Description Logics

Meghyn Bienvenu (CNRS – Univ. Paris Sud)

December 10, 2012

Next two courses: reasoning with description logics

We will start by giving a general introduction to the area.

Then we will present the two main classes of lightweight DLs:

- $\mathcal{EL}$
- DL-Lite

For DL-Lite, we will also show how reasoning can be distributed.

1. Brief Introduction to DLs

Motivation: why ontologies?

Ontologies are logical theories that formalize domain-specific knowledge, thereby making it available for machine processing.

Motivation: why ontologies?

Ontologies are logical theories that formalize domain-specific knowledge, thereby making it available for machine processing.

There are many reasons for adopting ontologies:

- to standardize the terminology of an application domain
  - meaning of terms is unambiguous, so less misunderstandings
  - by adopting a common vocabulary, easy to share information

Motivation: why ontologies?

Ontologies are logical theories that formalize domain-specific knowledge, thereby making it available for machine processing.

There are many reasons for adopting ontologies:

- to standardize the terminology of an application domain
  - meaning of terms is unambiguous, so less misunderstandings
  - by adopting a common vocabulary, easy to share information
- to support complex reasoning
  - uncover implicit connections between terms, or errors in modelling
  - exploit knowledge in the ontology during query answering, to get back a more complete set of answers to queries

Motivation: why ontologies?

Ontologies are logical theories that formalize domain-specific knowledge, thereby making it available for machine processing.

There are many reasons for adopting ontologies:

- to standardize the terminology of an application domain
  - meaning of terms is unambiguous, so less misunderstandings
  - by adopting a common vocabulary, easy to share information
- to support complex reasoning
  - uncover implicit connections between terms, or errors in modelling
  - exploit knowledge in the ontology during query answering, to get back a more complete set of answers to queries
- to present an intuitive and unified view of data sources
  - ontology can be used to enrich the data vocabulary, making it easier for users to formulate their queries
  - especially useful when integrating multiple data sources

Description logics as ontology languages

Description logics (DLs) are a popular way of specifying ontologies.

Formally, description logics are a family of fragments of first-order logic.

Wide variety of DLs, offering different levels of expressivity.

Complexity of reasoning well-understood (from PTIME to 2EXPTIME and beyond).

Lots of work on reasoning algorithms, DL reasoners available for use.

DLs form the basis of the web ontology language OWL (W3C standard).

DL basics: syntax

Basic building blocks

- atomic concepts (unary relations) — *Mother*, *Student*
- atomic roles (binary relations) — *ParentOf*, *PartOf*
- individuals (constants) — *marie*, *pierre*

DL basics: syntax

Basic building blocks

- atomic concepts (unary relations) — *Mother, Student*
- atomic roles (binary relations) — *ParentOf, PartOf*
- individuals (constants) — *marie, pierre*

Complex concepts

- concept constructors: $\top$, $\neg C$, $C \sqcap D$, $C \sqcup D$, $\exists R.C$, $\geq n\, R.C$, ...
- some examples to illustrate:
  - "Persons who are not parents": $\mathit{Person} \sqcap \neg \mathit{Parent}$
  - "Persons with at least one female child": $\mathit{Person} \sqcap \exists \mathit{ParentOf}.\mathit{Female}$
  - "Persons having at least three children": $\mathit{Person} \sqcap \geq 3\, \mathit{ParentOf}.\top$

DL basics: syntax

Basic building blocks

- atomic concepts (unary relations) — *Mother, Student*
- atomic roles (binary relations) — *ParentOf, PartOf*
- individuals (constants) — *marie, pierre*

Complex concepts

- concept constructors: $\top$, $\neg C$, $C \sqcap D$, $C \sqcup D$, $\exists R.C$, $\geq n\, R.C$, ...
- some examples to illustrate:
  - "Persons who are not parents": $\mathit{Person} \sqcap \neg \mathit{Parent}$
  - "Persons with at least one female child": $\mathit{Person} \sqcap \exists \mathit{ParentOf}.\mathit{Female}$
  - "Persons having at least three children": $\mathit{Person} \sqcap \geq 3\, \mathit{ParentOf}.\top$

Complex roles:

- role constructors: $^-$ (inverse), $\circ$ (composition), ...

DL basics: TBox and ABox

DL knowledge base = **TBox** (ontology) + **ABox** (data)

DL basics: TBox and ABox

DL knowledge base = **TBox** (ontology) + **ABox** (data)

A TBox describes *general knowledge about the domain*. It contains:

- concept inclusions $C \sqsubseteq D$, with C, D complex concepts
  use $C \equiv D$ when both $C \sqsubseteq D$ and $D \sqsubseteq C$

  $$Mother \equiv Female \sqcap \exists ParentOf.\top \qquad Parent \equiv Mother \sqcup Father$$

DL basics: TBox and ABox

DL knowledge base = **TBox** (ontology) + **ABox** (data)

A TBox describes *general knowledge about the domain*. It contains:

- concept inclusions $C \sqsubseteq D$, with C, D complex concepts
  use $C \equiv D$ when both $C \sqsubseteq D$ and $D \sqsubseteq C$

  $$Mother \equiv Female \sqcap \exists ParentOf.\top \qquad Parent \equiv Mother \sqcup Father$$

- in some DLs, also have role inclusions $R \sqsubseteq S$, or can specify other properties about roles (e.g. transitivity or functionality)

  $$ParentOf \equiv ChildOf^{-} \qquad ParentOf \sqsubseteq AncestorOf \qquad (\text{trans } AncestorOf)$$

DL basics: TBox and ABox

DL knowledge base = **TBox** (ontology) + **ABox** (data)

A TBox describes *general knowledge about the domain*. It contains:

- concept inclusions $C \sqsubseteq D$, with C, D complex concepts
  use $C \equiv D$ when both $C \sqsubseteq D$ and $D \sqsubseteq C$

  $$Mother \equiv Female \sqcap \exists ParentOf.\top \qquad Parent \equiv Mother \sqcup Father$$

- in some DLs, also have role inclusions $R \sqsubseteq S$, or can specify other properties about roles (e.g. transitivity or functionality)

  $$ParentOf \equiv ChildOf^{-} \qquad ParentOf \sqsubseteq AncestorOf \qquad (\text{trans } AncestorOf)$$

An ABox contains *facts about specific individuals*. It contains:

- concept assertions $C(a)$ and role assertions $R(a, b)$

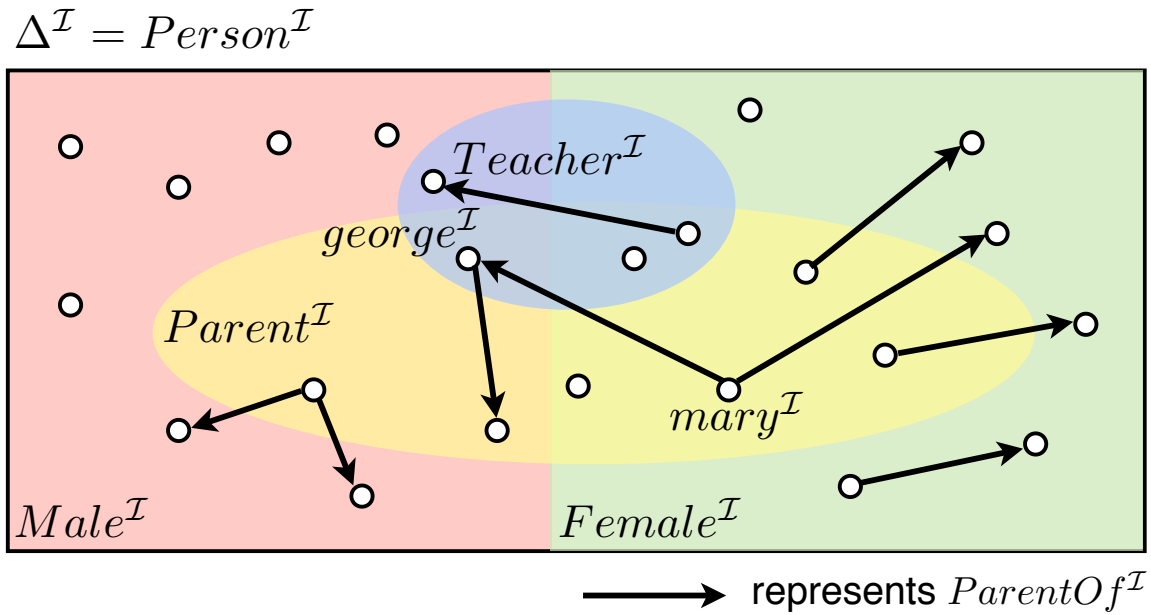
  $$ParentOf(mary, george) \qquad ChildOf(paul, george) \qquad AncestorOf(mary, paul)$$

DL basics: semantics

Interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$

- $\Delta^{\mathcal{I}}$ is a non-empty set (universe)
- $\cdot^{\mathcal{I}}$ is a function which maps
  - individual $a \mapsto$ an element of the universe $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$
  - atomic concept $A \mapsto$ unary relation $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$
  - atomic role $R \mapsto$ binary relation $R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$

Example interpretation:



DL basics: semantics

We next extend the function $\cdot^{\mathcal{I}}$ to complex concepts and roles, to formalize the meaning of the constructors.

