQ	datalog	FOL
CQ	*PTIME*	*coNP*	*PTIME*	*PTIME*	*undec.*
$CQ^{\neq}$	*PTIME*	*coNP*	*PTIME*	*PTIME*	*undec.*
PQ	*coNP*	*coNP*	*coNP*	*coNP*	*undec.*
datalog	*coNP*	*undec.*	*coNP*	*undec.*	*undec.*
FOL	*undec.*	*undec.*	*undec.*	*undec.*	*undec.*

INT[noconstr, GLAV]: intractability for positive queries and views

From [Calvanese&al. ICDE'00], given a graph $G = (N, E)$, we define $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$ and source database $\mathcal{C}$, with $\mathcal{S} = \{V_b, V_f, V_e\}$, and $\mathcal{G} = \{R_b, R_f, R_{rg}, R_{gr}, R_{rb}, R_{br}, R_{gb}, R_{bg}\}$

$\mathcal{M}$:

$V_b \rightsquigarrow R_b$

$V_f \rightsquigarrow R_f$

$V_e \rightsquigarrow R_{rg} \vee R_{gr} \vee R_{rb} \vee R_{br} \vee R_{gb} \vee R_{bg}$

$\mathcal{C}$:

$V_b{}^{\mathcal{C}} = \{(c, a) \mid a \in N, c \notin N\}$

$V_f{}^{\mathcal{C}} = \{(a, d) \mid a \in N, d \notin N\}$

$V_e{}^{\mathcal{C}} = \{(a, b), (b, a) \mid (a, b) \in E\}$

Query Q : $\{ (X, Z) \mid R_b(X, Y) \wedge M(Y, W) \wedge R_f(W, Z)\}$

where M describes all mismatched edge pairs (e.g., $\{(X, Z) \mid R_{rg}(X, Y) \wedge R_{rb}(Y, Z)\}$).

- If G is 3-colorable, then $\exists \mathcal{B}$ where M (and Q) is empty, i.e. $(c, d) \notin cert(Q, \mathcal{I}, \mathcal{C})$
- If G is not 3-colorable, then M is nonempty $\forall \mathcal{B}$, i.e. $(c, d) \in cert(Q, \mathcal{I}, \mathcal{C})$

$\Longrightarrow$ **coNP-hard data complexity** for positive queries and positive views.

INT[noconstr, GLAV]: in coNP for positive queries and views

In the case of positive queries and positive views:

- $\vec{\mathbf{t}} \notin cert(Q, \mathcal{I}, \mathcal{C})$ if and only if there is a database $\mathcal{B}$ for $\mathcal{I}$ such that $\vec{\mathbf{t}} \notin Q^{\mathcal{B}}$, and $\mathcal{B}$ satisfies $\mathcal{M}$ wrt $\mathcal{C}$
- Because of the form of $\mathcal{M}$

$$\forall \vec{\mathbf{x}} \left(\phi_{\mathcal{S}}(\vec{\mathbf{x}}) \rightarrow \exists \vec{\mathbf{y_1}} \alpha_1(\vec{\mathbf{x}}, \vec{\mathbf{y_1}}) \vee \ldots \vee \exists \vec{\mathbf{y_h}} \alpha_h(\vec{\mathbf{x}}, \vec{\mathbf{y_h}})\right)$$

  each tuple in $\mathcal{C}$ forces the existence of k tuples in any database that satisfies $\mathcal{M}$ wrt $\mathcal{C}$, where k is the maximal length of conjuncts in $\mathcal{M}$
- If $\mathcal{C}$ has n tuples, then there is a database $\mathcal{B}' \subseteq \mathcal{B}$ for $\mathcal{I}$ that satisfies $\mathcal{M}$ wrt $\mathcal{C}$ with at most $n \cdot k$ tuples. Since Q is monotone, $\vec{\mathbf{t}} \notin Q^{\mathcal{B}'}$
- Checking whether $\mathcal{B}'$ satisfies $\mathcal{M}$ wrt $\mathcal{C}$, and checking whether $\vec{\mathbf{t}} \notin Q^{\mathcal{B}'}$ can be done in PTIME wrt the size of $\mathcal{B}'$

$\Longrightarrow$ **coNP data complexity** for positive queries and positive views.

INT[noconstr, GLAV]: conjunctive user queries with inequalities

Consider the following $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$ and the following query Q (from [Fagin&al. ICDT'03]):

$$\begin{array}{lcl} \mathcal{M} & : & s(X,Y) \rightsquigarrow \{\ (X,Y) \mid T(X,Z) \wedge T(Z,Y)\ \} \\ \mathcal{C} & : & \{\ s(a,a)\ \} \\ Q & : & \{\ (\ ) \mid T(X,Y)\ \wedge\ X \neq Y)\ \} \end{array}$$

- $J_1 = \{T(a,a)\}$ is a solution, and $Q^{J_1} = false$
- if J is a universal solution, then both $T(a,X)$ and $T(X,a)$ are in J, with $X \neq a$ (otherwise $T(a,a)$ would be true in every solution)

$\Longrightarrow$ $cert(Q,\mathcal{I},\mathcal{C}) = false$, but $Q^J = true$ for every universal solution J for $\mathcal{I}$ relative to $\mathcal{C}$

$\Longrightarrow$ the notion of universal solution is not the right tool

INT[noconstr, GLAV]: conjunctive user queries with inequalities

- still polynomial with one inequalities
- coNP algorithm: guess equalities on variables in the canonical retrieved global database
- coNP-hard with six inequalities (see [Abiteboul&Duschka PODS'98])
- open problem for a number of inequalities between two and five

$\Longrightarrow$ **coNP-complete** for conjunctive user queries with inequalities.

INT[noconstr, GLAV]: connection to view-based query processing

View-based query processing: Answer a query based on a set of materialized views, rather than on the raw data in the database.

In GLAV data integration, **the views are the sources**.

Two approaches to view-based query processing:

- **View-based query rewriting**: query processing is divided in two steps
  1. re-express the query in terms of a **given query language** over the alphabet of $\mathcal{A}_{\mathcal{S}}$
  2. evaluate the rewriting over the source database $\mathcal{C}$
- **View-based query answering**: no limitation is posed on how queries are processed, and the only goal is to exploit all possible information, in particular the source database, to compute the certain answers to the query

INT[noconstr, LAV]: connection to rewriting

Query answering by rewriting:

- Given $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$, and given a query Q over $\mathcal{G}$, rewrite Q into a query, called $rew(Q, \mathcal{I})$, in the alphabet $\mathcal{A}_\mathcal{S}$ of the sources
- Evaluate the rewriting $rew(Q, \mathcal{I})$ over the source database

We are interested in sound rewritings (i.e., computing only tuples in $cert(Q, \mathcal{I}, \mathcal{C})$ for every source database $\mathcal{C}$) that are expressed in a given query language, and that are maximal for the class of queries expressible in such language. Sometimes, we are interested in exact rewritings, i.e., rewritings that are logically equivalent to the query, modulo $\mathcal{M}$.

But:

- *When does the rewriting compute all certain answers?*
- *What do we gain or loose by focusing on a given class of queries?*

Perfect rewriting

Define $cert_{[Q,\mathcal{I}]}(\cdot)$ to be the function that, with Q and $\mathcal{I}$ fixed, given source database $\mathcal{C}$, computes the certain answers $cert(Q, \mathcal{I}, \mathcal{C})$.

- $cert_{[Q,\mathcal{I}]}$ can be seen as a query on the alphabet $\mathcal{A}_\mathcal{S}$
- $cert_{[Q,\mathcal{I}]}$ is a (*sound*) rewriting of Q wrt $\mathcal{I}$
- No sound rewriting exists that is better than $cert_{[Q,\mathcal{I}]}$
- $cert_{[Q,\mathcal{I}]}$ is called the **perfect rewriting** of Q wrt $\mathcal{I}$

Properties of the perfect rewriting

- Can we express the perfect rewriting in a certain query language?
- How does a maximal rewriting for a given class of queries compare with the perfect rewriting?
  - From a semantical point of view
  - From a computational point of view
- Which is the computational complexity of (finding, evaluating) the perfect rewriting?

The case of conjunctive queries

Let $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$ be a GLAV data integration system, let Q and the queries in $\mathcal{M}$ be conjunctive queries (CQs), and let Q' be the **union of all maximal rewritings of Q for the class of CQs**. Then ([Levy&al. PODS'95], [Abiteboul&Duschka PODS'98]

- Q' is the maximal rewriting for the class of unions of conjunctive queries (UCQs)
- **Q' is the perfect rewriting of Q wrt $\mathcal{I}$**
- Q' is a PTIME query
- Q' is an exact rewriting (equivalent to Q for each database $\mathcal{B}$ of $\mathcal{I}$), if an exact rewriting exists

Does this "ideal situation" carry on to cases where Q and $\mathcal{M}$ allow for union?

