Changing the Instance Level of Ontologies: A Logic Approach

Maurizio Lenzerini

DASI-lab – Data and Service Integration Laboratory
Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
SAPIENZA Università di Roma

Joint work with Giuseppe De Giacomo, Antonella Poggi, Riccardo Rosati

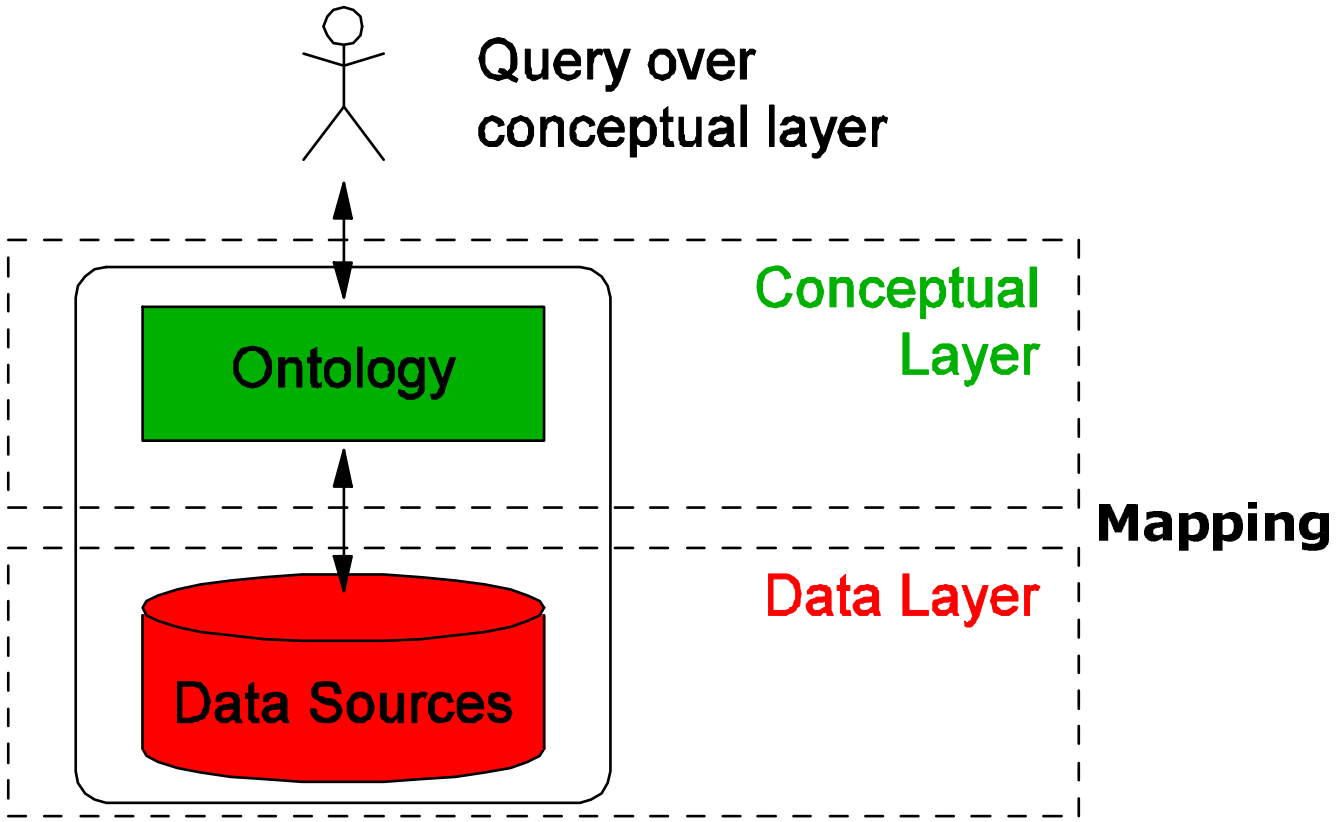
Invited talk at
Logic in Databases – LID 2008
Roma, Italy, May 19, 2008

Outline

- The general context: “write also” information integration
  - Ontology-based data integration
  - The Description Logic *DL-Lite*$_{\mathcal{F}}$
- Instance-level update of ontology
  - Semantics
  - The expressibility problem
  - The notion of approximation
- Approximated instance-based update in *DL-Lite*$_{\mathcal{F}}$
- Approximated instance-based erasure in *DL-Lite*$_{\mathcal{F}}$
- Discussion and conclusions

Ontology-based data integration

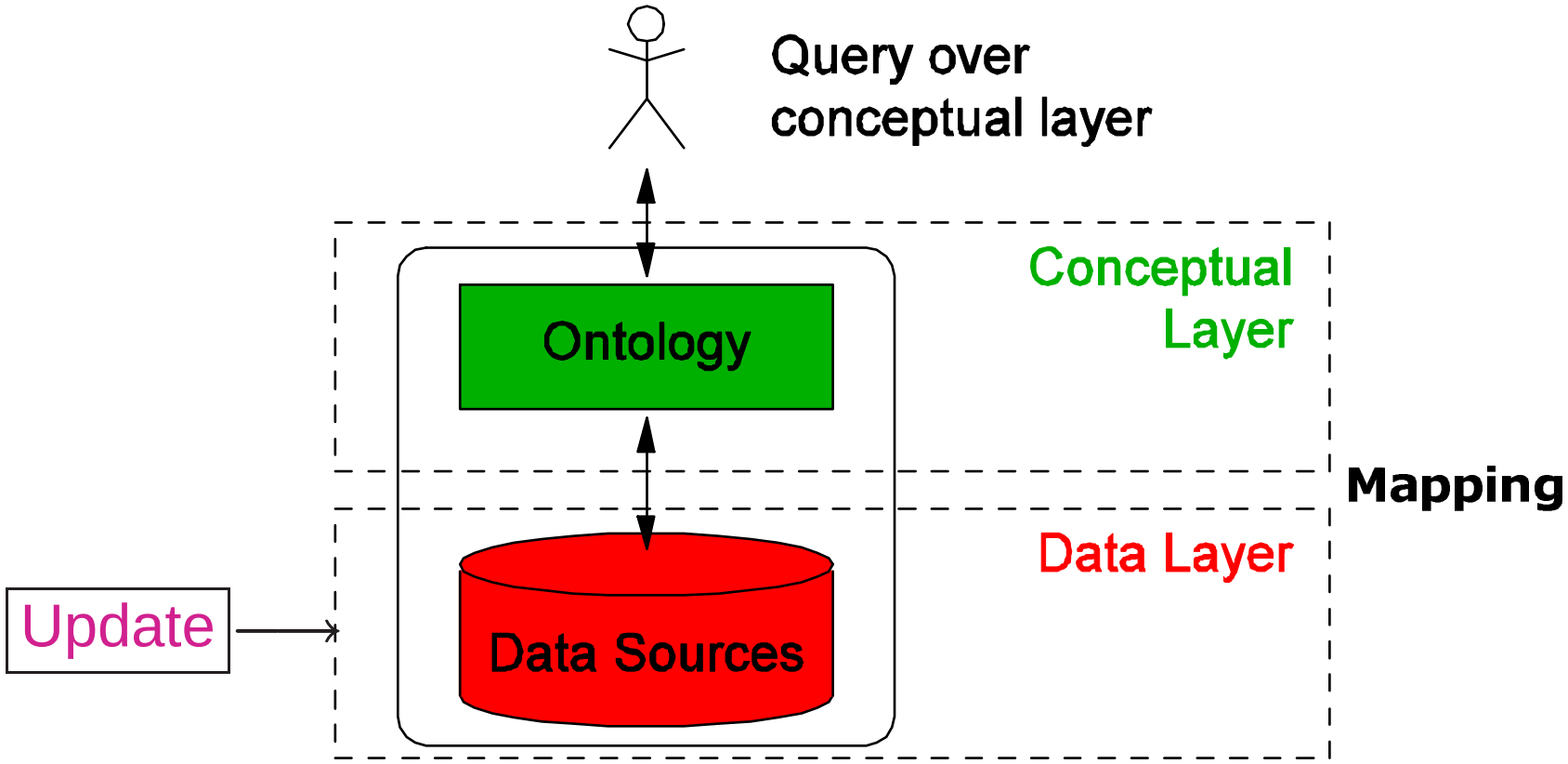
Ontology as a conceptual view over data sources