- $\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$ and $\bot^{\mathcal{I}} = \emptyset$
- $(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$
- $(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$
- $(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
- $(\exists R.C)^{\mathcal{I}} = \{u \mid \text{exists } v \text{ such that } (u, v) \in R^{\mathcal{I}} \text{ and } v \in C^{\mathcal{I}}\}$
- $(\forall R.C)^{\mathcal{I}} = \{u \mid \text{for every } v \text{ such that } (u, v) \in R^{\mathcal{I}} \text{ we have } v \in C^{\mathcal{I}}\}$
- $(\geq n\, R.C)^{\mathcal{I}} = \{u \mid \text{at least } n\ v \text{ such that } (u, v) \in R^{\mathcal{I}} \text{ and } v \in C^{\mathcal{I}}\}$
- $R^{-} = \{(u, v) | (v, u) \in R^{\mathcal{I}}\}$

DL basics: semantics

Satisfaction of TBox statements:

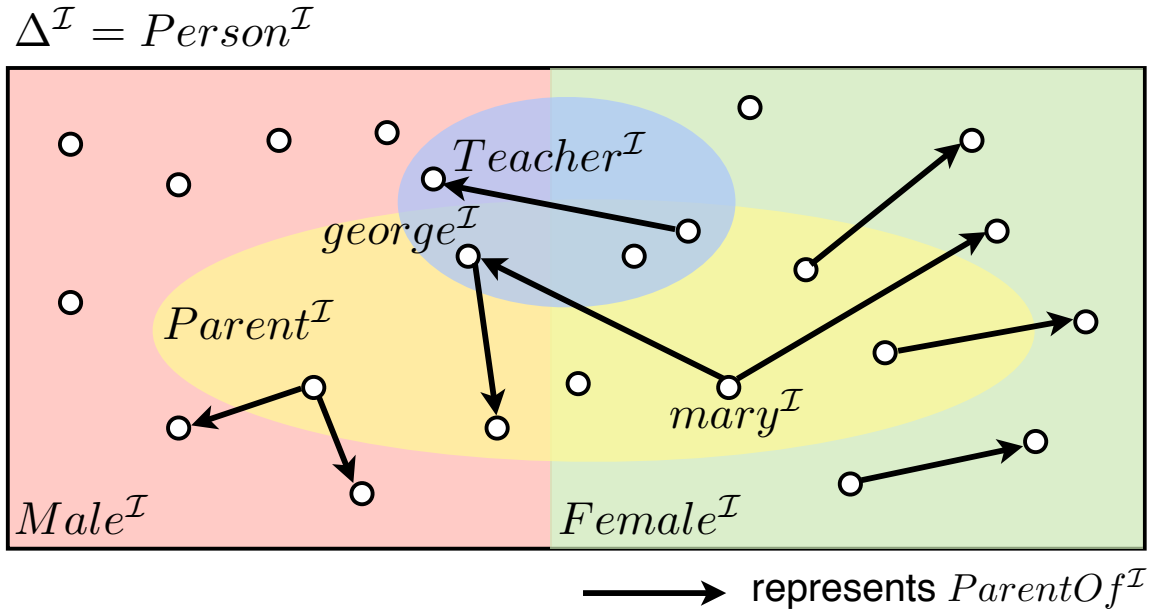
- $\mathcal{I}$ satisfies an inclusion $C \sqsubseteq D$ if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
- $\mathcal{I}$ satisfies an inclusion $R \sqsubseteq S$ if $R^{\mathcal{I}} \subseteq S^{\mathcal{I}}$
- $\mathcal{I}$ satisfies $(\text{trans}\, R)$ is $R^{\mathcal{I}}$ is a transitive relation
- ...

Satisfaction of ABox assertions:

- $\mathcal{I}$ satisfies an assertion $C(a)$ if $a^{\mathcal{I}} \in C^{\mathcal{I}}$
- $\mathcal{I}$ satisfies an assertion $R(a, b)$ if $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$

Example

Reconsider the interpretation $\mathcal{I}$:



Which of the following are satisfied in $\mathcal{I}$?

$Person \sqsubseteq Male \sqsubseteq Female$

$Female \sqsubseteq Parent$

$Parent \equiv \exists ParentOf.\top$

$\exists ParentOf.Male(mary)$

$Male \sqsubseteq \forall ParentOf.Male$

$Person \sqsubseteq \exists ParentOf.\top$

$Person \equiv \leq 2.ParentOf.\top$

$Teacher(george)$

DL basics: semantics

Models:

- $\mathcal{I}$ is a model of TBox $\mathcal{T}$ if $\mathcal{I}$ satisfies every statement in $\mathcal{T}$
- $\mathcal{I}$ is a model of ABox $\mathcal{A}$ if $\mathcal{I}$ satisfies every assertion in $\mathcal{A}$
- $\mathcal{I}$ is a model of KB $(\mathcal{T}, \mathcal{A})$ if $\mathcal{I}$ is a model of $\mathcal{T}$ and $\mathcal{A}$

Entailment:

- A TBox $\mathcal{T}$ entails an inclusion α (written $\mathcal{T} \models \alpha$)
  if every model of $\mathcal{T}$ satisfies α
- A KB $(\mathcal{T}, \mathcal{A})$ entails an ABox assertion α (written $(\mathcal{T}, \mathcal{A}) \models \alpha$)
  if every model $\mathcal{I}$ of $(\mathcal{T}, \mathcal{A})$ satisfies α

Defining a particular DL

To define a description logic, we need to specify:

- which concept constructors can be used
- which role constructors can be used
- what types of statements can appear in the TBox

For example, the DL $\mathcal{ALC}$ is defined by:

- concept constructors: $\top, \bot, \neg, \sqcup, \sqcap, \forall R.C, \exists R.C$
- no role constructors
- concept inclusions

The more expressive $\mathcal{SHIQ}$ is defined by:

- all $\mathcal{ALC}$ concept constructors, plus: $\geq n\, R.C, \leq n\, R.C$
- inverse roles (R^-)
- concept inclusions, role inclusions, and transitivity statements

Reasoning tasks

Classical reasoning tasks:

subsumption does $\mathcal{T} \models C \sqsubseteq D$?

classification find all atomic A,B such that $\mathcal{T} \models A \sqsubseteq B$

satisfiability is $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ satisfiable? (i.e. has a model)

instance queries does $(\mathcal{T}, \mathcal{A}) \models C(b)$?

Reasoning tasks

Classical reasoning tasks:

subsumption does $\mathcal{T} \models C \sqsubseteq D$?

classification find all atomic A,B such that $\mathcal{T} \models A \sqsubseteq B$

satisfiability is $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ satisfiable? (i.e. has a model)

instance queries does $(\mathcal{T}, \mathcal{A}) \models C(b)$?

Relations between these tasks:

- $\mathcal{T} \models C \sqsubseteq D$ if and only if $(\mathcal{T}, \{C(a)\}) \models D(a)$
- $\mathcal{K}$ satisfiable if and only if $\mathcal{K} \not\models B(a)$ (where B does not appear in $\mathcal{K}$)

Short history of DLs

1985-1995 Negative results (undecidability, or NP-hardness)

Tractable logics (e.g. $\mathcal{AL}$) based on $\sqcap$ and $\forall R.C$

Complexity: subsumption in P (without TBox)

Algorithms: structural subsumption

Short history of DLs

1985-1995 Negative results (undecidability, or NP-hardness)

Tractable logics (e.g. $\mathcal{AL}$) based on $\sqcap$ and $\forall R.C$

Complexity: subsumption in P (without TBox)

Algorithms: structural subsumption

1995-2005 Expressive logics like $\mathcal{SROIQ} \approx$ OWL 2 which offer:
$\neg$, $\sqcup$, $\exists R.C$, $\geq nR.C$, $\leq R.C$, R^{-}, $\{a\}$, $R \sqsubseteq S$, $(\text{trans}\, R)$, ...

Complexity: subsumption $\geq$ EXPTIME (with TBox)

Algorithms: tableaux method with optimizations

Despite high complexity, algorithms seem to work !

New challenges

1. **Large ontologies and lots of data**
   - need reasoning algorithms which scale up

2. **More expressive queries**
   - conjunctive queries (like in databases)

     $q(x, z) = \mathit{Female}(x) \wedge \mathit{ParentOf}(x, y) \wedge \mathit{Female}(y) \wedge \mathit{ParentOf}(y, z) \wedge \mathit{Female}(z)$
   - approach called "ontology-based data access" (OBDA)
   - difficulty: not reducible to classical reasoning tasks

Solution: new DLs of low complexity

Short history of DLs

1985-1995 Negative results (undecidability, or NP-hardness)
Tractable logics (e.g. $\mathcal{AL}$) based on $\sqcap$ and $\forall R.C$
Complexity: subsumption in P (without TBox)
Algorithms: normalization + structural comparison

1995-2005 Expressive logics like $\mathcal{SROIQ} \approx$ OWL 2 which offer:
$\neg$, $\sqcup$, $\exists R.C$, $\geq nR.C$, $\leq nR.C$, R^-, $\{a\}$, $R \sqsubseteq S$, $(\text{trans}\, R)$, ...
Complexity: subsumption $\geq$ EXPTIME (with TBox)
Algorithms: tableaux method with optimizations
Despite high complexity, algorithms seem to work !

2005-now Lightweight DLs, motivated by applications
$\mathcal{EL}$ family (OWL 2 EL) and *DL-Lite* family (OWL 2 QL)
Algorithms: forward chaining / saturation, query rewriting

2. $\mathcal{EL}$

The $\mathcal{EL}$ family

The logic $\mathcal{EL}$, and its extensions, are designed for applications requiring very large ontologies.

This family of DLs is well-suited for biomedical applications.

Examples of large biomedical ontologies:

- GO (Gene Ontology), around 20,000 concepts
- NCI (cancer ontology), around 30,000 concepts
- SNOMED (medical ontology), close to 400,000 concepts (!)

$$\mathsf{Pericarditis} \sqsubseteq \mathsf{Inflammation} \sqcap \exists \mathsf{loc}.\mathsf{Pericardium}$$
$$\mathsf{Pericardium} \sqsubseteq \mathsf{Tissue} \sqcap \exists \mathsf{partOf}.\mathsf{Heart} \qquad \mathsf{Inflammation} \sqsubseteq \mathsf{Disease}$$
$$\mathsf{Disease} \sqcap \exists \mathsf{loc}.\exists \mathsf{partOf}.\mathsf{Heart} \sqsubseteq \mathsf{HeartDisease}$$

Syntax of $\mathcal{EL}$

The basic logic $\mathcal{EL}$ allows complex concepts of the following form:

$$C := \top \mid A \mid C_1 \sqcap C_2 \mid \exists R.C$$

Inclusions $C_1 \sqsubseteq C_2$ and assertions $A(c)$, $R(c, d)$

Possible extensions:

- $\bot$ (to express disjoint classes)
- domain restrictions $\mathrm{dom}(R) \sqsubseteq C$
- range restrictions $\mathrm{range}(R) \sqsubseteq C$
- complex role inclusions $R_1 \circ \ldots \circ R_n \sqsubseteq R_{n+1}$ (transitive roles:$R \circ R \sqsubseteq R$)

OWL 2 EL includes all these extensions.

Canonical Model

We have seen that entailment of an inclusion or assertion involves considering all the models of the KB.

In $\mathcal{EL}$, for every KB $\mathcal{K}$, there is a single model $\mathcal{I}_\mathcal{K}$ which is guaranteed to give us the correct answers. Formally:

$$\mathcal{K} \models \alpha \quad \text{iff} \quad \mathcal{I}_\mathcal{K} \text{ satisfies } \alpha$$

for every TBox statement or ABox assertion α[1].

We call $\mathcal{I}_\mathcal{K}$ the canonical model of $\mathcal{K}$.

Intuitively, $\mathcal{I}_\mathcal{K}$ is obtained by an exhaustive application of the inclusions.

[1] Actually, $\mathcal{I}_\mathcal{K}$ also gives the right answers for conjunctive queries.