View-based query processing for UPQs

As we saw before, view-based query answering is coNP-complete in data complexity when we add (a very simple form of) union to the query language used to express queries over the global schema in the mapping [Calvanese&al. ICDE'00].

In other words, in this case $cert(Q, \mathcal{I}, \mathcal{C})$, with Q and $\mathcal{I}$ fixed, is a coNP-complete function, and therefore **the perfect rewriting $cert_{[Q,\mathcal{I}]}$ is a coNP-complete query**.

If in the mapping we use a query language with union, then the perfect rewriting is coNP-hard — we do not have the ideal situation we had for conjunctive queries.

Incompleteness and inconsistency

Constraints in $\mathcal{G}$	Type of mapping	Incompleteness	Inconsistency
no	GAV	**yes**/no	no
no	GLAV	**yes**	no
yes	*GAV*	**yes**	**yes**
yes	GLAV	**yes**	**yes**

INT[constr, GAV]: incompleteness and inconsistency

Let us consider a system with a global schema with constraints, and with a GAV mapping $\mathcal{M}$ with sound sources, whose assertions $g \rightsquigarrow \phi_{\mathcal{S}}$ have the logical form

$$\forall \vec{\mathbf{x}} \;\; \phi_{\mathcal{S}}(\vec{\mathbf{x}}) \rightarrow g(\vec{\mathbf{x}})$$

where ϕ_S is a conjunctive query, and g is an element of $\mathcal{G}$.

Basic observation: **since $\mathcal{G}$ does have constraints, the retrieved global database may not be legal for $\mathcal{G}$.**

INT[constr, GAV]: example

Global schema $\mathcal{G}$:

student$(Scode, Sname, Scity)$, $\quad key\{Scode\}$

university$(Ucode, Uname)$, $\quad key\{Ucode\}$

enrolled$(Scode, Ucode)$, $\quad key\{Scode, Ucode\}$

$$\text{enrolled}[Scode] \subseteq \text{student}[Scode]$$

$$\text{enrolled}[Ucode] \subseteq \text{university}[Ucode]$$

Sources $\mathcal{S}$: database relations $\mathsf{s}_1(X, Y, Z), \mathsf{s}_2(X, Y), \mathsf{s}_3(X, Y)$

Mapping $\mathcal{M}$:

$$\text{student}(X, Y, Z) \rightsquigarrow \{ (X, Y, Z) \mid \mathsf{s}_1(X, Y, Z, W) \}$$

$$\text{university}(X, Y) \rightsquigarrow \{ (X, Y) \mid \mathsf{s}_2(X, Y) \}$$

$$\text{enrolled}(X, Y) \rightsquigarrow \{ (X, Y) \mid \mathsf{s}_3(X, Y) \}$$

Constraints in GAV: example

University

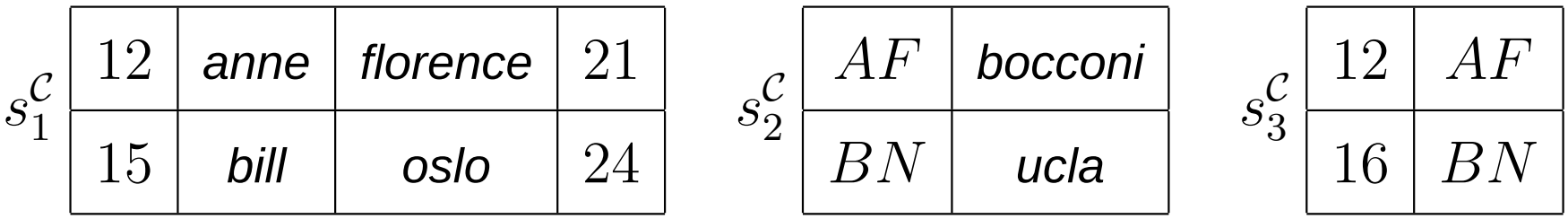
code	name
AF	*bocconi*
BN	*ucla*

Student

code	name	city
15	*bill*	*oslo*
12	*anne*	*florence*

Enrolled

Scode	Ucode
12	*AF*
16	*BN*



Example of source database and corresponding retrieved global database

Constraints in GAV: example of incompleteness

Source database $\mathcal{C}$:

$s_1^{\mathcal{C}}$

12	*anne*	*florence*	21
15	*bill*	*oslo*	24

$s_2^{\mathcal{C}}$

AF	*bocconi*
BN	*ucla*

$s_3^{\mathcal{C}}$

12	AF
16	BN

$s_3^{\mathcal{C}}(16, BN)$ and the mapping imply enrolled$^{\mathcal{B}}(16, BN)$, for all $\mathcal{B} \in sem^{\mathcal{C}}(\mathcal{I})$.

Due to the integrity constraints in the global schema, 16 **is the code of some student** in all $\mathcal{B} \in sem^{\mathcal{C}}(\mathcal{I})$.

Since $\mathcal{C}$ says nothing about the name and the city of the student with code 16, we must accept as legal for $\mathcal{I}$ wrt $\mathcal{C}$ all virtual global databases that differ in such attributes.

INT[constr, GAV]: unfolding is not sufficient

Mapping $\mathcal{M}$:

$$\text{student}(X, Y, Z) \rightsquigarrow \{ (X, Y, Z) \mid \mathsf{s}_1(X, Y, Z, W) \}$$

$$\text{university}(X, Y) \rightsquigarrow \{ (X, Y) \mid \mathsf{s}_2(X, Y) \}$$

$$\text{enrolled}(X, Y) \rightsquigarrow \{ (X, Y) \mid \mathsf{s}_3(X, Y) \}$$

$s_1^{\mathcal{C}}$

12	*anne*	*florence*	21
15	*bill*	*oslo*	24

$s_2^{\mathcal{C}}$

AF	*bocconi*
BN	*ucla*

$s_3^{\mathcal{C}}$

12	AF
16	BN

Query: $\{ (X) \mid \text{student}(X, Y, Z), \text{enrolled}(X, W) \}$

Unfolding wrt $\mathcal{M}$: $\{ (X) \mid s_1(X, Y, Z, V), s_3(X, W) \}$

retrieves only the answer $\{12\}$ from $\mathcal{C}$, although $\{12, 16\}$ is the correct answer. The simple unfolding strategy is **not sufficient** in our context.

Most GAV systems use the simple unfolding strategy!

Constraints in GAV: example of inconsistency

Source database $\mathcal{C}$:

$s_1^{\mathcal{C}}$

12	*anne*	*florence*	21
12	*bill*	*oslo*	24

$s_2^{\mathcal{C}}$

AF	*bocconi*
BN	*ucla*

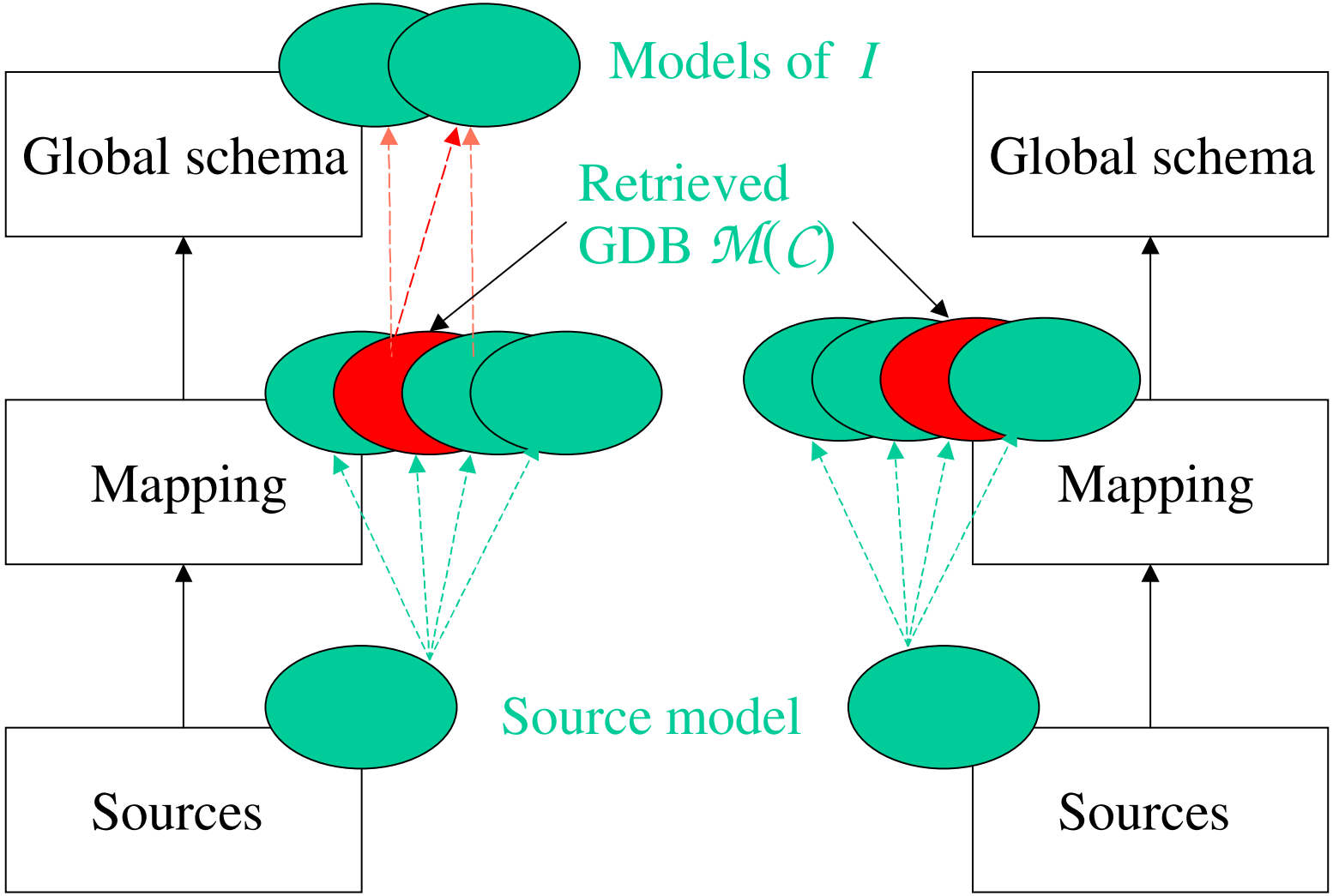
$s_3^{\mathcal{C}}$

12	AF
16	BN

$s_1^{\mathcal{C}}$ imply student$^{\mathcal{B}}(12, \textit{anne}, \textit{florence}, 21)$, and student$^{\mathcal{B}}(12, \textit{bill}, \textit{oslo}, 24)$, for all $\mathcal{B}$ that satisfies the mapping.

Due to the integrity constraints in the global schema, it follows that there is **no database that satisfies both the mapping and the global schema**, i.e., $sem^{\mathcal{C}}(\mathcal{I}) = \emptyset$.

INT[constr, GAV]: incompleteness and inconsistency



INT[constr, GAV]: the case of key and foreign key

We consider the case where the global schema contains

- key constraints: every relation has one key constituted by a set of attributes
- foreign key constraints, which are inclusion dependencies of the form

$$r_1[X_1, \ldots, X_n] \subseteq r_2[Y_1, \ldots, Y_n]$$

  where $\{Y_1, \ldots, Y_n\}$ is a subset of the key of r_2

It can be shown that such constraints are sufficient to capture:

- Frame-based languages
- The class definition part of most object-oriented languages (including UML)
- Semantic and conceptual data models (inculding the Extended Entity-Relationship model)
- Several ontology languages adopted in the research work on Semantic Web

INT[constr, GAV]: the case of key and foreign key

Given source database $\mathcal{C}$,

- If $\mathcal{M}(\mathcal{C})$ does violate key constraints, then $sem^{\mathcal{C}}(\mathcal{I}) = \emptyset$, and we are done (see later, for the case where violations are treated in different ways).
- Otherwise, we "chase" $\mathcal{M}(\mathcal{C})$ by means of the foreign key constraints $\mathcal{G}$ and we obtain a (possibly infinite) database for $\mathcal{G}$, denoted $chase_{\mathcal{G}}(\mathcal{M}(\mathcal{C}))$.

  Example of chase: we "apply" the foreign key constraint

  $$\text{enrolled}[Scode] \subseteq \text{student}[Scode]$$

  to $\{\text{enrolled}(16, BN), \text{student}(12, \textit{anne}, \textit{florence})\}$ and we obtain

  $$\{\text{enrolled}(16, BN)\}, \text{student}(12, \textit{anne}, \textit{florence}), \text{student}(16, x, y))\}$$

INT[constr, GAV]: special case

Properties of $chase_{\mathcal{G}}(\mathcal{M}(\mathcal{C}))$:

- $chase_{\mathcal{G}}(\mathcal{M}(\mathcal{C}))$ does not violate key constraints (if $\mathcal{M}(\mathcal{C})$ does violate key constraints)
- $chase_{\mathcal{G}}(\mathcal{M}(\mathcal{C}))$ may be infinite (in particular, when foreign key constraints are cyclic)
- $chase_{\mathcal{G}}(\mathcal{M}(\mathcal{C}))$ represents a **universal solution** for $\mathcal{I}$ and $\mathcal{C}$, i.e., for every database $\mathcal{B} \in sem^{\mathcal{C}}(\mathcal{I})$, there exists a homomorphism from $chase_{\mathcal{G}}(\mathcal{M}(\mathcal{C}))$ to $\mathcal{B}$
- if $\mathcal{M}(\mathcal{C})$ does violate key constraints, and q is a conjunctive query, then $\vec{\mathbf{t}} \in cert(q, \mathcal{I}, \mathcal{C})$ if and only if $t \in q^{chase_{\mathcal{G}}(\mathcal{M}(\mathcal{C}))}$

INT[constr, GAV]: special case

Techniques for processing a conjunctive query q posed to $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$:

1. We construct from $\mathcal{G}$ a suitable logic program $\mathcal{P}_{\mathcal{G}}$
2. We partially evaluate $\mathcal{P}_{\mathcal{G}}$ wrt q and $\mathcal{G}$, and obtain another query $exp_{\mathcal{G}}(q)$, called the expansion of q wrt $\mathcal{G}$
3. We unfold $exp_{\mathcal{G}}(q)$ wrt $\mathcal{M}$, and obtain a query $unf_{\mathcal{M}}(exp_{\mathcal{G}}(q))$ over the sources
4. We evaluate $unf_{\mathcal{M}}(exp_{\mathcal{G}}(q))$ over the source database $\mathcal{C}$

- Evaluating $unf_{\mathcal{M}}(exp_{\mathcal{G}}(q))$ over $\mathcal{C}$ is equivalent to evaluating q over $chase_{\mathcal{G}}(\mathcal{M}(\mathcal{C}))$, i.e., it computes $cert(q, \mathcal{I}, \mathcal{C})$
- $exp_{\mathcal{G}}(q)$ can be of exponential size wrt $\mathcal{G}$, but the whole process has polynomial time complexity wrt the size of $\mathcal{C}$.

INT[constr, GAV]: example

Suppose we have $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$, with $\mathcal{G}$:

$$\text{person}(Pcode, Age, CityOfBirth)$$

$$\text{student}(Scode, University)$$

$$\text{city}(Name, Major)$$

$$\begin{aligned} key(\text{person}) &= \{Pcode\} \\ key(\text{student}) &= \{Scode\} \\ key(\text{city}) &= \{Name\} \\ \text{person}[CityOfBirth] &\subseteq \text{city}[Name] \\ \text{city}[Major] &\subseteq \text{person}[PCode] \\ \text{student}[SCode] &\subseteq \text{person}[PCode] \end{aligned}$$

INT[constr, GAV]: example

The logic program $\mathcal{P}_{\mathcal{G}}$ is

$$
\begin{aligned}
\text{person}'(X, Y, Z) &\leftarrow \text{person}(X, Y, Z) \\
\text{student}'(X, Y) &\leftarrow \text{student}(X, Y) \\
\text{city}'(X, Y) &\leftarrow \text{city}(X, Y) \\
\text{city}'(X, f_1(X)) &\leftarrow \text{person}'(Y, Z, X) \\
\text{person}'(Y, f_2(Y), f_3(Y)) &\leftarrow \text{city}'(X, Y) \\
\text{person}'(X, f_4(X), f_5(X)) &\leftarrow \text{student}'(X, Y)
\end{aligned}
$$