Fundamental task: query answering

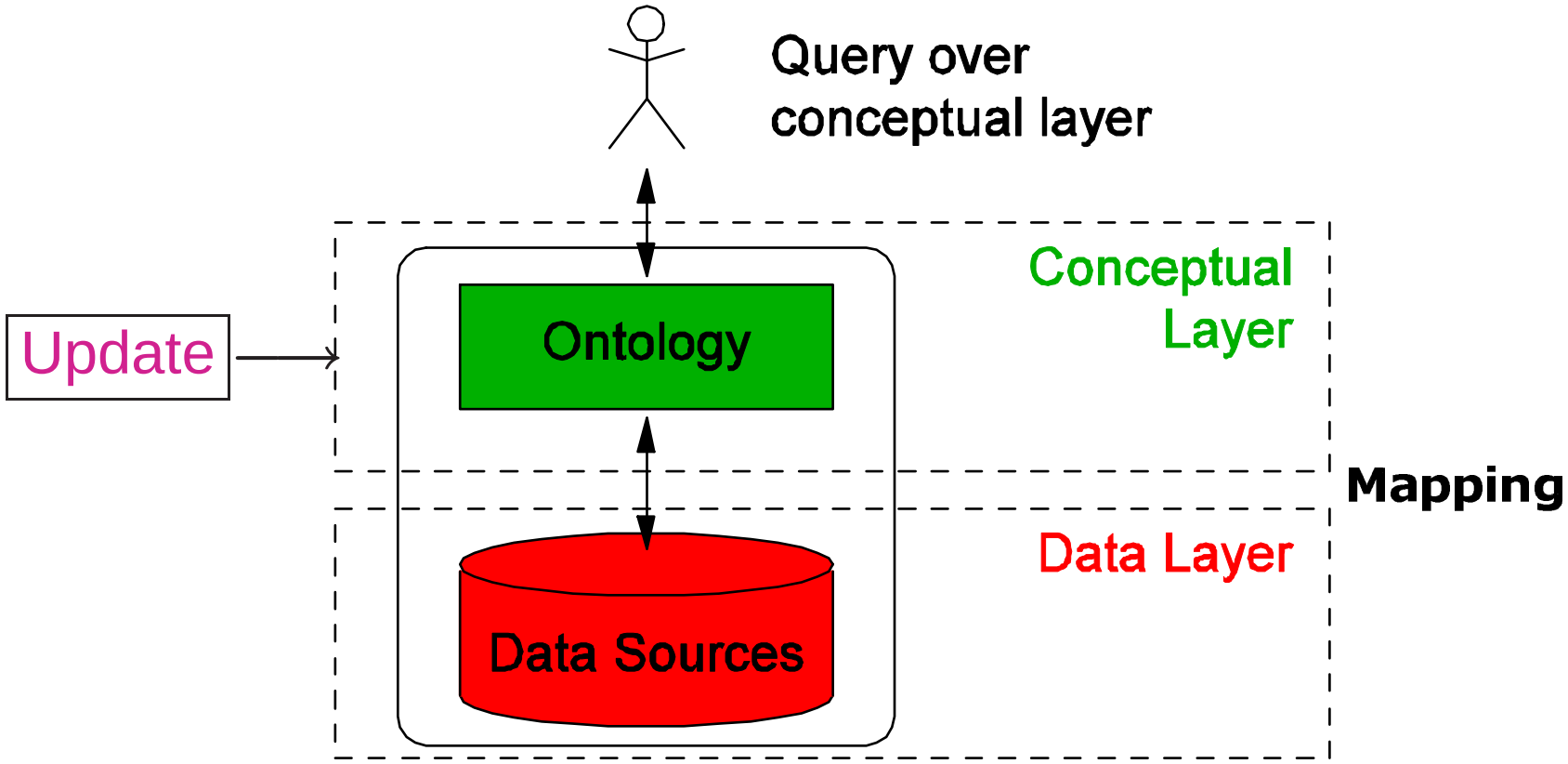
Ontology-based data integration

Current treatment of update.



“Write-also” ontology-based data integration

Our long-term goal.



Ontology-based data integration: the MASTRO system

- Ontology language
  - The Description Logic *DL-Lite*$_{\mathcal{F}}$
- Mapping language
  - Global-As-View (GAV) mappings, with suitable mechanisms for addressing the “impedance mismatch” problem
- Query language
  - Unions of conjunctive queries
- Semantics
  - Based on first-order logic
- Essentially optimal solution (wrt complexity of query answering)

Description Logic Ontologies

We start with an alphabet of unary predicate symbols (concepts), binary predicate symbols (roles), and constants (objects).

A DL ontology $\mathcal{K}$ is characterized by a pair $\langle \mathcal{T}, \mathcal{A} \rangle$ such that:

- the TBox $\mathcal{T}$ represents the intensional level of the ontology, i.e. it consists of a set of universal assertions (called inclusion assertions) on concepts and roles
- the ABox $\mathcal{A}$ represents the extensional level of the ontology, i.e. it consists of a set of membership assertions, stating that a given object (pair of objects) is an instance of a concept (role)

A conjunctive query q is a conjunction of atoms over the symbols of the alphabet

$$q = \{\vec{x} \mid \exists \vec{y}.conj(\vec{x}, \vec{y})\}$$

Example: $\{x \mid \exists y.Manager(x) \wedge Member(x, y) \wedge \exists Director(x)\}$

Description Logic Ontologies – Semantics

Pure first-order logic semantics for $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$.

An interpretation is a pair $\mathcal{I} = (\Delta^I, \cdot^I)$, where

- $\Delta^{\mathcal{I}}$ is a non-empty set, called the domain of $\mathcal{I}$
- $\cdot^{\mathcal{I}}$ is the interpretation function of $\mathcal{I}$, i.e., a function that assigns a different element of $\Delta^{\mathcal{I}}$ to each constant (unique name assumption), a subset of $\Delta^{\mathcal{I}}$ to each concept, and a subset of $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$ to each role

An interpretation is a model of $\mathcal{K}$ if it satisfies all inclusion assertions in $\mathcal{T}$ and all membership assertions in $\mathcal{A}$.

$\rightsquigarrow$ Open World Assumption: no negative fact is assumed by default.

Reasoning services

- Knowledge base satisfiability: check whether $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ has a model.

- Query answering amounts to computing certain answers to q wrt $\mathcal{K}$:

$$cert(q, \mathcal{K}) = \{\vec{c} \mid \mathcal{K} \models q(\vec{c})\}$$

  i.e., the tuples of constants that are answers to the query q in every model of $\mathcal{K}$.

  Note the difference wrt query evaluation in traditional databases!

Complexity of reasoning services depends on the expressive power of the language used to express $\mathcal{K}$.

The *DL-Lite* family

- Is a family of DLs optimized according to the tradeoff between expressive power and data complexity
- Two maximal languages that enjoy nice computational properties: *DL-Lite*$_{\mathcal{F}}$, *DL-Lite*$_{\mathcal{R}}$
- One language *DL-Lite*$_{\mathcal{A}}$ that carefully combines the two languages
- With minimal additions to *DL-Lite*$_{\mathcal{F}}$ or *DL-Lite*$_{\mathcal{R}}$, such nice properties are lost

DL-Lite$_{\mathcal{F}}$

Expressions

$$\begin{array}{rcl} B & ::= & A \mid \exists R \\ C & ::= & B \mid \neg B \\ R & ::= & P \mid P^- \end{array}$$

TBox assertions

$$\begin{array}{ll} B \sqsubseteq C & \textbf{inclusion assertion} \\ (funct\ R) & \textbf{functionality assertion} \end{array}$$