Example: construction of $\mathcal{I}_{\mathcal{K}}$

$\mathcal{K} =$

$R(a, b) \quad C(b)$

$C \sqsubseteq E \quad D \sqsubseteq \exists S.(B \sqcap D)$

$D \sqsubseteq \exists R.(A \sqcap D) \quad \exists R.E \sqsubseteq D$

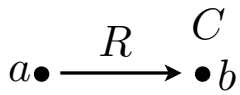
Example: construction of $\mathcal{I}_{\mathcal{K}}$

$\mathcal{K} =$

$R(a, b) \quad C(b)$

$C \sqsubseteq E \quad D \sqsubseteq \exists S.(B \sqcap D)$

$D \sqsubseteq \exists R.(A \sqcap D) \quad \exists R.E \sqsubseteq D$



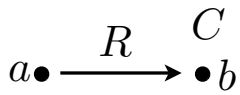
Example: construction of $\mathcal{I}_{\mathcal{K}}$

$\mathcal{K} =$

$R(a,b) \quad C(b)$

$C \sqsubseteq E \quad D \sqsubseteq \exists S.(B \sqcap D)$

$D \sqsubseteq \exists R.(A \sqcap D) \quad \exists R.E \sqsubseteq D$



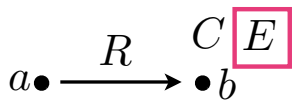
Example: construction of $\mathcal{I}_{\mathcal{K}}$

$\mathcal{K} =$

$R(a,b) \quad C(b)$

$C \sqsubseteq E \quad D \sqsubseteq \exists S.(B \sqcap D)$

$D \sqsubseteq \exists R.(A \sqcap D) \quad \exists R.E \sqsubseteq D$



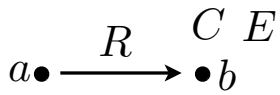
Example: construction of $\mathcal{I}_{\mathcal{K}}$

$\mathcal{K} =$

$R(a, b) \qquad C(b)$

$C \sqsubseteq E \qquad D \sqsubseteq \exists S.(B \sqcap D)$

$D \sqsubseteq \exists R.(A \sqcap D) \qquad \boxed{\exists R.E \sqsubseteq D}$



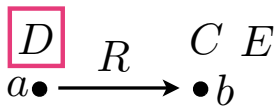
Example: construction of $\mathcal{I}_{\mathcal{K}}$

$\mathcal{K} =$

$R(a, b) \quad C(b)$

$C \sqsubseteq E \quad D \sqsubseteq \exists S.(B \sqcap D)$

$D \sqsubseteq \exists R.(A \sqcap D) \quad \boxed{\exists R.E \sqsubseteq D}$



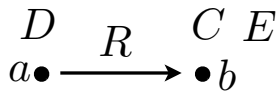
Example: construction of $\mathcal{I}_{\mathcal{K}}$

$\mathcal{K} =$

$R(a,b) \quad C(b)$

$C \sqsubseteq E \quad D \sqsubseteq \exists S.(B \sqcap D)$

$D \sqsubseteq \exists R.(A \sqcap D) \quad \exists R.E \sqsubseteq D$



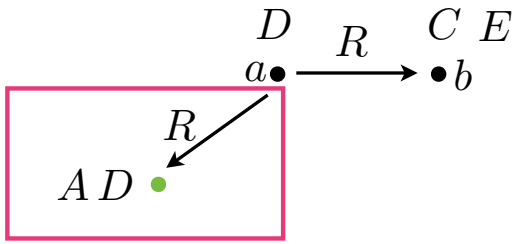
Example: construction of $\mathcal{I}_{\mathcal{K}}$

$\mathcal{K} =$

$R(a,b) \quad C(b)$

$C \sqsubseteq E \quad D \sqsubseteq \exists S.(B \sqcap D)$

$D \sqsubseteq \exists R.(A \sqcap D) \quad \exists R.E \sqsubseteq D$



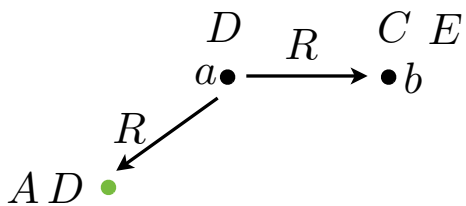
Example: construction of $\mathcal{I}_{\mathcal{K}}$

$\mathcal{K} =$

$R(a,b) \quad C(b)$

$C \sqsubseteq E \quad \boxed{D \sqsubseteq \exists S.(B \sqcap D)}$

$D \sqsubseteq \exists R.(A \sqcap D) \quad \exists R.E \sqsubseteq D$



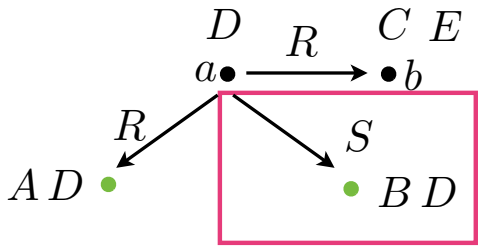
Example: construction of $\mathcal{I}_{\mathcal{K}}$

$\mathcal{K} =$

$R(a,b) \quad C(b)$

$C \sqsubseteq E \quad \boxed{D \sqsubseteq \exists S.(B \sqcap D)}$

$D \sqsubseteq \exists R.(A \sqcap D) \quad \exists R.E \sqsubseteq D$



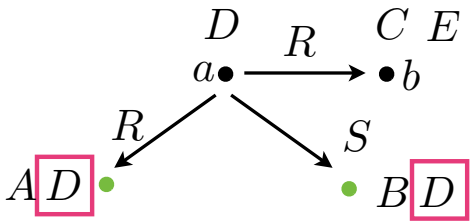
Example: construction of $\mathcal{I}_{\mathcal{K}}$

$\mathcal{K} =$

$R(a,b) \quad C(b)$

$C \sqsubseteq E \quad D \sqsubseteq \exists S.(B \sqcap D)$

$D \sqsubseteq \exists R.(A \sqcap D) \quad \exists R.E \sqsubseteq D$



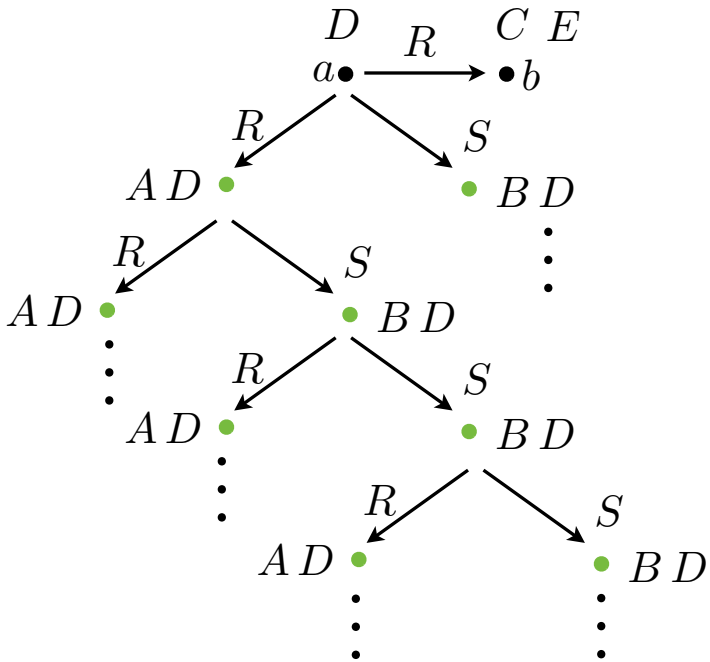
Example: construction of $\mathcal{I}_{\mathcal{K}}$

$\mathcal{K} =$

$R(a,b) \quad C(b)$

$C \sqsubseteq E \quad \boxed{D \sqsubseteq \exists S.(B \sqcap D)}$

$\boxed{D \sqsubseteq \exists R.(A \sqcap D)} \quad \exists R.E \sqsubseteq D$



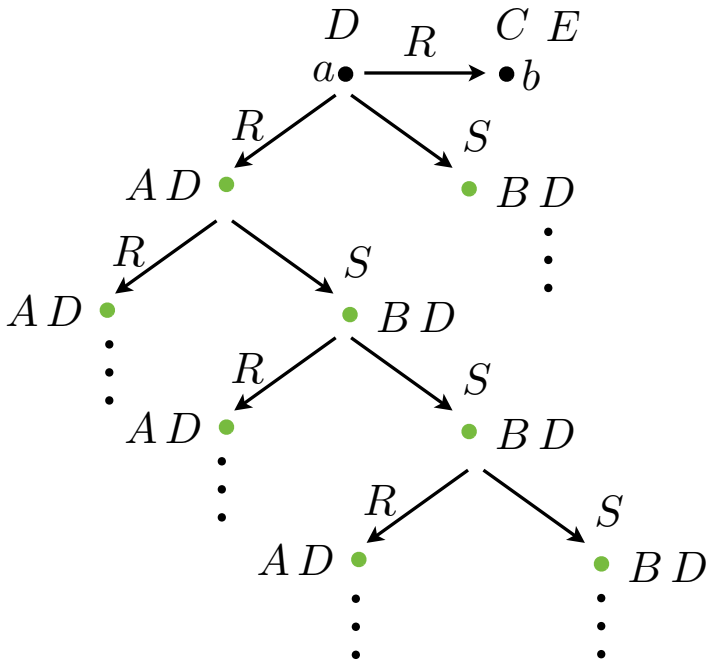
Example: construction of $\mathcal{I}_\mathcal{K}$

$\mathcal{K} =$

$R(a,b) \quad C(b)$

$C \sqsubseteq E \quad D \sqsubseteq \exists S.(B \sqcap D)$

$D \sqsubseteq \exists R.(A \sqcap D) \quad \exists R.E \sqsubseteq D$



Continuing forever...

we obtain the canonical model $\mathcal{I}_\mathcal{K}$

Compact canonical model

Problem: even though $\mathcal{I}_\mathcal{K}$ gives the right answers, we cannot use it directly since it may be infinite.

Solution: make it finite by exploiting repetitions in $\mathcal{I}_\mathcal{K}$

We define a compact canonical model $\mathcal{C}_\mathcal{K}$, which is a finite representation of $\mathcal{I}_\mathcal{K}$ and can be used for reasoning.

First, we must introduce a normal form for $\mathcal{EL}$ TBoxes.

Normalization of TBoxes

We say that an $\mathcal{EL}$ TBox $\mathcal{T}$ is in normal form if it contains only inclusions of the following forms:

$$A_1 \sqcap A_2 \sqsubseteq B \quad A \sqsubseteq \exists R.B \quad \exists R.A \sqsubseteq B$$

where A, A_1, A_2, B are atomic concepts (or $\top$).

For every TBox $\mathcal{T}$, we can construct in polynomial time a TBox $\mathcal{T}'$ in normal form such that:

- for every inclusion $C \sqsubseteq D$ which uses only atomic concepts from $\mathcal{T}$, we have $\mathcal{T} \models C \sqsubseteq D$ iff $\mathcal{T}' \models C \sqsubseteq D$

Note that $\mathcal{T}'$ may use extra atomic concepts which do not appear in $\mathcal{T}$.