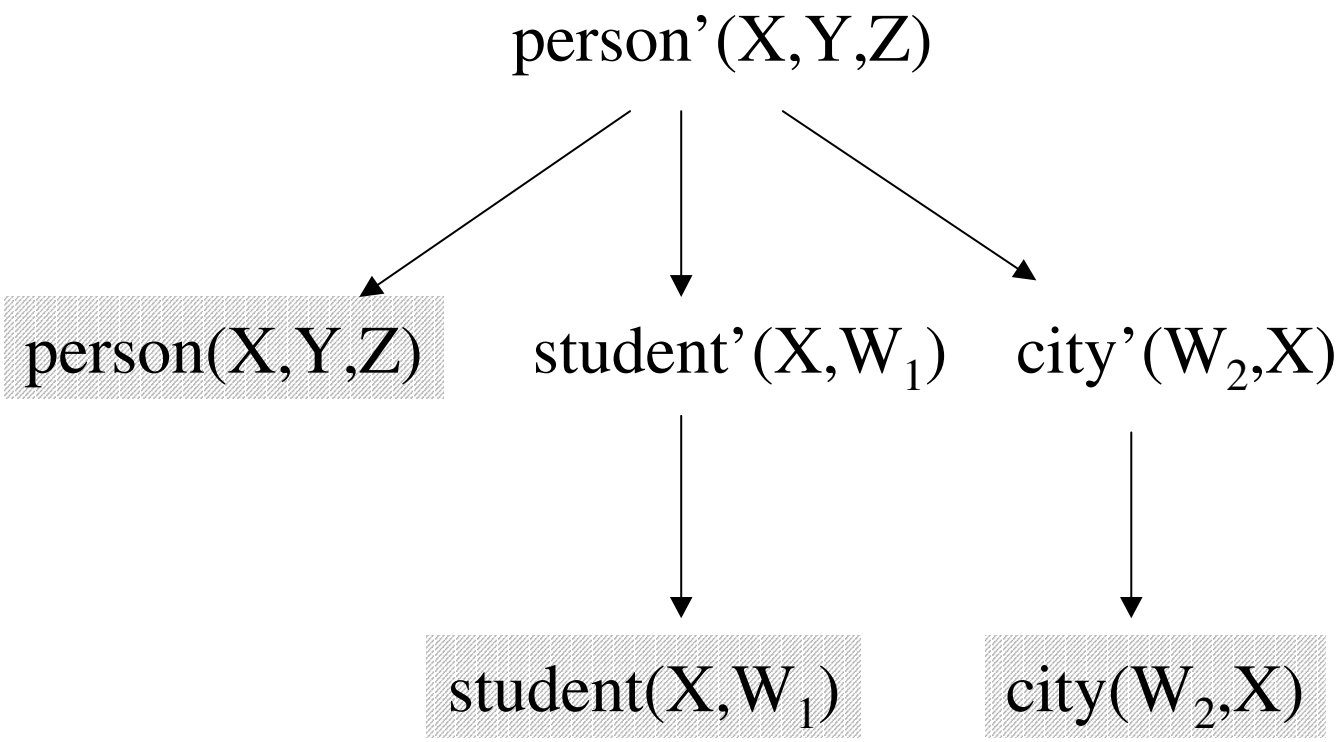
Consider the query

$$\{ (X) \mid \text{person}(X, Y, Z) \}$$

written as the rule

$$\text{q}(X) \leftarrow \text{person}'(X, Y, Z)$$

INT[constr, GAV]: example



$exp_{\mathcal{G}}(q)$ is

$$\{ (X) \mid \text{person}(X, Y, Z) \vee \text{student}(X, W) \vee \text{city}(Z, X) \}$$

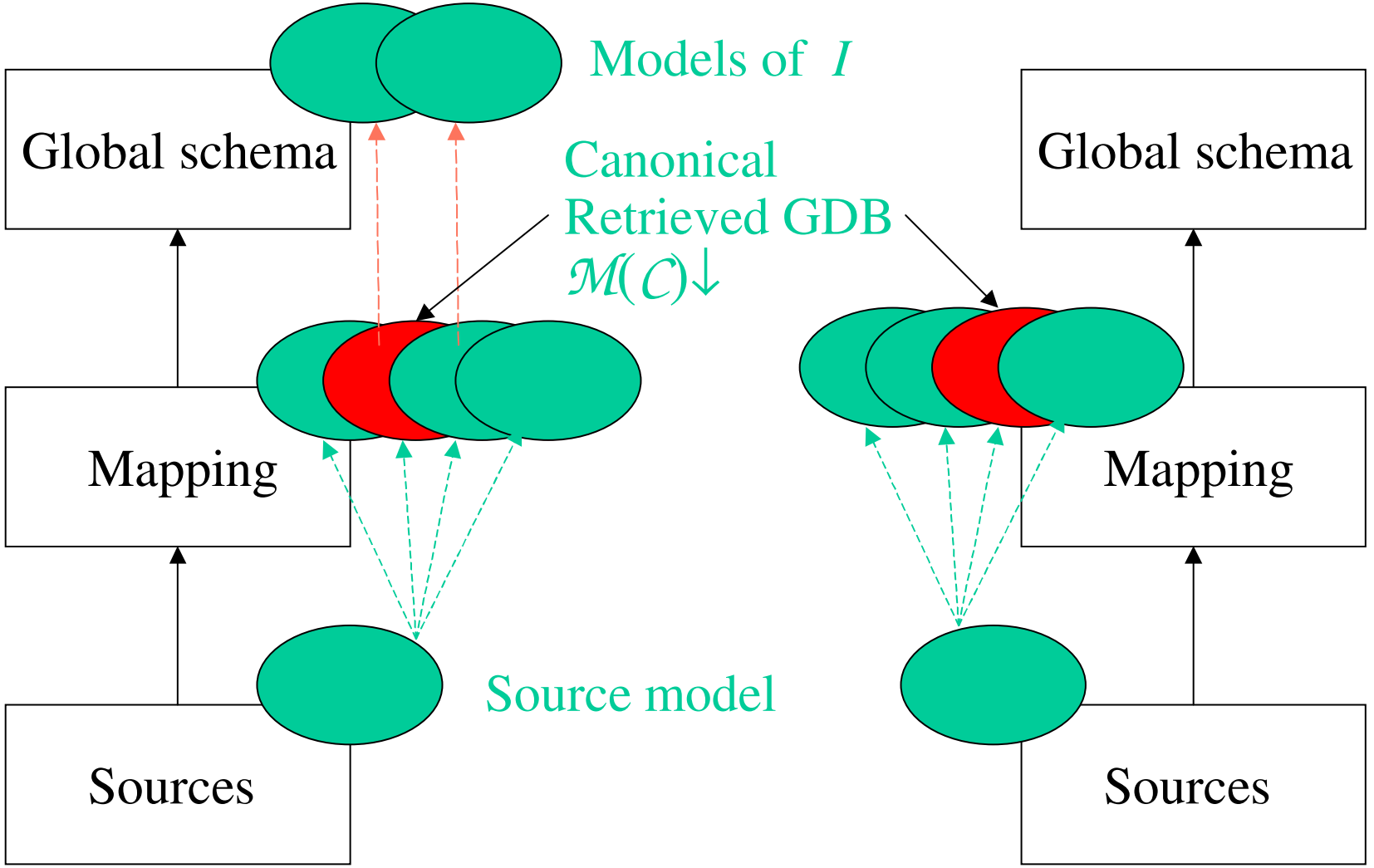
INT[constr, GAV]: algorithms and systems

- First algorithms appeared in [Calì&al. CAISE'02], [Calì&al. InfSysJ'03], and implemented in the IBIS data integration system (see [Calì&al. CAISE'03])
- Technique improved and extended in [Calì&al. PODS'03], [Calì&al. IJACI'03], in order to deal with more expressive integrity constraints, and with unions of conjunctive queries
- New technique implemented in the DisAtDis data integration system (see `http://www.dis.uniroma1.it/~disatdis`)
- The DisAtDis data integration system has been integrated with the MOMIS schema integration system (see [Bergamaschi& et al 2002]) in the context of the Sewasie European project
- More expressive queries leads to intractability/undecidability

Incompleteness and inconsistency

Constraints in $\mathcal{G}$	Type of mapping	Incompleteness	Inconsistency
no	GAV	**yes**/no	no
no	GLAV	**yes**	no
yes	GAV	**yes**	**yes**
yes	*GLAV*	**yes**	**yes**

INT[constr, GLAV]



INT[constr, GLAV]

- With functional dependencies [Duschka'97]
- With full dependencies [Duschka'97]
- With inclusion dependencies [Gryz'97] (sound but incomplete algorithm)
- With Description Logics integrity constraints [Calvanese&al. AAAI'00]
- With key and inclusion dependencies [Calì&al. '03]
- With acyclic tuple-generating dependencies [Halevy&al. ICDE'03]
- With weakly acyclic tuple-generating dependencies and equality-generating dependencies [Fagin&al. ICDT'03]

Outline

- Peer-based Distributed Information Systems
- Data integration
  - Approaches to data integration
  - Query answering in different approaches
  - Dealing with inconsistency
- Data exchange
- P2P data integration
- Conclusions

INT[constr, GAV]: Dealing with inconsistency

When for data integration system $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$ and source database $\mathcal{C}$, we have $sem^{\mathcal{C}}(\mathcal{I}) = \emptyset$, the first-order setting described above is **not adequate**.