ABox assertions (C is a general concept, and z is a variable)

$$C(a), \quad R(a, b) \quad \textbf{membership assertion}$$

Semantics of *DL-Lite*$_{\mathcal{F}}$

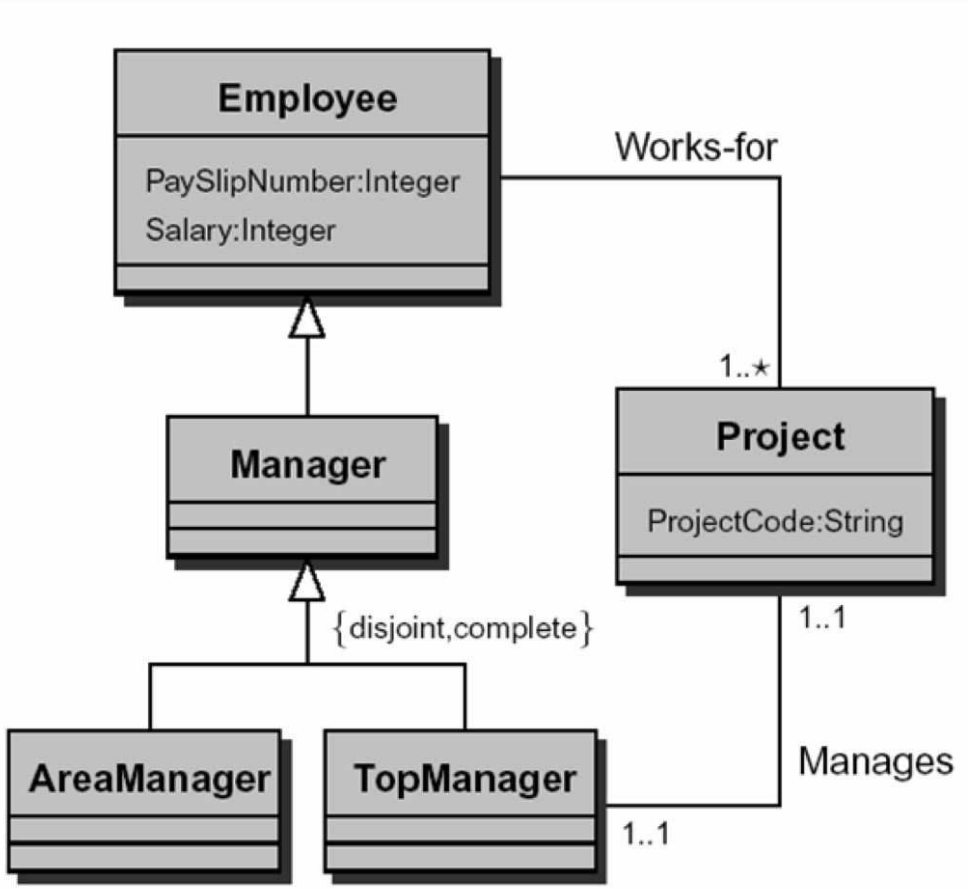
Construct	Syntax	Example	Semantics
atom. conc.	A	$Doctor$	$A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$
atom. role	P	$child$	$P^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$
exist. res.	$\exists P$	$\exists child$	$\{ d \mid \exists e.(d,e) \in P^{\mathcal{I}} \}$
exist. res.	$\exists P^-$	$\exists child^-$	$\{ e \mid \exists d.(d,e) \in P^{\mathcal{I}} \}$
negation	$\neg B$	$\neg Doctor$	$\Delta^{\mathcal{I}} \setminus Cl^{\mathcal{I}}$
incl. asser.	$B \sqsubseteq C$	$Father \sqsubseteq \exists child$	$Cl^{\mathcal{I}} \subseteq Cr^{\mathcal{I}}$
funct. asser.	(**funct** P)	(**funct** $succ$)	$\forall d,e,e'.(d,e),(d,e') \in P^{\mathcal{I}} \rightarrow e = e'$
funct. asser.	(**funct** P^-)	(**funct** $child^-$)	$\forall e,e',d.(e,d),(e',d) \in P^{\mathcal{I}} \rightarrow e = e'$
mem. asser.	$C(a)$	$Father(bob)$	$a^{\mathcal{I}} \in A^{\mathcal{I}}$
mem. asser.	$P(a,b)$	$child(bob,ann)$	$(a^{\mathcal{I}}, b^{\mathcal{I}}) \in P^{\mathcal{I}}$

- inclusion assertions $\longrightarrow$ inclusion dependencies or disjointness constraints
- functionality assertions $\longrightarrow$ functional dependencies
- membership assertions $\longrightarrow$ tuples on an incomplete database

Capturing basic ontology constructs in *DL-Lite*$_{\mathcal{F}}$

ISA between classes	$A_1 \sqsubseteq A_2$
disjointness between classes	$A_1 \sqsubseteq \neg A_2$
domain and range of relations	$\exists P \sqsubseteq A_1 \qquad \exists P^- \sqsubseteq A_2$
mandatory participation	$A_1 \sqsubseteq \exists P \qquad A_2 \sqsubseteq \exists P^-$
functionality of relations (in *DL-Lite*$_{\mathcal{F}}$)	$(funct\ P) \qquad (funct\ P^-)$
ISA between relations (in *DL-Lite*$_{\mathcal{R}}$)	$R_1 \sqsubseteq R_2$

DL-Lite – Example



$$
\begin{aligned}
Manager &\sqsubseteq Employee \\
AreaManager &\sqsubseteq Manager \\
TopManager &\sqsubseteq Manager \\
AreaManager &\sqsubseteq \neg TopManager \\
\exists WorksFor &\sqsubseteq Employee \\
\exists WorksFor^{-} &\sqsubseteq Project \\
Project &\sqsubseteq \exists WorksFor^{-} \\
(funct\ WorksFor) & \\
(funct\ WorksFor^{-}) & \\
\vdots &
\end{aligned}
$$

The sources and the mapping

- The data sources are wrapped into a relational database $\mathcal{DB}$ (constituted by the relational schema, and the extensions of the relations), so that we can query such data by using SQL.
- The database $\mathcal{DB}$ is independent from the ontology; in other words, our aim is to link to the ontology a collection of data that exist autonomously, and have not been necessarily structured with the purpose of storing the ontology instances.

Ontology with mappings to data sources

An ontology with mappings is a triple $\langle \mathcal{T}, \mathcal{M}, DB \rangle$ such that:

- $\mathcal{T}$ is a TBox;
- DB is a relational database;
- $\mathcal{M}$ is a set of mapping assertions, each one of the form

$$\Phi \rightsquigarrow \Psi$$

  where
  - Φ is an arbitrary SQL query over DB,
  - Ψ is a conjunctive query over $\mathcal{T}$ without non-distinguished variables, that possibly involves terms.

The set $Mod(\langle \mathcal{T}, \mathcal{M}, DB \rangle)$ of models of $\langle \mathcal{T}, \mathcal{M}, DB \rangle$ is the set of interpretations $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ such that:

- $\mathcal{I}$ is a model of $\mathcal{T}$;
- $\mathcal{I}$ satisfies $\mathcal{M}$ wrt DB, i.e., satisfies every assertion in $\mathcal{M}$ wrt DB.

Query answering: sketch of the algorithm

Given CQ q and $\mathcal{K} = \langle \mathcal{T}, \mathcal{M}, DB \rangle$ (assumed to be satisfiable), we compute $cert(q, \mathcal{K})$ as follows:

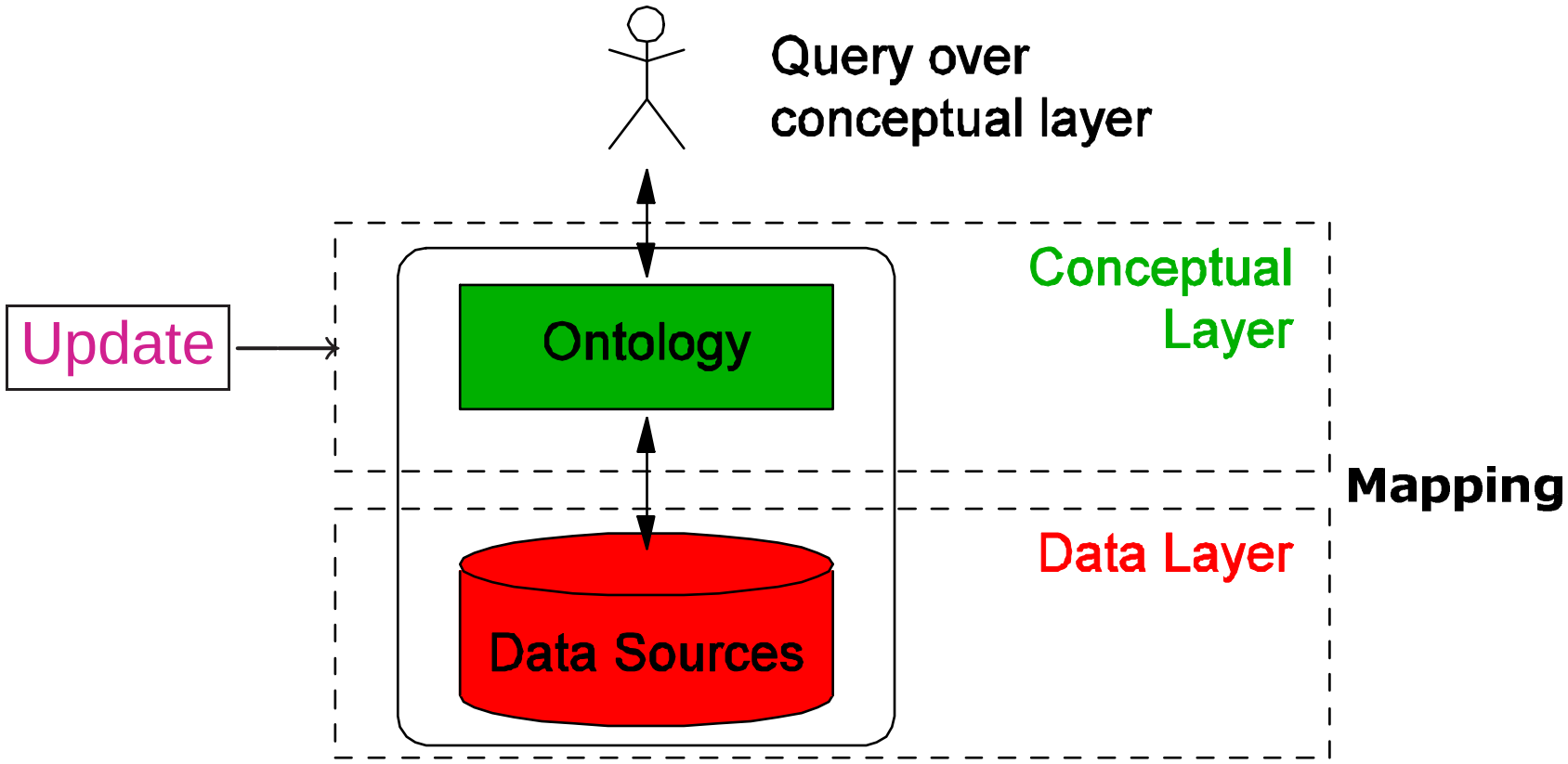
1. Using $\mathcal{T}$, reformulate CQ q as a union $r_{q,\mathcal{T}}$ of CQs
2. Using $\mathcal{M}$, unfold $r_{q,\mathcal{T}}$ to obtain a union $unfold(r_{q,\mathcal{T}})$ of CQs
3. Evaluate $unfold(r_{q,\mathcal{T}})$ directly over DB using RDBMS technology

Correctness of this algorithm shows FOL-reducibility of query answering,

$\rightsquigarrow$ Query answering is in P wrt to the size of $\mathcal{T}$ and $\mathcal{M}$, is in LOGSPACE wrt the size of DB, and can be done using RDBMS technology.

“Write-also” ontology-based data integration

Our long-term goal.



First step: instance-level update/erasure of the ontology (with TBox and Abox)

Second step: how to push update to the data sources

Instance-level update of DL ontologies: challenges

Basic assumption: the TBox describes invariant characteristics of the domain of interest, and therefore the universal assertions in the TBox are considered immutable (other papers address the issue of updating the intensional level also, e.g. [Haase & Stojanovic 2005]). It follows that an update is specified through a set of ABox assertions (new facts holding in the real world).

Challenges:

- What is the "right" semantics for the update?
- What to do when the asserted membership is inconsistent with $\mathcal{T}$?
- What to do when the asserted membership is consistent with $\mathcal{T}$, but is inconsistent with $\mathcal{K}$?

Only few papers on this issue (see [Liu & al 2006])

Update and revision

Generally speaking, we are dealing with the problem of modifying an ontology.

- Revision (and contraction): we have obtained new information about a world (that did not change)
- Update (and erasure): we bring the ontology up to date when the world described by it changes
  - syntactic approach
  - semantic approach

Instance level update: our approach

We follow the classical approaches to update proposed, for example, by [Winslett ’88 - ’90] and [Katsuno&Mendelzon ’91]

Such classical approaches need to be adapted in order to take into account a clear distinction between:

- TBox $\mathcal{T}$: invariant
- ABox $\mathcal{A}$: subject to changes

The key idea of our semantic approach:

The ontology represents a set of models (worlds). When the ontology is updated with a set of facts $\mathcal{F}$, we have to “update” each model. Updating a model m means to compute the models of $\mathcal{T}$ and $\mathcal{F}$ that are “close” to m.

Containment between interpretations

Let $\mathcal{I} = \langle \Delta, \cdot^{\mathcal{I}} \rangle$ and $\mathcal{I}' = \langle \Delta, \cdot^{\mathcal{I}'} \rangle$ be two interpretations (over the same alphabet).

We say that $\mathcal{I}$ is contained in $\mathcal{I}'$, written $\mathcal{I} \subseteq \mathcal{I}'$, iff $\mathcal{I}, \mathcal{I}'$ are such that:

- for every $c \in \Delta$ and atomic concept A: $c \in A^{\mathcal{I}}$ implies $c \in A^{\mathcal{I}'}$;
- for every $(c, d) \in \Delta \times \Delta$ and atomic role R: $(c, d) \in R^{\mathcal{I}}$ implies $(c, d) \in R^{\mathcal{I}'}$.

We say that $\mathcal{I}$ is properly contained $\mathcal{I}'$, written $\mathcal{I} \subset \mathcal{I}'$, iff $\mathcal{I} \subseteq \mathcal{I}'$ and $\mathcal{I}' \not\subseteq \mathcal{I}$.

Difference between interpretations

Let $\mathcal{I} = \langle \Delta, \cdot^{\mathcal{I}} \rangle$ and $\mathcal{I}' = \langle \Delta, \cdot^{\mathcal{I}'} \rangle$ be two interpretations (over the same alphabet).

We define the difference between $\mathcal{I}$ and $\mathcal{I}'$, written $\mathcal{I} \ominus \mathcal{I}'$, as the interpretation $(\Delta, \cdot^{\mathcal{I} \ominus \mathcal{I}'})$ such that:

- $A^{\mathcal{I} \ominus \mathcal{I}'} = A^{\mathcal{I}} \ominus A^{\mathcal{I}'}$, for every atomic concept A;
- $R^{\mathcal{I} \ominus \mathcal{I}'} = R^{\mathcal{I}} \ominus R^{\mathcal{I}'}$, for every atomic role R,

where $S \ominus S'$ denotes the symmetric difference between sets S and S', i.e. $S \ominus S' = (S \cup S') \setminus (S \cap S')$.

Update of a model

Let $\mathcal{T}$ be a TBox in a DL $\mathcal{L}$, $\mathcal{I}$ a model of $\mathcal{T}$, and $\mathcal{F}$ a finite set of membership assertions expressed in $\mathcal{L}$ such that $Mod(\mathcal{T}) \cap Mod(\mathcal{F}) \neq \emptyset$.

The update of $\mathcal{I}$ with $\mathcal{F}$ is defined as follows:

$$U^{\mathcal{T}}(\mathcal{I}, \mathcal{F}) = \{\, \mathcal{I}' \mid \mathcal{I}' \in Mod(\mathcal{T}) \cap Mod(\mathcal{F}), \text{ and there exists no } \mathcal{I}'' \in Mod(\mathcal{T}) \cap Mod(\mathcal{F}) \text{ s.t. } \mathcal{I} \ominus \mathcal{I}'' \subset \mathcal{I} \ominus \mathcal{I}' \,\}$$

Definition of update

Let $\mathcal{T}$ be a TBox expressed in a DL $\mathcal{L}$, $\Sigma \subseteq Mod(\mathcal{T})$ a set of models of $\mathcal{T}$ and $\mathcal{F}$ the update, i.e., a finite set of ABox assertions such that $Mod(\mathcal{T}) \cap Mod(\mathcal{F}) \neq \emptyset$.

The update of Σ with $\mathcal{F}$ (under fixed $\mathcal{T}$) is the (non-empty) set of models defined as follows

$$\Sigma \circ_{\mathcal{T}} \mathcal{F} = \bigcup_{\mathcal{I} \in \Sigma} U^{\mathcal{T}}(\mathcal{I}, \mathcal{F})$$

Given an ontology $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ we write

$$\mathcal{K} \circ_{\mathcal{T}} \mathcal{F} \quad \text{to mean} \quad Mod(\mathcal{K}) \circ_{\mathcal{T}} \mathcal{F}$$

Example

TBox $\mathcal{T}$: $\{ \quad \exists WillPlay \sqsubseteq AvailablePlayer,$
$AvailablePlayer \sqsubseteq Player,$
$Injured \sqsubseteq \neg AvailablePlayer \quad \}$

ABox $\mathcal{A}$: $\{WillPlay(john, allstargame06)\}$

Note that $AvailablePlayer(john)$, $Player(john)$ and $\neg Injuried(john)$ are true in every model of $\langle \mathcal{T}, \mathcal{A} \rangle$.