Example: the inclusion $\exists R.(\exists S.A \sqcap H) \sqsubseteq B \sqcap D$ would be replaced by

$$\exists R.E \sqsubseteq B \quad \exists R.E \sqsubseteq D \quad E \sqsubseteq F \quad E \sqsubseteq H \quad F \sqcap H \sqsubseteq E \quad F \sqsubseteq \exists S.A \quad \exists S.A \sqsubseteq F$$

Definition of $\mathcal{C}_\mathcal{K}$

Let $\mathcal{K}$ be composed of an ABox $\mathcal{A}$ (with individuals $\mathsf{Ind}(\mathcal{A})$) and a TBox $\mathcal{T}$ in normal form.

We begin by defining an initial interpretation $\mathcal{I}_0$ as follows:

- $\Delta^{\mathcal{I}_0} = \mathsf{Ind}(\mathcal{A}) \cup \{w_A \mid A \text{ atomic concept appearing in } \mathcal{K}\} \cup \{w_\top\}$
- $A^{\mathcal{I}_0} = \{a \mid A(a) \in \mathcal{A}\} \cup \{w_A\}$
- $R^{\mathcal{I}_0} = \{(a, b) \mid R(a, b) \in \mathcal{A}\}$

We obtain $\mathcal{I}_{i+1}$ from $\mathcal{I}_i$ by applying one of the following rules:

R1 If $e \in (A_1 \sqcap A_2)^{\mathcal{I}_i}$ and $A_1 \sqcap A_2 \sqsubseteq B \in \mathcal{T}$ and $e \notin B^{\mathcal{I}_i}$, then $B^{\mathcal{I}_{i+1}} = B^{\mathcal{I}_i} \cup \{e\}$

R2 If $e \in A^{\mathcal{I}_i}$ and $A \sqsubseteq \exists R.B \in \mathcal{T}$ and $(e, w_B) \notin R^{\mathcal{I}_i}$, then $R^{\mathcal{I}_{i+1}} = R^{\mathcal{I}_i} \cup \{(e, w_B)\}$

R3 If $(d, e) \in R^{\mathcal{I}_i}$ and $e \in A^{\mathcal{I}_i}$ and $\exists R.A \sqsubseteq B \in \mathcal{T}$ and $d \notin B^{\mathcal{I}_i}$, then $B^{\mathcal{I}_{i+1}} = B^{\mathcal{I}_i} \cup \{d\}$

When we have $\mathcal{I}_k$ and no more rules apply, set $\mathcal{C}_\mathcal{K} = \mathcal{I}_k$.

Example: construction of $\mathcal{C}_{\mathcal{K}}$

$\mathcal{K} =$

$R(a, b) \quad C(b)$

$\exists S.B \sqsubseteq B \quad C \sqsubseteq E \quad D \sqsubseteq \exists S.(B \sqcap D)$

$D \sqsubseteq \exists R.(A \sqcap D) \quad \exists R.E \sqsubseteq D$

Example: construction of $\mathcal{C}_{\mathcal{K}}$

$\mathcal{K} =$ $R(a,b) \quad C(b)$

$\exists S.B \sqsubseteq B \quad C \sqsubseteq E \quad D \sqsubseteq \exists S.(B \sqcap D)$
$D \sqsubseteq \exists R.(A \sqcap D) \quad \exists R.E \sqsubseteq D$

TBox normalization:

$$D \sqsubseteq \exists R.(A \sqcap D) \quad \rightsquigarrow \quad D \sqsubseteq \exists R.F \quad F \sqsubseteq A \quad F \sqsubseteq D \quad A \sqcap D \sqsubseteq F$$

$$D \sqsubseteq \exists S.(B \sqcap D) \quad \rightsquigarrow \quad D \sqsubseteq \exists S.G \quad G \sqsubseteq B \quad G \sqsubseteq D \quad B \sqcap D \sqsubseteq G$$

Example: construction of $\mathcal{C}_{\mathcal{K}}$

$\mathcal{K} =$

$R(a,b) \quad C(b)$

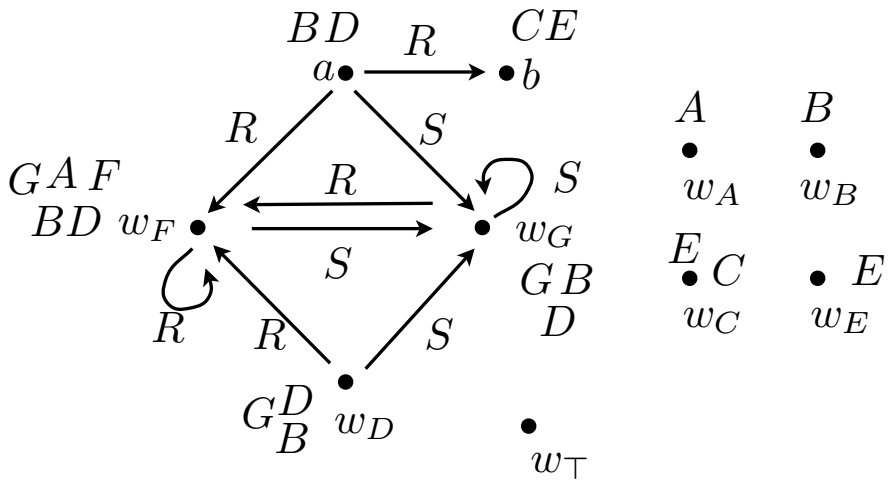
$C \sqsubseteq E \quad D \sqsubseteq \exists S.G \quad G \sqsubseteq B \quad G \sqsubseteq D$
$B \sqcap D \sqsubseteq G \quad D \sqsubseteq \exists R.F \quad A \sqcap D \sqsubseteq F$
$F \sqsubseteq A \quad F \sqsubseteq D \quad \exists R.E \sqsubseteq D \quad \exists S.B \sqsubseteq B$

Example: construction of $\mathcal{C}_{\mathcal{K}}$

$\mathcal{K} =$

$R(a,b) \quad C(b)$

$C \sqsubseteq E \quad D \sqsubseteq \exists S.G \quad G \sqsubseteq B \quad G \sqsubseteq D$
$B \sqcap D \sqsubseteq G \quad D \sqsubseteq \exists R.F \quad A \sqcap D \sqsubseteq F$
$F \sqsubseteq A \quad F \sqsubseteq D \quad \exists R.E \sqsubseteq D \quad \exists S.B \sqsubseteq B$



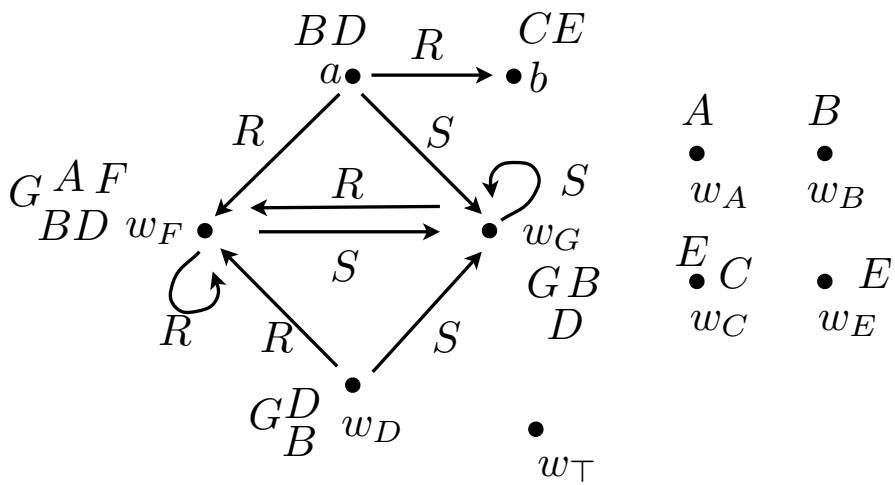
Result: $\mathcal{C}_{\mathcal{K}}$

Example: construction of $\mathcal{C}_{\mathcal{K}}$

$\mathcal{K} =$

$R(a,b) \quad C(b)$

$C \sqsubseteq E \quad D \sqsubseteq \exists S.G \quad G \sqsubseteq B \quad G \sqsubseteq D$
$B \sqcap D \sqsubseteq G \quad D \sqsubseteq \exists R.F \quad A \sqcap D \sqsubseteq F$
$F \sqsubseteq A \quad F \sqsubseteq D \quad \exists R.E \sqsubseteq D \quad \exists S.B \sqsubseteq B$



Properties of $\mathcal{C}_\mathcal{K}$

Some important properties of $\mathcal{C}_\mathcal{K}$:

- $\mathcal{C}_\mathcal{K}$ is a model of $\mathcal{K}$
- $\mathcal{C}_\mathcal{K}$ can be constructed in polynomial time in $|\mathcal{K}|$
- for every ABox assertion α:
  $\mathcal{K} \models \alpha$ iff $\mathcal{C}_\mathcal{K}$ satisfies α
- for every concept inclusion $A \sqsubseteq B$ (with A, B atomic concepts):
  $\mathcal{K} \models A \sqsubseteq B$ iff $\mathcal{C}_\mathcal{K}$ satisfies $B(w_A)$

Remark: $\mathcal{C}_\mathcal{K}$ does not give the right answers to conjunctive queries

Reasoning in $\mathcal{EL}$

To test whether an ABox assertion α is entailed from $\mathcal{K}$:

1. Normalize $\mathcal{T}$ and then construct $\mathcal{C}_\mathcal{K}$.
2. Check whether $\mathcal{C}_\mathcal{K}$ satisfies α.

To perform classification for a TBox $\mathcal{T}$:

1. Normalize $\mathcal{T}$ and then construct $\mathcal{C}_\mathcal{K}$, where $\mathcal{K} = (\mathcal{T}, \emptyset)$.
2. To test whether $\mathcal{T} \models A \sqsubseteq B$, check if $\mathcal{C}_\mathcal{K}$ satisfies $B(w_A)$.

To decide subsumption between general concepts, use new atomic concepts to represent the general concepts, e.g. if C is a complex concept, add $X_C \equiv C$ to $\mathcal{T}$ (where X_C is a fresh atomic concept).

Theorem. Subsumption and instance checking in $\mathcal{EL}$ are in PTIME.