- [Subrahmanian ACM-TODS'94]
- [Grant&al. IEEE-TKDE'95]
- [Dung CoopIS'96]
- [Lin&al. JICIS'98]
- [Yan&al. CoopIS'99]
- [Arenas&al. PODS'99]
- [Greco&al. LPAR'00]
- many approaches to KB revision and KB/DB update

Inconsistency: example

$$\begin{array}{rcl} \text{player}(Pcode, Pteam, PAge) & & \text{team}(Tcode, TCity, Tleader) \\ key(\text{player}) & = & \{Pcode\} \\ key(\text{team}) & = & \{Tcode\} \\ \text{player}[Pteam] & \subseteq & \text{team}[Tcode] \\ \text{team}[Tleader] & \subseteq & \text{player}[Pcode] \end{array}$$

$$\begin{array}{rcl} \text{player(X,Y,Z)} & \leadsto & \{\ (X,Y,Z) \mid \mathsf{s}_1(X,Y,Z,W)\ \} \\ \text{team(X,Y,Z)} & \leadsto & \{\ (X,Y,Z) \mid \mathsf{s}_2(X,Y,Z) \vee \mathsf{s}_3(X,Y,Z)\ \} \end{array}$$

Inconsistency: example

Source database $\mathcal{C}$:

$s_1^{\mathcal{C}}$:

9	*Batistuta*	*IN*	31
10	*Rivaldo*	*MI*	29

$s_2^{\mathcal{C}}$:

IN	*Inter*	8
MI	*Milan*	10

$s_3^{\mathcal{C}}$:

IN	*Inter*	9

player(X,Y,Z) $\leadsto \{ (X,Y,Z) \mid s_1(X,Y,Z,W) \}$

team(X,Y,Z) $\leadsto \{ (X,Y,Z) \mid s_2(X,Y,Z) \vee s_3(X,Y,Z) \}$

Retrieved global database:

Player:

9	*Batistuta*	*IN*
10	*Rivaldo*	*MI*

Team:

IN	*Inter*	8
MI	*Milan*	10
IN	*Inter*	9

Beyond first-order logic: loosely sound semantics

Given

- $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$, with a GAV mapping $\mathcal{M} = \{r_1 \rightsquigarrow V_1, \ \ldots \ , r_n \rightsquigarrow V_n\}$, where each V_i is a union of conjunctive queries
- source database $\mathcal{C}$ for $\mathcal{S}$,

we would like to focus on those databases for $\mathcal{I}$ that

1. satisfy $\mathcal{G}$ (constraints in $\mathcal{G}$ are rigid), and
2. **approximate as much as possible** the satisfaction of the mapping $\mathcal{M}$ wrt $\mathcal{C}$ (assertions in $\mathcal{M}$ are soft).

Beyond first-order logic: loosely sound semantics

We define an ordering between the global databases for $\mathcal{I}$ as follows. If $\mathcal{B}_1$ and $\mathcal{B}_2$ are two databases that satisfy $\mathcal{G}$, we say that $\mathcal{B}_1$ is better than $\mathcal{B}_2$ wrt $\mathcal{I}$ and $\mathcal{C}$, denoted as $\mathcal{B}_1 \gg_{\mathcal{C}}^{\mathcal{I}} \mathcal{B}_2$, if there exists an assertion $r_i \rightsquigarrow V_i$ in $\mathcal{M}$ such that

- $({r_i}^{\mathcal{B}_1} \cap {V_i}^{\mathcal{C}}) \supset ({r_i}^{\mathcal{B}_2} \cap {V_i}^{\mathcal{C}})$, and
- $({r_j}^{\mathcal{B}_1} \cap {V_j}^{\mathcal{C}}) \supseteq ({r_j}^{\mathcal{B}_2} \cap {V_j}^{\mathcal{C}})$ for all $r_j \rightsquigarrow V_j$ in $\mathcal{M}$ with $j \neq i$.

Intuitively, $\mathcal{B}_1$ has fewer deletions than $\mathcal{B}_2$ wrt the retrieved global database $\mathcal{M}(\mathcal{C})$ (see [Fagin&al. PODS'83]), and since the mapping is sound, this means that $\mathcal{B}_1$ is closer than $\mathcal{B}_2$ to the retrieved global database. In other words, $\mathcal{B}_1$ approximates the sound mapping better than $\mathcal{B}_2$.

Example

Consider $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$, with

- $\mathcal{G}$ containing relation $r(x, y)$ with key x
- $\mathcal{S}$ containing relations $s_1(x, y)$ and $s_2(x, y)$
- $\mathcal{M} = \{\ r \rightsquigarrow \{\ (x, y) \mid s_1(x, y) \vee s_2(x, y)\ \}\ \}$

and consider the source database $\mathcal{C} = \{\ s_1(a, d),\ s_1(b, d),\ s_2(a, e)\ \}$, so that the minimal retrieved global database is $\{\ r(a, d),\ r(b, d)\ ,\ r(a, e)\ \}$

We have that

- $\{\ r(a, d),\ r(b, d)\ \} \gg_{\mathcal{C}}^{\mathcal{I}} \{\ r(a, d)\ \}, \quad \{\ r(a, e),\ r(b, d)\ \} \gg_{\mathcal{C}}^{\mathcal{I}} \{\ r(a, e)\ \}$
- $\{\ r(a, d),\ r(b, d)\ \}$ and $\{\ r(a, e)\ \}$ are incomparable
- $\{\ r(a, e),\ r(b, d),\ r(c, e)\ \}$ and $\{\ r(a, e),\ r(b, d)\ \}$ are incomparable

Beyond first-order logic: loosely sound semantics

$\gg_{\mathcal{C}}^{\mathcal{I}}$ is a partial order.

A database $\mathcal{B}$ that satisfy $\mathcal{G}$ satisfies the mapping $\mathcal{M}$ with respect to $\mathcal{C}$ if $\mathcal{B}$ is maximal wrt $\gg_{\mathcal{C}}^{\mathcal{I}}$, i.e., for no other global database $\mathcal{B}'$ that satisfies $\mathcal{G}$, we have that $\mathcal{B}' \gg_{\mathcal{C}}^{\mathcal{I}} \mathcal{B}$:

$$sem^{\mathcal{C}}(\mathcal{I}) = \{\ \mathcal{B} \quad | \quad \mathcal{B} \text{ is a global database that satisfies } \mathcal{G}, \text{ and such that } \neg\exists \mathcal{B}' \text{ such that } \mathcal{B}' \text{ satisfies } \mathcal{G} \text{ and } \mathcal{B}' \gg_{\mathcal{C}}^{\mathcal{I}} \mathcal{B} \quad \}$$

The notion of model for $\mathcal{I}$ with respect to $\mathcal{C}$, and the notion of certain answer remain the same, given the new definition of satisfaction of mapping.

Loosely sound semantics: the case of INT[constr, GAV]

We assume that only key and foreign key constraints are in $\mathcal{G}$. Given $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$, and source database $\mathcal{C}$, we define the DATALOG$^\neg$ program $\mathcal{P}(\mathcal{I}, \mathcal{C})$ obtained by adding to the set of facts $\mathcal{C}$ the following set of rules:

- for each $\mathsf{g} \rightsquigarrow \{(\vec{\mathbf{x}}) \mid body_1(\vec{\mathbf{x}}, \vec{\mathbf{y}}_1) \vee \cdots \vee body_m(\vec{\mathbf{x}}, \vec{\mathbf{y}}_m)\}$ in $\mathcal{M}$, the rules:

$$\mathsf{g}_{\mathcal{C}}(\vec{\mathbf{X}}) \leftarrow body_1(\vec{\mathbf{X}}, \vec{\mathbf{Y}}_1) \quad \ldots \quad \mathsf{g}_{\mathcal{C}}(\vec{\mathbf{X}}) \leftarrow body_m(\vec{\mathbf{X}}, \vec{\mathbf{Y}}_m)$$

- for each relation $g \in \mathcal{G}$, the rules

$$\mathsf{g}(\vec{\mathbf{X}}, \vec{\mathbf{Y}}) \leftarrow \mathsf{g}_{\mathcal{C}}(\vec{\mathbf{X}}, \vec{\mathbf{Y}}) \ , \ not \ \overline{\mathsf{g}}(\vec{\mathbf{X}}, \vec{\mathbf{Y}})$$
$$\overline{\mathsf{g}}(\vec{\mathbf{X}}, \vec{\mathbf{Y}}) \leftarrow \mathsf{g}(\vec{\mathbf{X}}, \vec{\mathbf{Z}}) \ , \ \vec{\mathbf{Y}} \neq \vec{\mathbf{Z}}$$

  - in $\mathsf{g}(\vec{\mathbf{X}}, \vec{\mathbf{Y}})$, $\vec{\mathbf{X}}$ is the key of g
  - $\vec{\mathbf{Y}} \neq \vec{\mathbf{Z}}$ means that there exists i such that $Y_i \neq Z_i$.