Update $\mathcal{F}$: $Injured(john)$

Result $\langle \mathcal{T}, \mathcal{A} \rangle \circ_{\mathcal{T}} \mathcal{F}$: $\{\mathcal{I} \in Mod(\mathcal{T}) \mid john \in Injured^{\mathcal{I}} \wedge john \in Player^{\mathcal{I}}\}$

Note: result according to a syntactic approach is $\langle \mathcal{T}, \{Injured(john)\} \rangle$

Update expressibility

Consider the previous example.

TBox $\mathcal{T}$: $\{ \quad \exists WillPlay \sqsubseteq AvailablePlayer,$
$AvailablePlayer \sqsubseteq Player,$
$Injured \sqsubseteq \neg AvailablePlayer \quad \}$

ABox $\mathcal{A}$: $\{WillPlay(john, allstargame06)\}$

Note that $\langle \mathcal{T}, \mathcal{A} \rangle$ is expressed in *DL-Lite*$_{\mathcal{F}}$.

Update $\mathcal{F}$: $Injured(john)$

Result $\langle \mathcal{T}, \mathcal{A} \rangle \circ_{\mathcal{T}} \mathcal{F}$: $\{\mathcal{I} \in Mod(\mathcal{T}) \mid john \in Injured^{\mathcal{I}} \wedge john \in Player^{\mathcal{I}}\}$

The above set of models are captured by a *DL-Lite*$_{\mathcal{F}}$ ontology.

Result expressed in *DL-Lite*$_{\mathcal{F}}$: $\langle \mathcal{T}, \{Injured(john), Player(john)\} \rangle$

Update expressibility

Let $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ be an ontology expressed in a DL $\mathcal{L}$, and $\mathcal{F}$ a set of membership assertions expressed in $\mathcal{L}$ such that $Mod(\mathcal{T}) \cap Mod(\mathcal{F}) \neq \emptyset$.

We say that the update of $\mathcal{K}$ with $\mathcal{F}$ is expressible in $\mathcal{L}$ if there exists an ABox $\mathcal{A}'$ expressed in $\mathcal{L}$ such that $\mathcal{K} \circ_{\mathcal{T}} \mathcal{F} = Mod(\langle \mathcal{T}, \mathcal{A}' \rangle)$

Results in [Liu&etal '06] show that for several DL languages updates may not be expressible, even if the TBox is not considered.

Example:

TBox: $\forall x(\neg A(x) \vee \neg B(x) \vee \neg C(x))$ ABox: $A(d), B(d)$

Update: $C(d)$

The result of the update should imply $A(d) \vee B(d)$, without implying neither $A(d)$, nor $B(d)$. If the DL $\mathcal{L}$ used to express the ontology does not allow any form of positive disjunction, the update cannot be expressed in $\mathcal{L}$.

The expressibility problem in *DL-Lite*$_{\mathcal{F}}$

The TBox $\{\exists P^- \sqsubseteq A_1, A_2 \sqsubseteq \neg\exists P\}$ and the ABox $\{\exists P(c)\}$ imply that there exists an object that is both a P-successor of c, and an instance of A_1.

Now consider the update $\{A_2(c)\}$. As a result of the update, $A_2(c)$ must be logically implied, hence we must remove $\exists P(c)$ from the ABox, but the fact that there is an instance of A_1 must remain logically implied after the update.

We must then ensure that the new ABox logically implies $\exists z A_1(z)$, with no constant satisfying A_1. Taking into account the form of the original ontology, it can be shown that the new ABox cannot be expressed in *DL-Lite*$_{\mathcal{F}}$.

Expressibility vs representation systems

A representation system $\mathcal{S}$ for incomplete databases is strong wrt a class of updates $\mathcal{L}_u$ if for each $S \in \mathcal{S}$ and each update $u \in \mathcal{L}_u$, there exists an $S' \in \mathcal{S}$ such that $rep(S') = u(rep(S))$.

A representation system $\mathcal{S}$ for incomplete databases is weak wrt a class of updates $\mathcal{L}_u$ and a class of queries $\mathcal{L}_q$ if for each $S \in \mathcal{S}$ and for each update $u \in \mathcal{L}_u$ there exists an $S' \in \mathcal{S}$ such that $rep(S') \equiv_{\mathcal{L}_q} u(rep(S))$.

The above results imply that *DL-Lite*$_{\mathcal{F}}$ is not a strong representation system wrt our update operator.

Also, it can be easily shown that *DL-Lite*$_{\mathcal{F}}$ is not even a weak representation system wrt our update operator and the class of conjunctive queries.

DL-Lite$_{\mathcal{FS}}$: A variant of *DL-Lite*$_{\mathcal{F}}$

Expressions

$$\begin{array}{rcl} B & ::= & A \mid \exists R \\ C & ::= & B \mid \neg B \\ R & ::= & P \mid P^- \end{array}$$

TBox assertions

$$\begin{array}{ll} B \sqsubseteq C & \textbf{inclusion assertion} \\ (funct\ R) & \textbf{functionality assertion} \end{array}$$

ABox assertions (C is a general concept, and z is a variable)

$$C(a), \quad R(a,b), \quad C(z) \quad \textbf{membership assertion}$$

Results on update

- The result of an update is always expressible within our variant of *DL-Lite*$_{\mathcal{F}}$: i.e., there always exists a new ABox that reflects the changes of the update to the original ontology, according to our semantics (obviously the TBox remains unchanged as required)
- The new ABox resulting from an update
  - has a size which is polynomially bounded by the size of the original ontology and the update, and
  - can be computed in polynomial time

Sketch of the algorithm

The algorithm takes as input a *DL-Lite*$_{\mathcal{FS}}$ satisfiable ontology $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$, and a finite set of ground (i.e., not involving variables) membership assertions $\mathcal{F}$, and returns either ERROR (if $\langle \mathcal{T}, \mathcal{F} \rangle$ is unsatisfiable), or an ABox $\mathcal{A}'$ (otherwise).

After the satisfiability check,

- it inserts into $\mathcal{A}'$ all the membership assertions in $\mathcal{A}$ and $\mathcal{F}$
- it computes the set $\mathcal{F}'$ of membership assertions that are logically implied by $\mathcal{K}$ and contradict $\mathcal{F}$ according to $\mathcal{T}$
- Finally, the algorithm deletes all $g \in \mathcal{F}'$ from $\mathcal{A}'$, and inserts into $\mathcal{A}'$ the membership assertions that are logically implied by the membership assertions deleted and do not contradict $\mathcal{F}$

Back to the non-expressibility problem

- Updates cannot be expressed in general in *DL-Lite*$_{\mathcal{F}}$
- We will see that also erasure cannot be expressed in general in *DL-Lite*$_{\mathcal{F}}$, nor even in *DL-Lite*$_{\mathcal{FS}}$
- We need a fundamental tool for addressing the non-expressibility problem

We cope with the non-expressibility (for both update and erasure) issue by means of approximation

The notion of approximation

Let $\mathcal{T}$ be a TBox in a DL $\mathcal{L}$, and let $\Sigma \subseteq \mathit{Mod}(\mathcal{T})$.

The DL ontology $\mathcal{K}$ is a sound $(\mathcal{L}, \mathcal{T})$-approximation of Σ in $\mathcal{L}$, if

- $\mathcal{K}$ is expressed in $\mathcal{L}$,
- $\mathcal{K}$ is of the form $\langle \mathcal{T}, \mathcal{A} \rangle$,
- $\Sigma \subseteq \mathit{Mod}(\mathcal{K})$.

The DL ontology $\mathcal{K}$ is a maximal $(\mathcal{L}, \mathcal{T})$-approximation of Σ if

- $\mathcal{K}$ is a sound $(\mathcal{L}, \mathcal{T})$-approximation of Σ, and
- there exists no ontology $\mathcal{K}'$ that is a sound $(\mathcal{L}, \mathcal{T})$-approximation of Σ, such that $\mathit{Mod}(\mathcal{K}') \subset \mathit{Mod}(\mathcal{K})$.