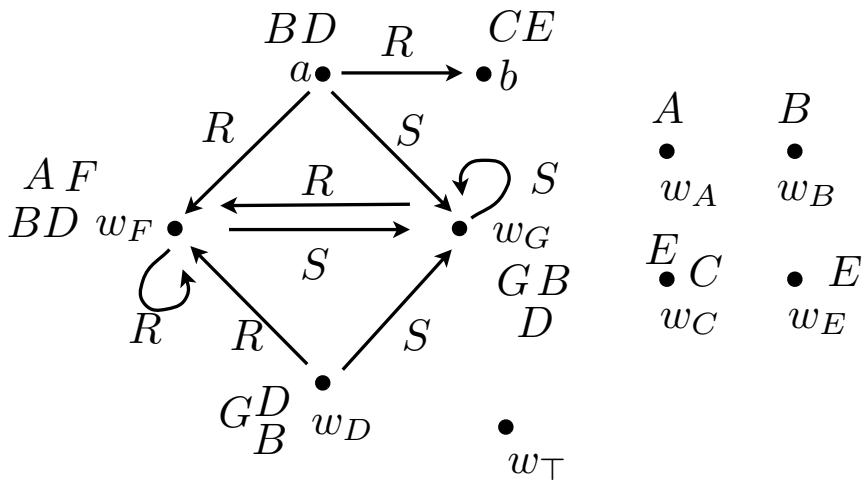
Note that with only $\sqcap$ and $\forall R.C$, these problems are EXPTIME-hard!

Example: using $\mathcal{C}_\mathcal{K}$ for reasoning

$\mathcal{K}$ =

$R(a,b) \quad C(b)$

$C \sqsubseteq E \quad D \sqsubseteq \exists S.G \quad G \sqsubseteq B \quad G \sqsubseteq D$
$B \sqcap D \sqsubseteq G \quad D \sqsubseteq \exists R.F \quad A \sqcap D \sqsubseteq F$
$F \sqsubseteq A \quad F \sqsubseteq D \quad \exists R.E \sqsubseteq D \quad \exists S.B \sqsubseteq B$



Which of the following holds?

$A(a) \quad B(a) \quad S(a,a) \quad D \sqsubseteq F \quad D \sqsubseteq B \quad F \sqsubseteq D$

Proof theory for $\mathcal{EL}$

The following rules can be used to derive inclusions from a normalized TBox $\mathcal{T}$.

$$\frac{}{A \sqsubseteq A} \qquad \frac{}{A \sqsubseteq \top} \qquad \frac{}{\alpha} \text{ with } \alpha \in \mathcal{T}$$

$$\frac{A \sqsubseteq B \qquad B \sqsubseteq C}{A \sqsubseteq C} \qquad \frac{A \sqsubseteq B \qquad A \sqsubseteq C \qquad B \sqcap C \sqsubseteq D}{A \sqsubseteq D}$$

$$\frac{A \sqsubseteq B \qquad B \sqsubseteq \exists R.C}{A \sqsubseteq \exists R.C} \qquad \frac{A \sqsubseteq \exists R.B \qquad B \sqsubseteq C \qquad \exists R.C \sqsubseteq D}{A \sqsubseteq D}$$

Theorem. The above proof theory is sound (i.e. it only yields inclusions which are entailed from $\mathcal{T}$) and complete for classification (i.e. every entailed inclusion $A \sqsubseteq B$, with atomic A, B, can be derived).

Example: proof theory

Consider the TBox $\mathcal{T}$ consisting of the following axioms:

$$A \sqsubseteq B \quad A \sqsubseteq D \quad B \sqsubseteq \exists R.E \quad E \sqsubseteq \exists R.F \quad F \sqsubseteq \exists R.G \quad \exists R.G \sqsubseteq G \quad G \sqcap D \sqsubseteq H$$

We give a proof that $\mathcal{T} \models A \sqsubseteq H$:

$$\frac{F \sqsubseteq \exists R.G \quad G \sqsubseteq G \quad \exists R.G \sqsubseteq G}{F \sqsubseteq G}$$

$$\frac{A \sqsubseteq B \quad B \sqsubseteq \exists R.E}{A \sqsubseteq \exists R.E} \qquad \frac{E \sqsubseteq \exists R.F \quad F \sqsubseteq G \quad \exists R.G \sqsubseteq G}{E \sqsubseteq G}$$

$$\frac{A \sqsubseteq \exists R.E \quad E \sqsubseteq G \quad \exists R.G \sqsubseteq G}{A \sqsubseteq G}$$

$$\frac{A \sqsubseteq G \quad A \sqsubseteq D \quad G \sqcap D \sqsubseteq A}{A \sqsubseteq H}$$

Extensions of $\mathcal{EL}$

We can add all of the following without losing tractability:

- $\bot$
- $\mathsf{dom}(R) \sqsubseteq C$, $\mathsf{range}(R) \sqsubseteq C$
- $R_1 \circ \ldots \circ R_n \sqsubseteq R_{n+1}$

However, just one of the following makes subsumption EXPTIME-hard:

- negation $\neg$
- disjunction $\sqcup$
- at-least or at-most restrictions: $\geq 2R$, $\leq 1R$
- functional roles $(\mathsf{funct}\, R)$
- inverse roles R^-

Intractability of $\mathcal{EL}^{\neg}$

Let $\mathcal{EL}^{\neg}$ be the extension of $\mathcal{EL}$ with $\neg$.

Using $\mathcal{EL}^{\neg}$ concepts, we can simulate $\mathcal{ALC}$ concepts:

- in place of $\bot$, we use $\neg\top$
- in place of $\forall R.C$, we use $\neg\exists R.\neg C$
- in place of $C \sqcup D$, we use $\neg(\neg C \sqcap \neg D)$

Since subsumption is EXPTIME-hard in $\mathcal{ALC}$, the same is true for $\mathcal{EL}^{\neg}$.

We even get EXPTIME-hardness for $\mathcal{EL}^{(\neg)}$, which allows negation only in front of atomic concepts or $\top$:

- replace $\neg C$ by $\neg A_C$ (A_C fresh atomic concept), and then add $A_C \equiv C$ to the TBox

Intractability of $\mathcal{ELU}$

We reduce subsumption in $\mathcal{EL}^{(\neg)}$ to subsumption in $\mathcal{ELU}$ $(= \mathcal{EL} + \sqcup)$.

Let $\mathcal{T}$ be a $\mathcal{EL}^{(\neg)}$ TBox, and let $\mathcal{T}'$ be obtained by:

- introducing a new atomic concept $A_\bot$
- introducing a new atomic concept $\overline{A}$, for every atomic concept A in $\mathcal{T}$
- replacing every occurrence of $\neg A$ in $\mathcal{T}$ by $\overline{A}$
- adding $\top \sqsubseteq A \sqcup \overline{A}$, $A \sqcap \overline{A} \sqsubseteq A_\bot$, and $A_\bot \sqsubseteq A$ for every atomic A in $\mathcal{T}$
- adding $\exists R.A_\bot \sqsubseteq A_\bot$ for every atomic role R in $\mathcal{T}$

It can be shown that

$$\mathcal{T} \models A \sqsubseteq B \quad \textit{iff} \quad \mathcal{T}' \models A \sqsubseteq B$$

for every pair of atomic concepts A, B in $\mathcal{T}$.

It follows that subsumption in $\mathcal{ELU}$ is EXPTIME-hard.

3. *DL-Lite*

The *DL-Lite* family

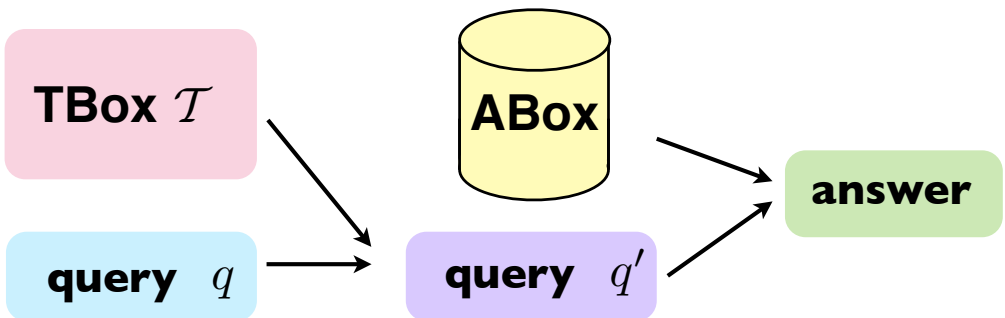
Objective:

useful ontology language allowing efficient conjunctive query answering

Idea: exploit the efficiency of relational DB systems

General approach: query rewriting

- ABox is stored as a traditional database
- the input query is rewritten to integrate the relevant information from the TBox
- the new query is evaluated over the database



Syntax of *DL-Lite*

We present the dialect *DL-Lite*$_{\mathcal{R}}$ (which underlies OWL2 QL).

Assertions: $A(c)$, $R(c, d)$

Inclusions: $B_1 \sqsubseteq B_2$, $B_1 \sqsubseteq \neg B_2$, $S_1 \sqsubseteq S_2$, $S_1 \sqsubseteq \neg S_2$ where

$$B := \top \mid A \mid \exists S \qquad\qquad S := R \mid R^{-}$$

with A an atomic concept and R an atomic role

Other *DL-Lite* dialects allow:

- functional roles (funct S)
- cardinality restrictions ($\geq q\, S$, $\leq q\, S$)
- Horn inclusions ($B_1 \sqcap \ldots \sqcap B_n \sqsubseteq (\neg) B_{n+1}$)
- roles which are symmetric, asymmetric, reflexive, or anti-reflexive

Conjunctive queries

Conjunctive queries are an important subclass of first-order logic queries.

They correspond to select-project-join queries in relational DBs .

Formally: a conjunctive query (CQ) has the form

$$q(x_1, \ldots, x_k) \quad = \quad \exists x_{k+1}, \ldots, x_m \ \alpha_1 \wedge \ldots \wedge \alpha_r$$

where $\alpha_1, \ldots, \alpha_r$ are assertions involving individuals or variables from $x_1, \ldots, x_m$. We call $x_1, \ldots, x_k$ the answer variables and $x_{k+1}, \ldots, x_m$ the quantified variables.

Boolean CQs are CQs which have no answer variables.