Loosely sound semantics: the case of INT[constr, GAV]

The above rules force each stable model T of $\mathcal{P}(\mathcal{I}, \mathcal{C})$ to be such that, for each g in $\mathcal{G}$, g^T is a maximal subset of the tuples from the minimal retrieved global database that are consistent with the key constraint for g.

- $t \in cert(q, \mathcal{I}, \mathcal{C})$ under the new semantics if and only if $t \in q^T$ for each stable model T of the DATALOG$^\neg$ program $\mathcal{P}(\mathcal{I}, \mathcal{C}) \cup \{exp_{\mathcal{G}}(q)\}$
- Determining $t \in cert(q, \mathcal{I}, \mathcal{C})$ under the new semantics is coNP-complete wrt data complexity
- Technique implemented using the $\mathcal{DLV}$ system (Infomix European project)

Note: a stable model of a DATALOG$^\neg$ program Π is any set σ of ground atoms that coincides with the unique minimal Herbrand model of the DATALOG progam Π_σ, where Π_σ is obtained from Π by deleting every rule that has a negative literal $\neg B$ with $B \in \sigma$, and all negative literals in the bodies of the remaining rules

Summary of results from [Calì&al. PODS'03]

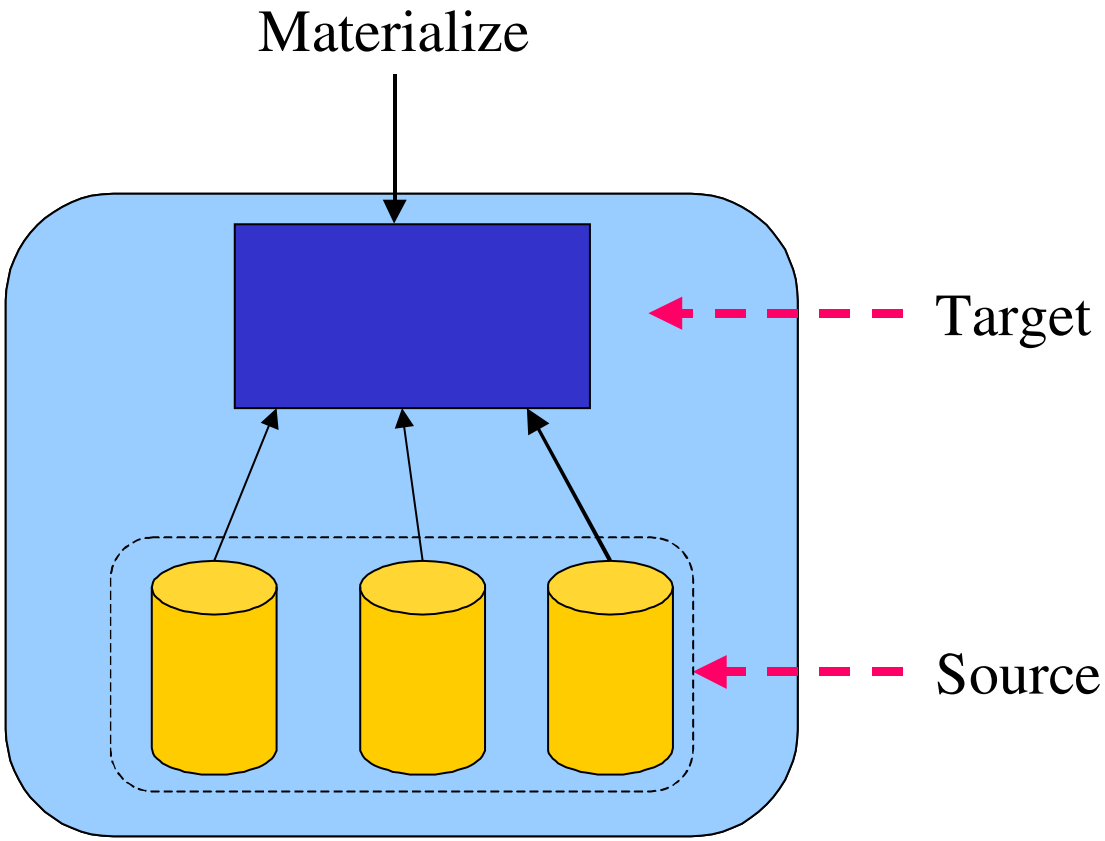
KDs	IDs	sound	loosely-sound
no	GEN	PTIME	PTIME
yes	no	PTIME	coNP
yes	FK	PTIME	coNP
yes	FK,UN	PTIME	coNP
yes	NKC	PTIME	coNP
yes	1KC	undecidable	undecidable
yes	GEN	undecidable	undecidable

Legenda: FK = foreign key dependencies, GEN = general IDs, UN = unary IDs;

Outline

- Peer-based Distributed Information Systems
- Data integration
  - Approaches to data integration
  - Query answering in different approaches
  - Dealing with inconsistency
- Data exchange
- P2P data integration
- Conclusions

Data exchange



Formal framework for data exchange

From [Fagin&al. ICDT'03], a data exchange setting $\mathcal{E} = (S, T, \Sigma_{st}, \Sigma_t)$ consists of

- a source schema S
- a target schema T
- a set Σ_{st} of source-to-target dependencies, each one of the form (tuple generating dependency, tgd)

$$\forall \vec{\mathbf{x}} \, (\phi_S(\vec{\mathbf{x}}) \rightarrow \exists \vec{\mathbf{y}} \phi_T(\vec{\mathbf{x}}, \vec{\mathbf{y}}))$$

  with $\phi_S(\vec{\mathbf{x}})$ conjunction of atoms over S, and $\phi_T(\vec{\mathbf{x}}, \vec{\mathbf{y}})$ conjunction of atoms over T (cfr. GLAV mappings in data integration)
- a set Σ_t of target dependencies, each one of the form (tgd, or equality generating dependency)

$$\forall \vec{\mathbf{x}} \, (\phi_S(\vec{\mathbf{x}}) \rightarrow \exists \vec{\mathbf{y}} \phi_T(\vec{\mathbf{x}}, \vec{\mathbf{y}})) \quad \text{or} \quad \forall \vec{\mathbf{x}} \, (\phi_T(\vec{\mathbf{x}}) \rightarrow (x_1 = x_2))$$

Formal framework for data exchange

The data exchange problem associated with the data exchange setting $\mathcal{E} = (S, T, \Sigma_{st}, \Sigma_t)$ is the following:

- given a finite instance $\mathcal{C}$ of S (source instance)
- find a finite instance J of T (target instance) such that (I, J) satisfies Σ_{st}, and J satisfies Σ_t.

Such a J is called a solution for $\mathcal{E}$ wrt $\mathcal{C}$, or simply for $\mathcal{C}$. The set of all solutions is denoted by Sol($\mathcal{C}$).

Example of data exchange

Σ_{st}:

$$\{\ \forall a \forall b \forall c\, (P(a,b,c) \rightarrow \exists Y \exists Z\ \ T(a,Y,Z))$$
$$\forall a \forall b \forall c\, (Q(a,b,c) \rightarrow \exists X \exists U\ \ T(X,b,U))$$
$$\forall a \forall b \forall c\, (R(a,b,c) \rightarrow \exists V \exists W\ \ T(V,W,c))\ \}$$

$$\mathcal{C} = \{P(a_0,b_1,c_1),\ Q(a_2,b_0,c_2),\ R(a_3,b_3,c_0)\}$$

Possible solutions:

$$J = \{T(a_0,Y_0,Z_0),\ T(X_0,b_0,U_0),\ T(V_0,W_0,c_0)\}$$

$$J_1 = \{T(a_0,b_0,c_0)\}$$

$$J_2 = \{T(a_0,b_0,Z_1),\ T(V_1,W_1,c_0)\}$$

Data exchange: results

The following results appear in [Fagin&al. ICDT'03]:

- If $\mathcal{C}$ is a source instance and J, J' are universal solutions for $\mathcal{C}$, then J and J' are homomorphically equivalent
- Let $\mathcal{C}, \mathcal{C}'$ be two source instances, J a universal solution for $\mathcal{C}$, and J' a universal solution for $\mathcal{C}'$. Then Sol($\mathcal{C}$)=Sol($\mathcal{C}'$) if and only if J and J' are homomorphically equivalent
- If the tgds in Σ_t are weakly acyclic (i.e., cycles do not involve existentially quantified variables), then the existence of a solution for $\mathcal{C}$ can be checked in polynomial time wrt the size of $\mathcal{C}$. Moreover, if a solution for $\mathcal{C}$ exists, then a universal solution for $\mathcal{C}$ can be produced in polynomial time wrt the size of $\mathcal{C}$ (by chasing $\mathcal{C}$)

Data exchange: materialization and certain answers

- Materialization

  It follows that a universal solution for $\mathcal{I}$ wrt $\mathcal{C}$ is the right structure to materialize in the target

- Certain answers

  Let $\mathcal{E} = (S, T, \Sigma_{st}, \Sigma_t)$ be a data exchange setting, let Q be a k-ary query over the target schema T, and let $\mathcal{C}$ be a source instance.