Properties of approximation

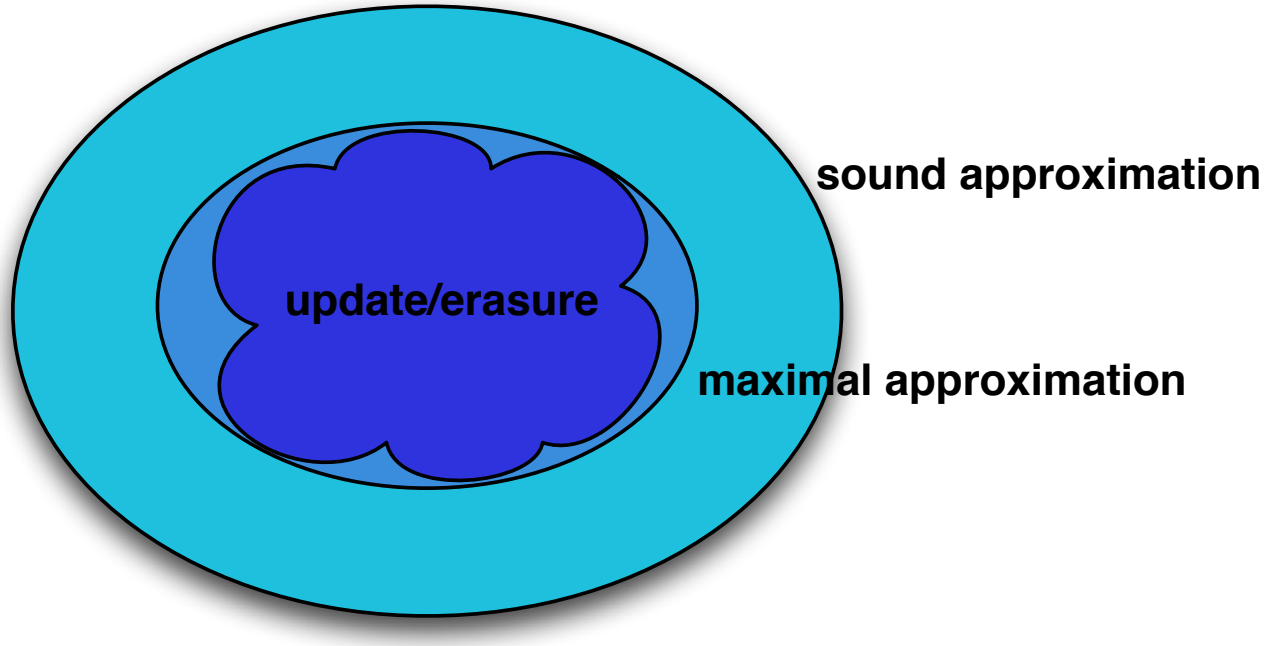
Let Σ be a set of models, and $\mathcal{T}$ a TBox in a DL $\mathcal{L}$ such that $\Sigma \subseteq Mod(\mathcal{T})$.

- If an ontology $\mathcal{K}$ exists that is a maximal $(\mathcal{L}, \mathcal{T})$-approximation of Σ, then all maximal $(\mathcal{L}, \mathcal{T})$-approximations of Σ are equivalent to $\mathcal{K}$.
- If $\mathcal{K}$ is a (or, the) maximal $(\mathcal{L}, \mathcal{T})$-approximation of Σ, then for every membership assertion α in $\mathcal{L}$ it holds that $\Sigma \models \alpha$ iff $\mathcal{K} \models \alpha$.

Thus, the maximal $(\mathcal{L}, \mathcal{T})$-approximation captures exactly all membership assertions in the DL $\mathcal{L}$ that are logically implied by Σ.

Note: maximal $(\mathcal{L}, \mathcal{T})$-approximations may not exist.

Approximation of update



$(\mathcal{L}, \mathcal{T})$-Update

Let $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$, $\mathcal{K}^a = \langle \mathcal{T}, \mathcal{A}^a \rangle$ be two ontologies in a DL $\mathcal{L}$ and $\mathcal{F}$ a finite set of membership assertions expressed in $\mathcal{L}$ such that $Mod(\mathcal{T}) \cap Mod(\mathcal{F}) \neq \emptyset$. We say that $\mathcal{K}^a$ is an $(\mathcal{L}, \mathcal{T})$-update of $\mathcal{K}$ with $\mathcal{F}$ if $\mathcal{K}^a$ is a maximal $(\mathcal{L}, \mathcal{T})$-approximation of $\mathcal{K} \circ_{\mathcal{T}} \mathcal{F}$.

Approximated update in *DL-Lite*$_{\mathcal{F}}$

We assume that $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ is satisfiable, and $Mod(\mathcal{T} \cup \mathcal{F}) \neq \emptyset$.

Algorithm $ComputeUpdate^{app}(\mathcal{T}, \mathcal{A}, \mathcal{F})$:

1. compute the ABox $\mathcal{A}^u = ComputeUpdate(\mathcal{T}, \mathcal{A}, \mathcal{F})$;
2. return the projection of A^u to *DL-Lite*$_{\mathcal{F}}$, i.e. the ABox obtained by deleting from $\mathcal{A}^u$ all the assertions that are not *DL-Lite*$_{\mathcal{F}}$ membership assertions.

Computing the approximated update in *DL-Lite*$_{\mathcal{F}}$ is a polynomial time task.

Moreover: let $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ be a *DL-Lite*$_{\mathcal{F}}$ ontology, let $\mathcal{F}$ be a finite set of *DL-Lite*$_{\mathcal{F}}$ membership assertions such that $Mod(\mathcal{T} \cup \mathcal{F}) \neq \emptyset$, and let $\mathcal{K}^a = \langle \mathcal{T}, \mathcal{A}^a \rangle$, where $\mathcal{A}^a$ is the ABox returned by $ComputeUpdate^{app}(\mathcal{T}, \mathcal{A}, \mathcal{F})$. Then, for every membership assertion α in *DL-Lite*$_{\mathcal{F}}$, we have that $\mathcal{K} \circ_{\mathcal{T}} \mathcal{F} \models \alpha$ iff $\mathcal{K}^a \models \alpha$.

Thus, *DL-Lite*$_{\mathcal{F}}$ is a weak representation system wrt our update operator and the class of atomic queries.

Instance-level erasure

Intuitively, the erasure of a set σ of membership assertions from a knowledge base means that we aim at changing the knowledge base in such a way that none of the assertions in σ is logically implied

If $\mathcal{F}$ is a set of membership assertions, $\neg\mathcal{F}$ denotes the set of membership assertions $\{\neg g_i \mid g_i \in \mathcal{F}\}$

Let $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ be an ontology expressed in a DL $\mathcal{L}$, and $\mathcal{F}$ a finite set of membership assertions expressed in $\mathcal{L}$ such that $Mod(\mathcal{T} \cup \neg\mathcal{F}) \neq \emptyset$

The erasure of $\mathcal{K}$ with $\mathcal{F}$ is defined as follows:

$$\mathcal{K} \bullet_{\mathcal{T}} \mathcal{F} = Mod(\mathcal{K}) \cup \Big(\bigcup_{\mathcal{I} \in Mod(\mathcal{K})} U^{\mathcal{T}}(\mathcal{I}, \neg\mathcal{F}) \Big).$$