Semantics of conjunctive queries

Satisfaction in an interpretation:

- $\mathcal{I}$ satisfies a Boolean CQ $\exists x_1, \ldots, x_n\ \alpha_1 \wedge \ldots \wedge \alpha_m$ if there exists a function $\pi : \{x_1, \ldots, x_n\} \to \Delta^{\mathcal{I}}$ such that $\mathcal{I}$ satisfies each α'_i, where α_i is obtained by replacing any occurrence of x_j by $\pi(x_j)$

Entailment of a Boolean query:

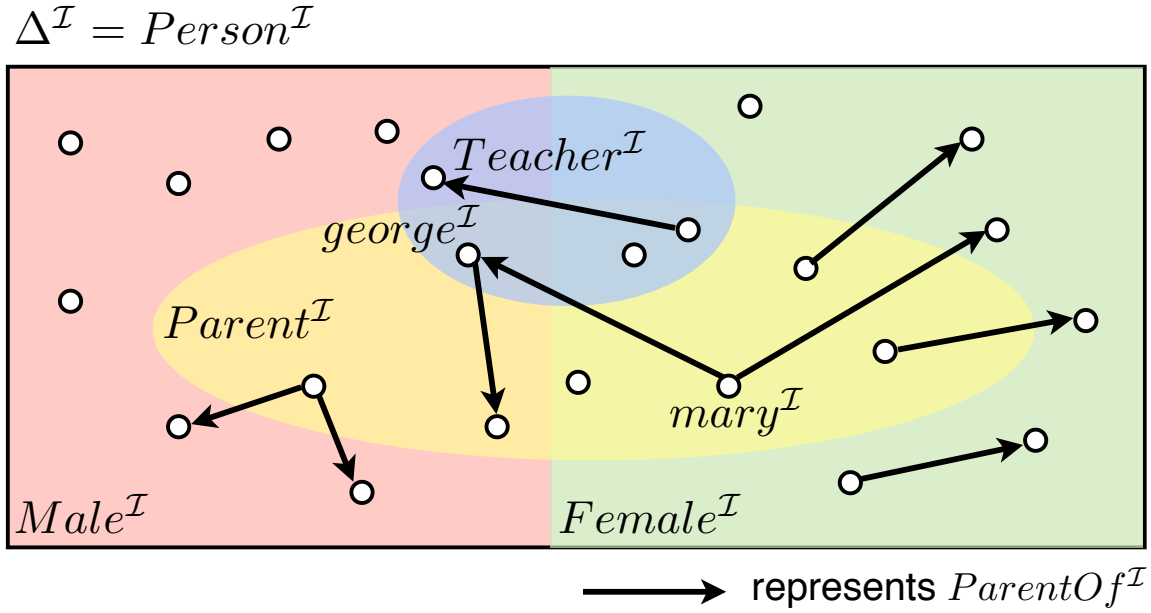
- A Boolean CQ q is entailed from $\mathcal{K}$ (written $\mathcal{K} \models q$) iff every model of $\mathcal{K}$ satisfies q

Certain answers to a query:

- a tuple $(a_1, \ldots, a_k)$ of individuals is a (certain) answer to $q(x_1, \ldots, x_k)$ w.r.t. $\mathcal{K}$ iff $\mathcal{K} \models q[a_1, \ldots, a_k]$
  (where $q[a_1, \ldots, a_k]$ is q with every x_i replaced by a_i)
- we denote by $\mathrm{cert}(q, \mathcal{K})$ the certain answers to q w.r.t. $\mathcal{K}$

Example: Satisfaction in an interpretation

Reconsider the interpretation $\mathcal{I}$:



Which of the following CQs are satisfied in $\mathcal{I}$?

- $\exists x\, \mathit{ParentOf}(x, \mathit{george}) \wedge \mathit{Female}(x)$
- $\exists x\, \mathit{ParentOf}(\mathit{george}, x) \wedge \mathit{Female}(x)$
- $\exists x\, \mathit{ParentOf}(\mathit{mary}, x) \wedge \mathit{Female}(x)$
- $\mathit{ParentOf}(\mathit{mary}, \mathit{george})$
- $\exists x, y\, \mathit{Male}(x) \wedge \mathit{ParentOf}(x, y) \wedge \mathit{Female}(y)$
- $\exists x\, \mathit{ParentOf}(x, x)$
- $\exists x, y\, \mathit{Teacher}(x) \wedge \mathit{ParentOf}(x, y) \wedge \mathit{Teacher}(y)$
- $\exists x\, \mathit{Teacher}(x) \wedge \mathit{Female}(x)$

Example: Certain answers

Consider the DL-Lite KB $\mathcal{K}$ consisting of the TBox:

$\mathit{Parent} \sqsubseteq \exists \mathit{ParentOf}$ $\quad \exists \mathit{ParentOf} \sqsubseteq \mathit{Parent}$ $\quad \mathit{Mother} \sqsubseteq \mathit{Parent}$ $\quad \mathit{Father} \sqsubseteq \mathit{Parent}$

$\mathit{Mother} \sqsubseteq \mathit{Female}$ $\quad \mathit{Father} \sqsubseteq \mathit{Male}$ $\quad \mathit{ParentOf} \sqsubseteq \mathit{ChildOf}^-$ $\quad \mathit{ChildOf}^- \sqsubseteq \mathit{ParentOf}$

and the ABox:

$\mathit{Mother}(\mathit{mary})$ $\quad \mathit{Father}(\mathit{paul})$ $\quad \mathit{ParentOf}(\mathit{julie}, \mathit{marc})$ $\quad \mathit{ParentOf}(\mathit{marc}, \mathit{paul})$

Determine the certain answers to the following queries.

- $\mathit{Female}(x)$
- $\mathit{ChildOf}(x, y)$
- $\exists y\, \mathit{ParentOf}(x, y)$
- $\mathit{ParentOf}(x, y) \wedge \mathit{ChildOf}(y, z)$

First-order queries

Atomic formulas have the forms $A(t)$, $R(t, t')$ or $t = t'$, where A is an atomic concept, R is an atomic role, and t, t' are terms (either individuals or variables).

First-order queries are constructed from atomic formulas using:

- conjunction $\wedge$
- disjunction $\vee$
- negation $\neg$
- universal quantification $\forall x \varphi$ (with x a variable)
- existential quantification $\exists x \varphi$ (with x a variable)

We write $\mathcal{I} \models q$ if the interpretation $\mathcal{I}$ satisfies Boolean query q (under usual first-order semantics).

Query rewriting

Given an ABox $\mathcal{A}$, we define the interpretation $\mathcal{I}_\mathcal{A}$ as follows:

- $\Delta^{\mathcal{I}_\mathcal{A}}$ contains the individuals in $\mathcal{A}$
- $A^{\mathcal{I}_\mathcal{A}} = \{a \mid A(a) \in \mathcal{A}\}$
- $R^{\mathcal{I}_\mathcal{A}} = \{(a, b) \mid R(a, b) \in \mathcal{A}\}$

A first-order query q' is called a perfect rewriting of a CQ q w.r.t. a TBox $\mathcal{T}$ if and only if we have

$$\vec{a} \in \mathsf{cert}(q, \mathcal{K}) \quad \Leftrightarrow \quad \mathcal{I}_\mathcal{A} \models q[\vec{a}]$$

for every ABox $\mathcal{A}$.

Existence of perfect rewritings:

- In DL-Lite, a perfect rewriting exists for every CQ and TBox.
- In $\mathcal{EL}$, a perfect rewriting need not exist.

Example

$\mathcal{T}$

$$\text{Professeur} \sqsubseteq \text{Enseignant} \qquad \text{Enseignant} \sqsubseteq \exists \text{enseigne}$$
$$\text{Mcf} \sqsubseteq \text{Enseignant} \qquad \exists \text{enseigne} \sqsubseteq \text{Enseignant}$$
$$\text{Mcf} \sqsubseteq \neg \text{Professeur} \qquad \exists \text{enseigne}^{-} \sqsubseteq \text{Cours}$$
$$\text{Professeur} \sqsubseteq \text{HDR} \qquad \exists \, \text{directeur} \sqsubseteq \text{Professeur}$$

$\mathcal{A}$

$$\text{Professeur}(\text{Marie}) \qquad \text{enseigne}(\text{Paul}, \text{INF100})$$
$$\text{Mcf}(\text{Jean}) \qquad \text{HDR}(\text{Paul}) \qquad \text{directeur}(\text{Marc}, \text{deptInfo})$$

Example

$\mathcal{T}$

$$\text{Professeur} \sqsubseteq \text{Enseignant} \qquad \text{Enseignant} \sqsubseteq \exists \text{enseigne}$$
$$\text{Mcf} \sqsubseteq \text{Enseignant} \qquad \exists \text{enseigne} \sqsubseteq \text{Enseignant}$$
$$\text{Mcf} \sqsubseteq \neg \text{Professeur} \qquad \exists \text{enseigne}^{-} \sqsubseteq \text{Cours}$$
$$\text{Professeur} \sqsubseteq \text{HDR} \qquad \exists \, \text{directeur} \sqsubseteq \text{Professeur}$$

$\mathcal{A}$

$$\text{Professeur}(\text{Marie}) \qquad \text{enseigne}(\text{Paul}, \text{INF100})$$
$$\text{Mcf}(\text{Jean}) \qquad \text{HDR}(\text{Paul}) \qquad \text{directeur}(\text{Marc}, \text{deptInfo})$$

q

$$\text{Enseignant}(x) \wedge \text{HDR}(x)$$

Example

$\mathcal{T}$

$$\text{Professeur} \sqsubseteq \text{Enseignant} \qquad \text{Enseignant} \sqsubseteq \exists \text{enseigne}$$
$$\text{Mcf} \sqsubseteq \text{Enseignant} \qquad \exists \text{enseigne} \sqsubseteq \text{Enseignant}$$
$$\text{Mcf} \sqsubseteq \neg \text{Professeur} \qquad \exists \text{enseigne}^{-} \sqsubseteq \text{Cours}$$
$$\text{Professeur} \sqsubseteq \text{HDR} \qquad \exists \, \text{directeur} \sqsubseteq \text{Professeur}$$

$\mathcal{A}$

$$\text{Professeur}(\text{Marie}) \qquad \text{enseigne}(\text{Paul}, \text{INF100})$$
$$\text{Mcf}(\text{Jean}) \qquad \text{HDR}(\text{Paul}) \qquad \text{directeur}(\text{Marc}, \text{deptInfo})$$

q

$$\text{Enseignant}(x) \wedge \text{HDR}(x)$$

q'

$$(\text{Enseignant}(x) \wedge \text{HDR}(x))$$

Example

$\mathcal{T}$

$\boxed{\text{Professeur} \sqsubseteq \text{Enseignant}}$ $\quad \text{Enseignant} \sqsubseteq \exists \text{enseigne}$

$\text{Mcf} \sqsubseteq \text{Enseignant} \quad \exists \text{enseigne} \sqsubseteq \text{Enseignant}$

$\text{Mcf} \sqsubseteq \neg \text{Professeur} \quad \exists \text{enseigne}^{-} \sqsubseteq \text{Cours}$

$\boxed{\text{Professeur} \sqsubseteq \text{HDR}} \quad \exists \, \text{directeur} \sqsubseteq \text{Professeur}$

$\mathcal{A}$

$\text{Professeur}(\text{Marie}) \quad \text{enseigne}(\text{Paul}, \text{INF100})$

$\text{Mcf}(\text{Jean}) \quad \text{HDR}(\text{Paul}) \quad \text{directeur}(\text{Marc}, \text{deptInfo})$

q

$\text{Enseignant}(x) \wedge \text{HDR}(x)$

q'