  The **certain answers of Q wrt $\mathcal{E}$ and $\mathcal{C}$**, denoted $cert(Q, \mathcal{E}, \mathcal{C})$, is the set of all the k-tuples of constants in Const(S) such that, for every solution J for $\mathcal{C}$ of the data exchange problem associated to the above setting, we have that $t \in Q^J$ (cfr. certain answers in data integration).

Data exchange: query answering

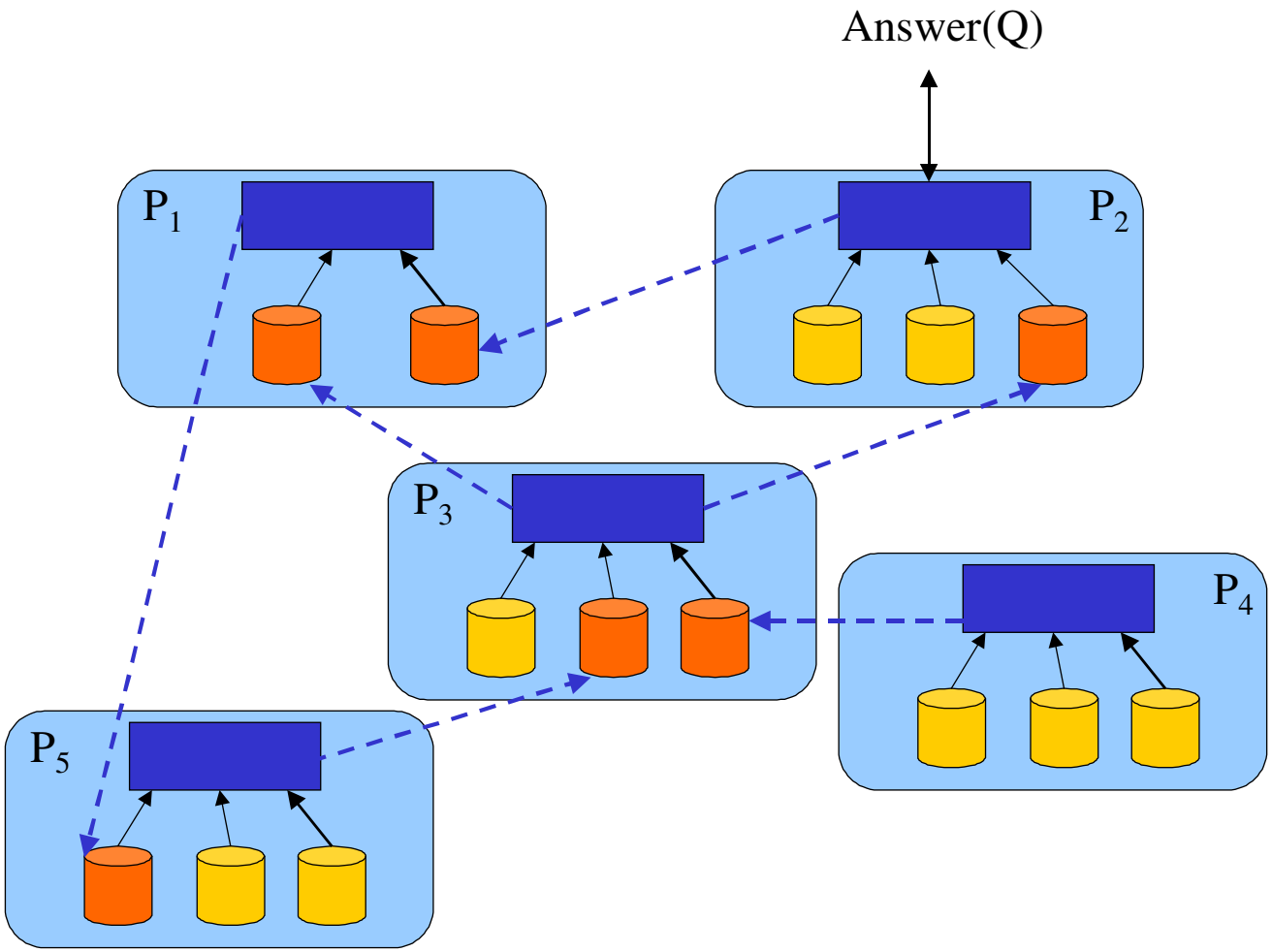
Results from [Fagin&al. ICDT'03]:

- If Q is a union of conjunctive queries, $\mathcal{C}$ is a source instance, and J is a universal solution for $\mathcal{C}$, then $cert(Q, \mathcal{E}, \mathcal{C}) = Q^J$
- If for every conjunctive query $cert(Q, \mathcal{E}, \mathcal{C}) = Q^J$, then J is a universal solution for $\mathcal{C}$
- If the tgds in Σ_t are weakly acyclic, and Q is a union of conjunctive queries, then for every source instance $\mathcal{C}$, the set $cert(Q, \mathcal{E}, \mathcal{C})$ can be computed in polynomial time wrt the size of $\mathcal{C}$

Outline

- Peer-based Distributed Information Systems
- Data integration
  - Approaches to data integration
  - Query answering in different approaches
  - Dealing with inconsistency
- Data exchange
- P2P data integration
- Conclusions

P2P data integration



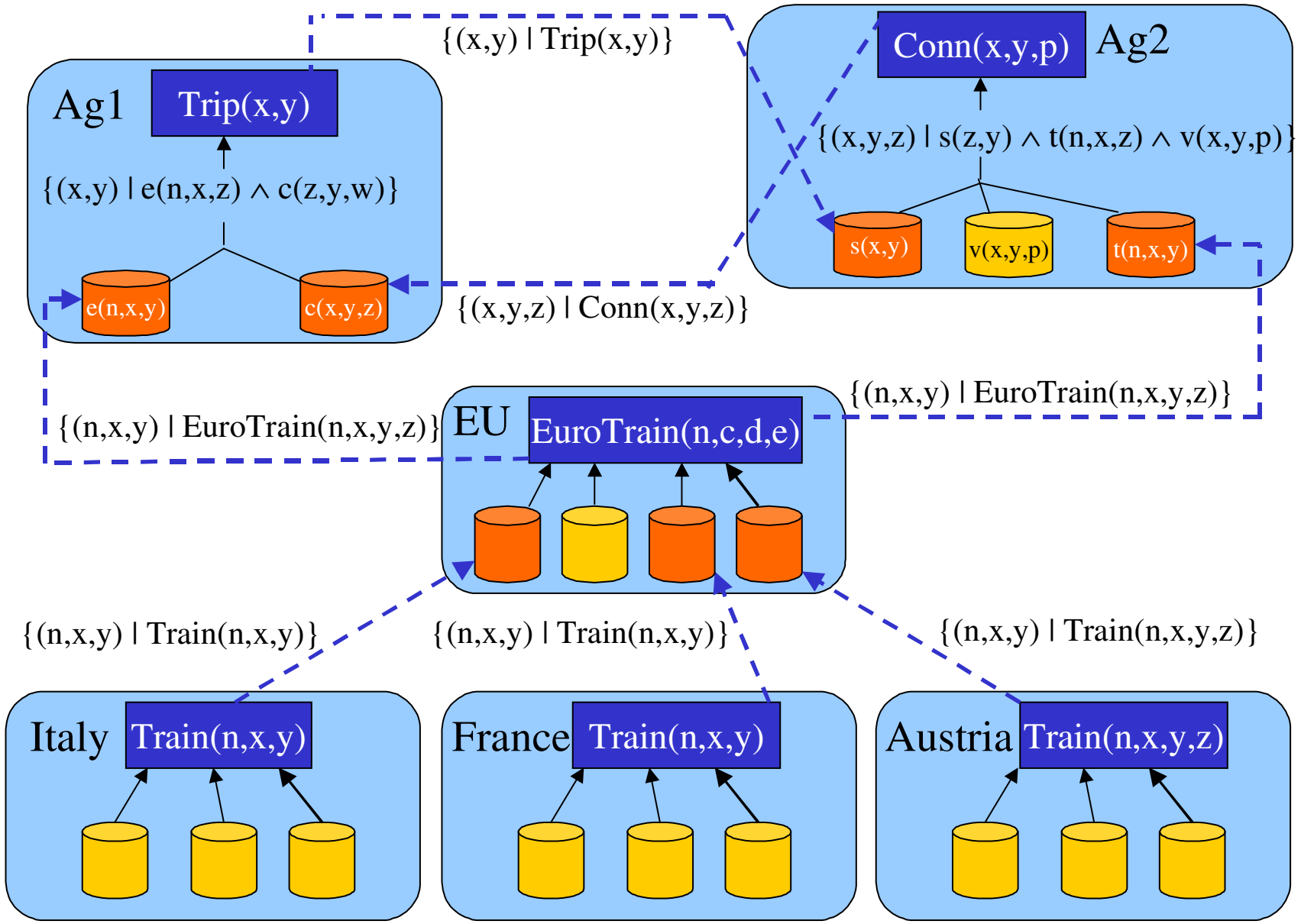
Formal framework for P2P data integration