Expressibility of erasure in *DL-Lite*$_{\mathcal{F}}$

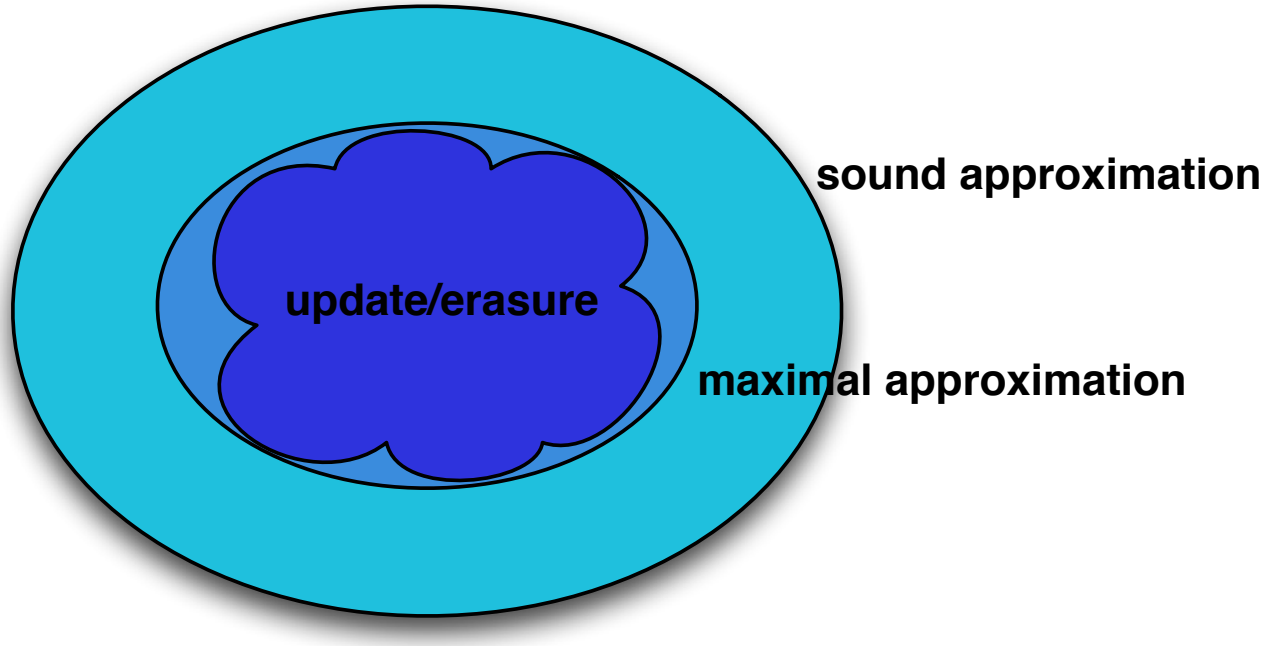
The result of the erasure cannot be always expressed in terms of a *DL-Lite*$_{\mathcal{F}}$ ontology, and, differently from the case of the update, neither in terms of *DL-Lite*$_{\mathcal{FS}}$.

Let $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ be the *DL-Lite*$_{\mathcal{F}}$ ontology such that $\mathcal{T} = \{A \sqsubseteq B, A \sqsubseteq C\}$ and $\mathcal{A} = \{A(d)\}$. Let $\mathcal{F} = \{C(d)\}$. By definition,

$$\mathcal{K} \bullet_{\mathcal{T}} \mathcal{F} = Mod(\mathcal{K}) \cup (\mathcal{K} \circ_{\mathcal{T}} \{\neg C(a)\})$$

Thus, each model $\mathcal{I}^p$ in $\mathcal{K} \bullet_{\mathcal{T}} \mathcal{F}$ is obtained from a model $\mathcal{I}$ of $\mathcal{K}$ by either not modifying anything, or by modifying the interpretation of d so that d does not belong to $A^{\mathcal{I}^p}$. Hence, for each $\mathcal{I}^p$, either $d \in A^{\mathcal{I}^p}, d \in C^{\mathcal{I}^p}$ or $d \notin A^{\mathcal{I}^p}, d \notin C^{\mathcal{I}^p}$. Clearly there is no *DL-Lite*$_{\mathcal{FS}}$ ontology that is able to capture this set of models.

Approximation of erasure



$(\mathcal{L}, \mathcal{T})$-Erasure

Let $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$, $\mathcal{K}^a = \langle \mathcal{T}, \mathcal{A}^a \rangle$ be two ontologies in a DL $\mathcal{L}$ and $\mathcal{F}$ a finite set of membership assertions expressed in $\mathcal{L}$ such that $Mod(\mathcal{T}) \cap Mod(\neg\mathcal{F}) \neq \emptyset$. We say that $\mathcal{K}^a$ is an $(\mathcal{L}, \mathcal{T})$-erasure of $\mathcal{K}$ with $\mathcal{F}$ if $\mathcal{K}^a$ is a maximal $(\mathcal{L}, \mathcal{T})$-approximation of $\mathcal{K} \bullet_{\mathcal{T}} \mathcal{F}$.

Approximated erasure in *DL-Lite*$_{\mathcal{F}}$

We assume that $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ is consistent and that $Mod(\mathcal{T} \cup \neg\mathcal{F}) \neq \emptyset$.

Algorithm $ComputeErasure^{app}(\mathcal{T}, \mathcal{A}, \mathcal{F})$:

1. compute the *DL-Lite*$_{\mathcal{FS}}$ ABox $\mathcal{A}^u = ComputeUpdate(\mathcal{T}, \mathcal{A}, \neg\mathcal{F})$;
2. compute the projection of $\mathcal{A}^u$ to *DL-Lite*$_{\mathcal{F}}$, i.e. deletes from $\mathcal{A}^u$ all the assertions that are not *DL-Lite*$_{\mathcal{F}}$ membership assertions.

The algorithm $ComputeErasure^{app}$ terminates and runs in time polynomial with respect to the size of its input.

Moreover: let $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ be a *DL-Lite*$_{\mathcal{F}}$ ontology, let $\mathcal{F}$ be a finite set of *DL-Lite*$_{\mathcal{F}}$ membership assertions such that $Mod(\mathcal{T} \cup \neg\mathcal{F}) \neq \emptyset$, and let $\mathcal{K}^a = \langle \mathcal{T}, \mathcal{A}^a \rangle$, where $\mathcal{A}^a$ is the ABox returned by $ComputeErasure^{app}(\mathcal{T}, \mathcal{A}, \mathcal{F})$. Then, for every membership assertion α in *DL-Lite*$_{\mathcal{F}}$, we have that $\mathcal{K} \bullet_{\mathcal{T}} \mathcal{F} \models \alpha$ iff $\mathcal{K}^a \models \alpha$.

Thus, *DL-Lite*$_{\mathcal{F}}$ is a weak representation system wrt our erasure operator and the class of atomic queries.

Example

Let $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ be the *DL-Lite*$_\mathcal{F}$ ontology such that $\mathcal{T} = \{A \sqsubseteq B, A \sqsubseteq C\}$ and $\mathcal{A} = \{A(d)\}$. Let $\mathcal{F} = \{C(d)\}$.

We want to compute the (*DL-Lite*$_\mathcal{F}$, $\mathcal{T}$)-erasure of $\mathcal{K}$ with $\mathcal{F} = \{C(d)\}$. First, we apply the update algorithm $\mathit{ComputeUpdate}$ and compute the ABox $\mathcal{A}^u = \mathit{ComputeUpdate}(\mathcal{T}, \mathcal{A}, \{\neg C(d)\})$. This returns an ABox that is obtained from $\mathcal{A}$ by removing the assertion $A(d)$, and introducing the assertions $\neg C(d)$ and $B(d)$. Second, we perform the projection of $\mathcal{A}^u$ in *DL-Lite*$_\mathcal{F}$, and obtain the *DL-Lite*$_\mathcal{F}$ ABox $\mathcal{A}^a = \{B(d)\}$.

Update and erasure in data integration systems

Recall that the set $Mod(\langle \mathcal{T}, \mathcal{M}, DB \rangle)$ of models of $\langle \mathcal{T}, \mathcal{M}, DB \rangle$ is the set of interpretations $\mathcal{I} = (\Delta^I, \cdot^I)$ such that:

- $\mathcal{I}$ is a model of $\mathcal{T}$;
- $\mathcal{I}$ satisfies $\mathcal{M}$ wrt DB, i.e., satisfies every assertion in $\mathcal{M}$ wrt DB.