$(\text{Enseignant}(x) \wedge \text{HDR}(x)) \boxed{\vee \text{Professeur}(x)}$

Example

$\mathcal{T}$

$\mathsf{Professeur} \sqsubseteq \mathsf{Enseignant}$ $\quad$ $\mathsf{Enseignant} \sqsubseteq \exists \mathsf{enseigne}$

$\boxed{\mathsf{Mcf} \sqsubseteq \mathsf{Enseignant}}$ $\quad$ $\exists \mathsf{enseigne} \sqsubseteq \mathsf{Enseignant}$

$\mathsf{Mcf} \sqsubseteq \neg \mathsf{Professeur}$ $\quad$ $\exists \mathsf{enseigne}^{-} \sqsubseteq \mathsf{Cours}$

$\mathsf{Professeur} \sqsubseteq \mathsf{HDR}$ $\quad$ $\exists\, \mathsf{directeur} \sqsubseteq \mathsf{Professeur}$

$\mathcal{A}$

$\mathsf{Professeur}(\mathsf{Marie})$ $\quad$ $\mathsf{enseigne}(\mathsf{Paul}, \mathsf{INF100})$

$\mathsf{Mcf}(\mathsf{Jean})$ $\quad$ $\mathsf{HDR}(\mathsf{Paul})$ $\quad$ $\mathsf{directeur}(\mathsf{Marc}, \mathsf{deptInfo})$

q

$\mathsf{Enseignant}(x) \wedge \mathsf{HDR}(x)$

q'

$(\mathsf{Enseignant}(x) \wedge \mathsf{HDR}(x)) \vee \mathsf{Professeur}(x) \vee \boxed{(\mathsf{Mcf}(x) \wedge \mathsf{HDR}(x))}$

Example

$\mathcal{T}$

$\mathsf{Professeur} \sqsubseteq \mathsf{Enseignant}$ $\quad$ $\mathsf{Enseignant} \sqsubseteq \exists \mathsf{enseigne}$

$\mathsf{Mcf} \sqsubseteq \mathsf{Enseignant}$ $\quad$ $\boxed{\exists \mathsf{enseigne} \sqsubseteq \mathsf{Enseignant}}$

$\mathsf{Mcf} \sqsubseteq \neg \mathsf{Professeur}$ $\quad$ $\exists \mathsf{enseigne}^{-} \sqsubseteq \mathsf{Cours}$

$\mathsf{Professeur} \sqsubseteq \mathsf{HDR}$ $\quad$ $\exists\, \mathsf{directeur} \sqsubseteq \mathsf{Professeur}$

$\mathcal{A}$

$\mathsf{Professeur}(\mathsf{Marie})$ $\quad$ $\mathsf{enseigne}(\mathsf{Paul}, \mathsf{INF100})$

$\mathsf{Mcf}(\mathsf{Jean})$ $\quad$ $\mathsf{HDR}(\mathsf{Paul})$ $\quad$ $\mathsf{directeur}(\mathsf{Marc}, \mathsf{deptInfo})$

q

$\mathsf{Enseignant}(x) \wedge \mathsf{HDR}(x)$

q'

$(\mathsf{Enseignant}(x) \wedge \mathsf{HDR}(x)) \vee \mathsf{Professeur}(x) \vee (\mathsf{Mcf}(x) \wedge \mathsf{HDR}(x))$

$\boxed{\vee(\exists y.\mathsf{enseigne}(x, y) \wedge \mathsf{HDR}(x))}$

Example

$\mathcal{T}$

$\text{Professeur} \sqsubseteq \text{Enseignant}$ $\quad \text{Enseignant} \sqsubseteq \exists \text{enseigne}$

$\text{Mcf} \sqsubseteq \text{Enseignant}$ $\quad \exists \text{enseigne} \sqsubseteq \text{Enseignant}$

$\text{Mcf} \sqsubseteq \neg \text{Professeur}$ $\quad \exists \text{enseigne}^{-} \sqsubseteq \text{Cours}$

$\text{Professeur} \sqsubseteq \text{HDR}$ $\quad \boxed{\exists \, \text{directeur} \sqsubseteq \text{Professeur}}$

$\mathcal{A}$

$\text{Professeur}(\text{Marie})$ $\quad \text{enseigne}(\text{Paul}, \text{INF100})$

$\text{Mcf}(\text{Jean})$ $\quad \text{HDR}(\text{Paul})$ $\quad \text{directeur}(\text{Marc}, \text{deptInfo})$

q

$\text{Enseignant}(x) \wedge \text{HDR}(x)$

q'

$(\text{Enseignant}(x) \wedge \text{HDR}(x)) \vee \boxed{\text{Professeur}(x)} \vee (\text{Mcf}(x) \wedge \text{HDR}(x))$
$\vee (\exists y.\text{enseigne}(x, y) \wedge \text{HDR}(x)) \boxed{\vee (\exists y.\text{directeur}(x, y))}$

Example

$\mathcal{T}$

$\mathsf{Professeur} \sqsubseteq \mathsf{Enseignant}$ $\quad$ $\mathsf{Enseignant} \sqsubseteq \exists \mathsf{enseigne}$

$\mathsf{Mcf} \sqsubseteq \mathsf{Enseignant}$ $\quad$ $\exists \mathsf{enseigne} \sqsubseteq \mathsf{Enseignant}$

$\mathsf{Mcf} \sqsubseteq \neg \mathsf{Professeur}$ $\quad$ $\exists \mathsf{enseigne}^{-} \sqsubseteq \mathsf{Cours}$

$\mathsf{Professeur} \sqsubseteq \mathsf{HDR}$ $\quad$ $\exists\, \mathsf{directeur} \sqsubseteq \mathsf{Professeur}$

$\mathcal{A}$

$\mathsf{Professeur}(\mathsf{Marie})$ $\quad$ $\mathsf{enseigne}(\mathsf{Paul}, \mathsf{INF100})$

$\mathsf{Mcf}(\mathsf{Jean})$ $\quad$ $\mathsf{HDR}(\mathsf{Paul})$ $\quad$ $\mathsf{directeur}(\mathsf{Marc}, \mathsf{deptInfo})$

q

$\mathsf{Enseignant}(x) \wedge \mathsf{HDR}(x)$

q'

$(\mathsf{Enseignant}(x) \wedge \mathsf{HDR}(x)) \vee \mathsf{Professeur}(x) \vee (\mathsf{Mcf}(x) \wedge \mathsf{HDR}(x))$
$\vee (\exists y.\mathsf{enseigne}(x, y) \wedge \mathsf{HDR}(x)) \vee (\exists y.\mathsf{directeur}(x, y))$

Example

$\mathcal{T}$

$$\text{Professeur} \sqsubseteq \text{Enseignant} \qquad \text{Enseignant} \sqsubseteq \exists \text{enseigne}$$
$$\text{Mcf} \sqsubseteq \text{Enseignant} \qquad \exists \text{enseigne} \sqsubseteq \text{Enseignant}$$
$$\text{Mcf} \sqsubseteq \neg \text{Professeur} \qquad \exists \text{enseigne}^{-} \sqsubseteq \text{Cours}$$
$$\text{Professeur} \sqsubseteq \text{HDR} \qquad \exists \, \text{directeur} \sqsubseteq \text{Professeur}$$

$\mathcal{A}$

$$\boxed{\text{Professeur}(\text{Marie})} \qquad \boxed{\text{enseigne}(\text{Paul}, \text{INF100})}$$
$$\text{Mcf}(\text{Jean}) \qquad \boxed{\text{HDR}(\text{Paul})} \qquad \boxed{\text{directeur}(\text{Marc}, \text{deptInfo})}$$

q

$$\text{Enseignant}(x) \wedge \text{HDR}(x)$$

$x = \text{Marie}$

q'

$$(\text{Enseignant}(x) \wedge \text{HDR}(x)) \vee \boxed{\text{Professeur}(x)} \vee (\text{Mcf}(x) \wedge \text{HDR}(x))$$
$$\vee \boxed{(\exists y.\text{enseigne}(x, y) \wedge \text{HDR}(x))} \vee \boxed{(\exists y.\text{directeur}(x, y))}$$

$x = \text{Paul}, y = \text{INF100}$ $\qquad$ $x = \text{Marc}, y = \text{deptInfo}$

Answers: Marie, Paul, Marc

Rewriting algorithm: atoms

Let I be an inclusion that is applicable to atom α. Then $\mathsf{ra}(\alpha, I)$ is defined as follows:

- if $\alpha = A(x)$ and $I = B \sqsubseteq A$, then $\mathsf{ra}(\alpha, I) = B(x)$
- if $\alpha = A(x)$ and $I = \exists R \sqsubseteq A$, then $\mathsf{ra}(\alpha, I) = R(x, \_)$
- if $\alpha = A(x)$ and $I = \exists R^- \sqsubseteq A$, then $\mathsf{ra}(\alpha, I) = R(\_, x)$
- if $\alpha = R(x, \_)$ and $I = A \sqsubseteq \exists R$, then $\mathsf{ra}(\alpha, I) = A(x)$
- if $\alpha = R(x, \_)$ and $I = \exists S \sqsubseteq \exists R$, then $\mathsf{ra}(\alpha, I) = S(x, \_)$
- if $\alpha = R(x, \_)$ and $I = \exists S^- \sqsubseteq \exists R$, then $\mathsf{ra}(\alpha, I) = S(\_, x)$
- if $\alpha = R(\_, x)$ and $I = A \sqsubseteq \exists R^-$, then $\mathsf{ra}(\alpha, I) = A(x)$
- if $\alpha = R(\_, x)$ and $I = \exists S \sqsubseteq \exists R^-$, then $\mathsf{ra}(\alpha, I) = S(x, \_)$
- if $\alpha = R(\_, x)$ and $I = \exists S^- \sqsubseteq \exists R^-$, then $\mathsf{ra}(\alpha, I) = S(\_, x)$
- if $\alpha = R(x, y)$ and $I = S \sqsubseteq R$ or $I = S^- \sqsubseteq R^-$, then $\mathsf{ra}(\alpha, I) = S(x, y)$
- if $\alpha = R(x, y)$ and $I = S \sqsubseteq R^-$ or $I = S^- \sqsubseteq R$, then $\mathsf{ra}(\alpha, I) = S(x, y)$

Rewriting algorithm

Algorithm PerfectRef$(q, \mathcal{T})$
Input: conjunctive query q, TBox $\mathcal{T}$
Output: a query PR which is a perfect rewriting of q given $\mathcal{T}$

$PR := \{q\}$
repeat until $PR' = PR$
  $PR' := PR$
  for each $\alpha \in PR'$ **do**
    for each $\alpha \in q$ **do**
      if I is applicable to α
        $PR := PR \cup \{q[\alpha/\mathsf{ra}(\alpha, I)]\}$
    for each $\alpha, \beta \in q$ **do**
      if α and β unify
        $PR := PR \cup \{\tau(\mathit{reduce}(q, \alpha, \beta))\}$
return PR

τ: replaces unbound variables by '_'
reduce: replaces α and β by most general unifier

Satisfiability

Satisfiability in $DL\text{-}Lite_{\mathcal{R}}$ can also be reduced to database querying.