The following framework is inspired by [Halevy&al. ICDE'03]:

- A P2P system Π is constituted by a set of peers $\{P_1, \ldots, P_n\}$
- Each peer P_i of Π is constituted by
  - a schema $\mathcal{G}_i$ (the peer schema)
  - a set $\mathcal{L}$ of local sources
  - a set $\mathcal{E}$ of external sources
  - a set of local mappings from the sources $\mathcal{S} = \mathcal{L} \cup \mathcal{E}$ to the peer schema, each of the form: $\exists \vec{\mathbf{z}} \phi_{\mathcal{S}}(\vec{\mathbf{x}}, \vec{\mathbf{z}}) \rightsquigarrow \exists \vec{\mathbf{y}} \phi_{\mathcal{G}}(\vec{\mathbf{x}}, \vec{\mathbf{y}})$
  - each P2P mapping is an assertion assigning to each external source $E \in \mathcal{E}$ of P_i a query over another peer P_j, each one of the form: $E \rightsquigarrow \exists \vec{\mathbf{y}} \phi_j(\vec{\mathbf{x}}, \vec{\mathbf{y}})$

Note that each peer can be conceived as a GLAV data integration system, or as a data exchange setting.

P2P data integration: example



Formal framework for P2P data integration

- A local source database $\mathcal{C}$ for Π is a database for the set $\mathcal{L}$ of all local source predicates in the various peers of Π
- An external source database $\mathcal{D}$ for Π is a database for the set $\mathcal{E}$ of all external source predicates in the various peers of Π
- The union of a local source database $\mathcal{C}$ for Π and an external source database $\mathcal{D}$ for Π is a database for $\mathcal{S} = \mathcal{L} \cup \mathcal{E}$, and is called a source database for Π
- A global database for Π is a database for the union $\mathcal{G}$ of all peer schemas of Π (which are assumed to be pairwise disjoint)
- A global database for Π is said to be legal wrt $\mathcal{G}$ if it satisfies all peer schemas

Semantics of P2P data integration

Given a local source database $\mathcal{C}$ for Π, the set of models of Π relative to $\mathcal{C}$ is:

$sem^{\mathcal{C}}(\mathbf{\Pi}) = \{\ \mathcal{B}$ | $\mathcal{B}$ is a global database for Π that is legal wrt $\mathcal{G}$, and
$\exists$ an external source database $\mathcal{D}$ for Π such that
– $\mathcal{B}$ satisfies all local mapping assertions wrt $\mathcal{C} \cup \mathcal{D}$
– $\mathcal{B}$ satisfies all P2P mapping assertions wrt $\mathcal{D}$ $\}$

- $\mathcal{B}$ satisfies a local mapping assertion $\exists \vec{\mathbf{z}} \phi_{\mathcal{S}}(\vec{\mathbf{x}}, \vec{\mathbf{z}}) \rightsquigarrow \exists \vec{\mathbf{y}} \phi_i(\vec{\mathbf{x}}, \vec{\mathbf{y}})$ wrt $\mathcal{C} \cup \mathcal{D}$ if $(\exists \vec{\mathbf{z}} \phi_S(\vec{\mathbf{x}}, \vec{\mathbf{z}}))^{\mathcal{C} \cup \mathcal{D}} \subseteq \exists (\vec{\mathbf{y}} \phi_i(\vec{\mathbf{x}}, \vec{\mathbf{y}}))^{\mathcal{B}}$
- the meaning of $\mathcal{B}$ satisfying a P2P mapping assertion wrt $\mathcal{D}$ may vary in the various approaches

The set of certain answers to a query Q posed to a peer P of Π wrt the source database $\mathcal{C}$ is the set $cert(Q^P, \Pi, \mathcal{C})$ of tuples that satisfy Q in all the models of Π relative to $\mathcal{C}$.

First order logic semantics of P2P data integration

According to most approaches (see [Halevy&al. ICDE'03], [Bernstein&al. WebDB '02]), the semantics of P2P mapping assertions in P2P system Π is given in terms of first order logic (FOL):

Satisfaction of a P2P mapping assertion

$$E_j \quad \rightsquigarrow \quad \exists \vec{\mathbf{y}} \phi_i(\vec{\mathbf{x}}, \vec{\mathbf{y}})$$

of Π by a global database $\mathcal{B}$ wrt $\mathcal{D}$ means

- satisfaction of the FOL formula

$$\forall \vec{\mathbf{x}}(\exists \vec{\mathbf{y}} \phi_i(\vec{\mathbf{x}}, \vec{\mathbf{y}}) \rightarrow E_j(\vec{\mathbf{x}}))$$

- which is equivalent to the condition

$$(\exists \vec{\mathbf{y}} \phi_i(\vec{\mathbf{x}}, \vec{\mathbf{y}}))^{\mathcal{B}} \subseteq E_j^{\mathcal{D}}$$

First order logic semantics of P2P data integration

In [Calvanese&al. 2003], it is argued that the FOL semantics is not adequate for P2P data integration, mainly because

- The system is modeled by a flat FOL theory, with no formal separation between the various peers
- The modular structure of the system is not reflected in the semantics
- Bad computational properties: computing the set of certain answers to a conjunctive query Q posed to a peer is undecidable, even for simple P2P systems, i.e., for P2P systems where all peer schemas are empty (see [Halevy&al. ICDE'03], [Koch FOIKS'02])
- In order to recover decidability, one has to limit the expressive power of P2P mappings (e.g., acyclicity is assumed in [Halevy&al. ICDE'03])

Epistemic semantics for P2P data integration

In [Calvanese&al. 2003], a new semantics is proposed for P2P data integration, based on epistemic logic

- A P2P mapping $E_j \rightsquigarrow \exists\vec{\mathbf{y}}\phi_i(\vec{\mathbf{x}},\vec{\mathbf{y}})$ is interpreted as the formula

$$\forall\vec{\mathbf{x}}((\mathbf{K}\,\exists\vec{\mathbf{y}}\phi_i(\vec{\mathbf{x}},\vec{\mathbf{y}}))\rightarrow E_j(\vec{\mathbf{x}}))$$

  which imposes that only the certain answers to $\exists\vec{\mathbf{y}}\phi_i(\vec{\mathbf{x}},\vec{\mathbf{y}})$ at peer i are transferred to peer j (peer i communicates to peer j only facts that are certain, i.e., true in every model of the P2P system)
- The modular structure of the system is now reflected in the semantics (by virtue of the modal semantics of epistemic logics)
- Good computational properties: for simple P2P systems, computing the set of certain answers to a conjunctive query Q wrt a local source database $\mathcal{C}$ is not only decidable, but also polynomial time in the size of $\mathcal{C}$, even for cyclic mappings

Outline

- Peer-based Distributed Information Systems
- Data integration
  - Approaches to data integration
  - Query answering in different approaches
  - Dealing with inconsistency
- Data exchange
- P2P data integration
- Conclusions

Conclusions

Many open problems and issues, including

- More on P2P data integration
- Several interesting classes of integrity constraints in peer schemas
- Global schema (or target schema, or peer schemas) expressed in terms of semi-structured data (with constraints)
- Dealing with inconsistencies vs. data cleaning
- Going beyond the “unique domain assumption”
- Limitations in accessing the sources
- Exact sources as opposed to sound sources
- How to incorporate the notion of data quality (source reliability, accuracy, etc.)
- Optimization (reasoning on queries and views)

Acknowledgements

Special thanks to

- **Andrea Calí**
- **Diego Calvanese**
- **Giuseppe De Giacomo**
- **Domenico Lembo**
- **Riccardo Rosati**
- **Moshe Y. Vardi**