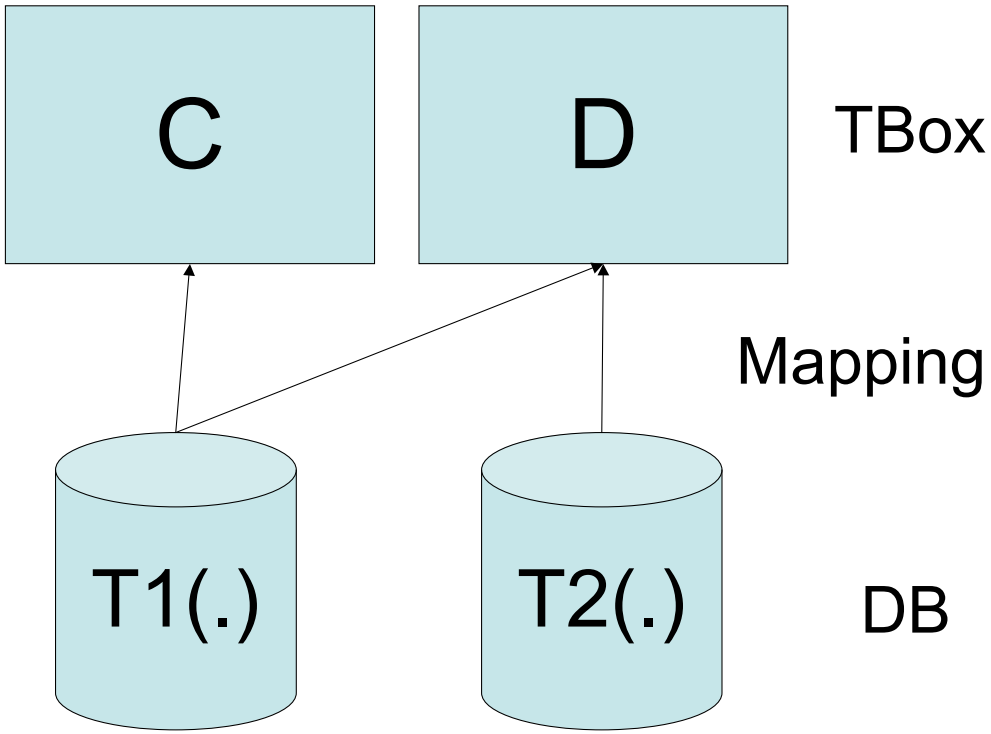
We say that DB' realizes the update of $\langle \mathcal{T}, \mathcal{M}, DB \rangle$ with $\mathcal{F}$ if

$$Mod(\langle \mathcal{T}, \mathcal{M}, DB' \rangle) = Mod(\langle \mathcal{T}, \mathcal{M}, DB \rangle) \circ_{\mathcal{T}} \mathcal{F}$$

We say that DB' realizes the erasure of $\langle \mathcal{T}, \mathcal{M}, DB \rangle$ with $\mathcal{F}$ if

$$Mod(\langle \mathcal{T}, \mathcal{M}, DB' \rangle) = Mod(\langle \mathcal{T}, \mathcal{M}, DB \rangle) \bullet_{\mathcal{T}} \mathcal{F}$$

Update and erasure in data integration systems

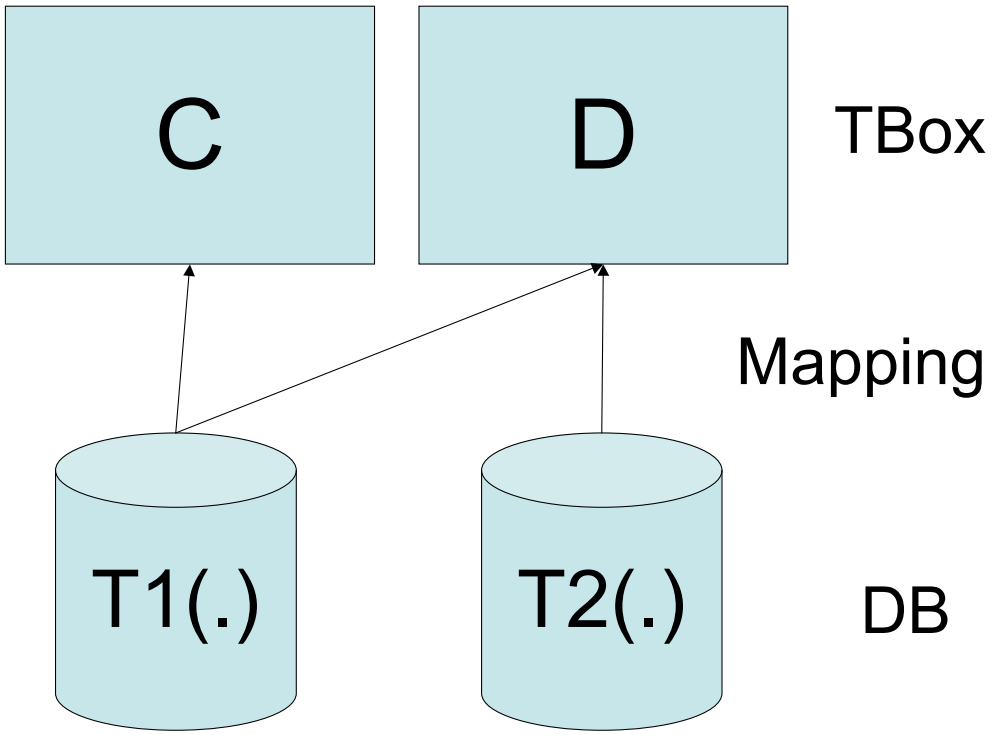


Suppose $C(a)$ is not logically implied by $\langle \mathcal{T}, \mathcal{M}, DB \rangle$.

Update $\{ C(a) \}$:

not realizable

Update and erasure in data integration systems

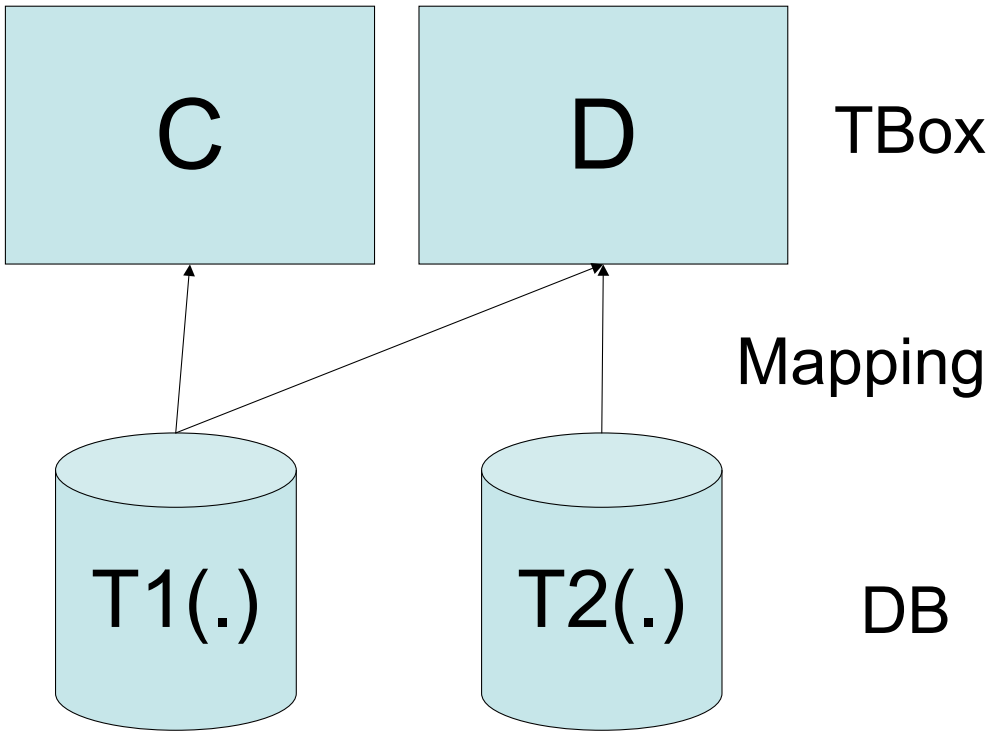


Suppose $C(a), D(a)$ are not logically implied by $\langle \mathcal{T}, \mathcal{M}, DB \rangle$.

Update $\{ C(a), D(a) \}$:

realizable by inserting $T1(a)$ in DB, or by inserting $T1(a), T2(a)$ in DB

Update and erasure in data integration systems



Suppose $C(a), D(a)$ are logically implied by $\langle \mathcal{T}, \mathcal{M}, DB \rangle$.

Erase $\{ C(a) \}$:

realizable by removing $T1(a)$ and inserting $T2(a)$ in DB

Conclusions

- We have presented a study of instance-level update and erasure in DLs, and we have resorted to the notion of approximation for dealing with the non-expressibility problem
- Algorithms and complexity analysis for approximated update and erasure in *DL-Lite*$_{\mathcal{F}}$
- Algorithms implemented in the Mastro system (without mappings)

Future work:

- "Write-also" data integration (first results on "simple" mappings", but what about complex mappings?)
- Study the connection between pushing updates to the data sources and computing "inverse mappings" in data exchange
- Study the connection between pushing updates to the data sources and view update