1. Compute the set NI of all entailed negative inclusions (i.e. inclusions of the form $B \sqsubseteq \neg B'$ or $R \sqsubseteq \neg R'$).
2. For each inclusion φ in NI, let $unsat(\varphi)$ be the first-order query which describes when φ is not satisfied. For example:
   - $\varphi = A \sqsubseteq \neg D$,
   - $\varphi = \exists R \sqsubseteq \neg \exists S^{-}$,

Satisfiability

Satisfiability in *DL-Lite*$_{\mathcal{R}}$ can also be reduced to database querying.

1. Compute the set *NI* of all entailed negative inclusions (i.e. inclusions of the form $B \sqsubseteq \neg B'$ or $R \sqsubseteq \neg R'$).
2. For each inclusion φ in *NI*, let $\mathit{unsat}(\varphi)$ be the first-order query which describes when φ is not satisfied. For example:
   - $\varphi = A \sqsubseteq \neg D$, then $\mathit{unsat}(\varphi) = \exists x\ A(x) \wedge D(x)$
   - $\varphi = \exists R \sqsubseteq \neg \exists S^{-}$, then $\mathit{unsat}(\varphi) = \exists x, y, z\ R(x, y) \wedge S(z, x)$

Satisfiability

Satisfiability in *DL-Lite*$_{\mathcal{R}}$ can also be reduced to database querying.

1. Compute the set *NI* of all entailed negative inclusions (i.e. inclusions of the form $B \sqsubseteq \neg B'$ or $R \sqsubseteq \neg R'$).
2. For each inclusion φ in *NI*, let *unsat*(φ) be the first-order query which describes when φ is not satisfied. For example:
   - $\varphi = A \sqsubseteq \neg D$, then $\mathit{unsat}(\varphi) = \exists x\ A(x) \wedge D(x)$
   - $\varphi = \exists R \sqsubseteq \neg \exists S^{-}$, then $\mathit{unsat}(\varphi) = \exists x, y, z\ R(x, y) \wedge S(z, x)$
3. Evaluate the following union of conjunctive queries

$$\bigvee_{\varphi \in NI} \mathit{unsat}(\varphi)$$

over $\mathcal{I}_{\mathcal{A}}$.

Complexity of *DL-Lite*

Two complexity measures:

- combined complexity: in terms of the size of the TBox and ABox
- data complexity: only in terms of the size of the ABox
  - appropriate when $|\mathcal{A}| >> |\mathcal{T}|$

For *DL-Lite*$_{\mathcal{R}}$, satisfiability and conjunctive query answering are:

- in AC^0 ($\subsetneq$ *LOGSPACE* $\subseteq$ *PTIME*) for data complexity
- *NLOGSPACE*-complete for combined complexity

Same low data complexity as querying relational databases

4. Distributed reasoning in *DL-Lite*

Distributed *DL-Lite*$_\mathcal{R}$ setting

Each peer $\mathcal{P}_i$ in the system has:

- its own vocabulary (set of concepts and roles)
  - use index i to identify $\mathcal{P}_i$'s relations, e.g. A_i, r_i
- its own *DL-Lite*$_\mathcal{R}$ TBox $\mathcal{T}_i$, in its vocabulary
- its own *DL-Lite*$_\mathcal{R}$ ABox $\mathcal{A}_i$, in its vocabulary
- mappings = *DL-Lite*$_\mathcal{R}$ inclusions using vocabulary of different peers
  - a mapping between peers $\mathcal{P}_i$ and $\mathcal{P}_i$ belongs to both $\mathcal{T}_i$ and $\mathcal{T}_j$

Goal: use the algorithm DeCA to perform distributed query rewriting

Propositional encoding of a TBox

We will only give the translation for positive inclusions, since these are what is needed for rewriting queries.

Concepts and roles

- $\text{prop}(A) = A$ and $prop(R) = R$
- $\text{prop}(R^-) = R^-$
- $\text{prop}(\exists R) = R^{\exists}$ and $\text{prop}(\exists R^-) = R^{\exists -}$

Inclusions

- $\text{prop}(B \sqsubseteq C) = \text{prop}(B) \rightarrow \text{prop}(C)$
- $\text{prop}(R \sqsubseteq S) = \{R \rightarrow S, R^- \rightarrow S^-, R^{\exists} \rightarrow S^{\exists}, R^{\exists -} \rightarrow S^{\exists -}\}$
- $\text{prop}(R^- \sqsubseteq S) = \{R^- \rightarrow S, R \rightarrow S^-, R^{\exists -} \rightarrow S^{\exists}, R^{\exists} \rightarrow S^{\exists -}\}$
- $\text{prop}(R \sqsubseteq S^-) = \{R \rightarrow S,^- R^- \rightarrow S, R^{\exists} \rightarrow S^{\exists -}, R^{\exists -} \rightarrow S^{\exists}\}$
- $\text{prop}(R^- \sqsubseteq S^-) = \{R \rightarrow S, R^- \rightarrow S^-, R^{\exists} \rightarrow S^{\exists}, R^{\exists -} \rightarrow S^{\exists -}\}$

Computing rewritings of an atom

Let $AR(\alpha, PI)$ be the set of atoms β such that there exists $I_1, \ldots, I_n \in PI$ and $\alpha = \gamma_0, \gamma_1, \ldots, \gamma_n = \beta$ such that for each $1 \leq i \leq n$, $\gamma_i = \mathsf{ra}(\gamma_{i-1}, I_i)$.

Theorem. Let PI be a set of positive inclusions. Then $\beta \in AR(\alpha, PI)$ if and only if $\mathsf{prop}(PI) \cup \{\neg V_1\} \models \neg V_2$ where:

- $\alpha = A(x)$, $\beta = B(x)$, $V_1 = A$, and $V_2 = B$
- $\alpha = A(x)$, $\beta = R(x, \_)$, $V_1 = A$, and $V_2 = R^{\exists}$
- $\alpha = A(x)$, $\beta = R(\_, x)$, $V_1 = A$, and $V_2 = R^{\exists -}$
- $\alpha = R(x, y)$, $\beta = s(x, y)$, $V_1 = R$, and $V_2 = S$
- $\alpha = R(x, y)$, $\beta = s(y, x)$, $V_1 = R$, and $V_2 = S^{-}$
- $\alpha = R(x, \_)$, $\beta = A(x)$, $V_1 = R^{\exists}$, and $V_2 = A$
- $\alpha = R(x, \_)$, $\beta = S(x, \_)$, $V_1 = R^{\exists}$, and $V_2 = S^{\exists}$
- $\alpha = R(x, \_)$, $\beta = S(\_, x)$, $V_1 = R^{\exists}$, and $V_2 = S^{\exists -}$
- ...

Rewriting atoms using DeCA: if α uses the vocabulary of $\mathcal{P}_i$, then $\beta \in AR(\alpha, PI)$ if and only if $\neg V_2 \in \mathsf{DeCA}^i(\neg V_1)$

Decentralized query rewriting algorithm

Algorithm PerfectRef$^i(q, \mathcal{T})$
Input: conjunctive query q using $\mathcal{P}_i$'s vocabulary, TBox $\mathcal{T}_i$
Output: union of conjunctive queries PR over the global vocabulary

$PR := \{q\}$
$PR' := PR$
while $PR' \neq \emptyset$
  (a) **for each** $q' = \alpha_1 \wedge \ldots \wedge \alpha_n \in PR'$ **do**
    $PR'' = \textcircled{\wedge}_{j=1}^{n} AR(\alpha_j, PI)$ // compute using DeCA^i
    $PR' = \emptyset$
  (b) **for each** $q'' \in PR''$, and each pair $\alpha, \beta \in q''$ **do**
    if α and β unify
      $PR' := PR' \cup \{\tau(reduce(q'', \alpha, \beta))\}$
  $PR = PR \cup PR' \cup PR''$
return PR

5. Current challenges

Current challenges

1. Improve performance
   - *DL-Lite*: rewritten queries can be huge!
     - lots of work on alternative query rewriting algorithms
     - combined rewriting approach (saturation+rewriting)
2. Extension to more expressive data-tractable languages
   - Horn-$\mathcal{SHIQ}$: PTIME (data complexity), EXPTIME (combined), extends both $\mathcal{EL}$ and DL-Lite
3. Make ontologies easier to build, modify, and understand
   - modularity, forgetting, ... (see next course!)
   - debugging, explanation, ...

6. Bibliography

References: Intro to DLs

Comprehensive textbook

The Description Logic Handbook: Theory, Implementation and Applications. Edited by Franz Baader, Diego Calvanese, Deborah McGuinness, Daniele Nardi, and Peter Patel-Schneider.
Cambridge University Press (2003).

For more recent view on the field, proceedings of yearly DL workshop:

http://dl.kr.org/workshops/

References: EL family

- Franz Baader, Sebastian Brandt, Carsten Lutz. *Pushing the EL Envelope*. Proceedings of IJCAI (2005).
- Carsten Lutz, David Toman, Frank Wolter. *Conjunctive Query Answering in the Description Logic EL Using a Relational Database System*. Proceedings of IJCAI (2009).

References: DL-Lite family

- Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, Riccardo Rosati. *Tractable Reasoning and Efficient Query Answering in Description Logics: The DL-Lite Family*. Journal of Automated Reasoning 39(3): 385-429 (2007).
- Alessandro Artale, Diego Calvanese, Roman Kontchakov, Michael Zakharyaschev. *The DL-Lite Family and Relations*. Journal of Artificial Intelligence Research 36: 1-69 (2009).
- Roman Kontchakov, Carsten Lutz, David Toman, Frank Wolter, Michael Zakharyaschev. *The Combined Approach to Query Answering in DL-Lite*. Proceedings of KR (2010).

References: Distributed DLs

- Alexander Borgida, Luciano Serafini. *Distributed Description Logics: Assimilating Information from Peer Sources*. Journal of Data Semantics (2003).
- Luciano Serafini, Alexander Borgida, Andrei Tamilin. *Aspects of Distributed and Modular Ontology Reasoning*. Proceedings of IJCAI (2005).
- Nada Abdallah, François Goasdoué, Marie-Christine Rousset. *DL-LITE$_R$ in the Light of Propositional Logic for Decentralized Data Management*. Proceedings of IJCAI (2009).