Introduction to Description Logic
and Query Rewriting

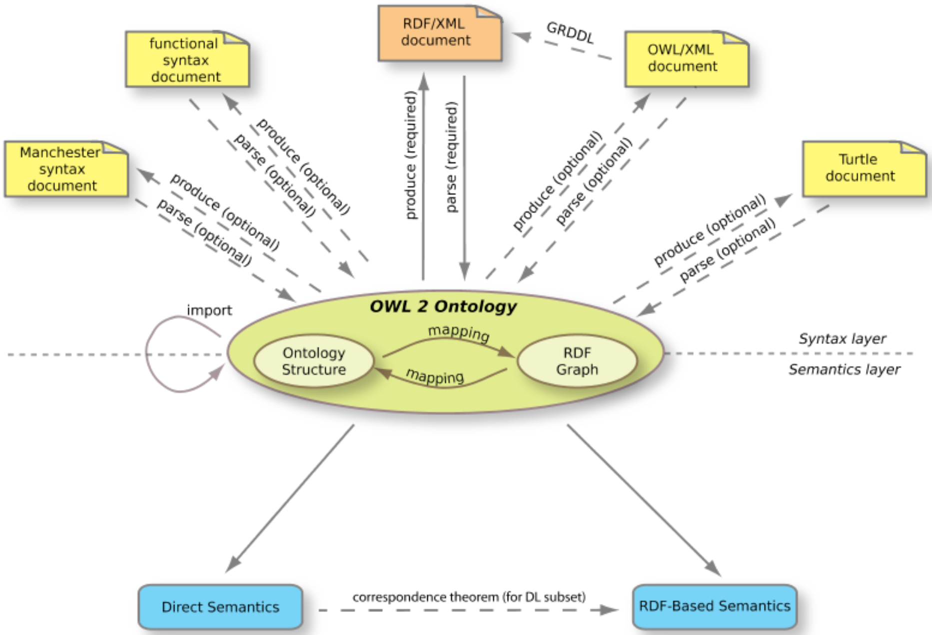
Roman Kontchakov

Department of Computer Science and Inf. Systems, Birkbeck College, London

http://www.dcs.bbk.ac.uk/~roman

Alessandro Artale, Diego Calvanese, Carsten Lutz, Mariano Rodríguez-Muro, David Toman, Frank Wolter and Michael Zakharyaschev

The Web Ontology Language



Part 1

DL Basics

Description Logics by Example

SubClassOf(**ObjectIntersectionOf**(Person,
ObjectSomeValuesFrom(takesCourse, Course)), Student)

$$\text{Person} \sqcap \exists \text{takesCourse}.\text{Course} \sqsubseteq \text{Student}$$

SubObjectPropertyOf(mastersDegreeFrom, degreeFrom)

$$\text{mastersDegreeFrom} \sqsubseteq \text{degreeFrom}$$

SubClassOf(**ObjectSomeValuesFrom**(
ObjectInverseOf(takesCourse), **owl:Thing**), Course)

$$\text{takesCourse}^{-}.\top \sqsubseteq \text{Course}$$

ClassAssertion(Student, john)

$$\text{Student}(\text{john})$$

ObjectPropertyAssertion(takesCourse, john, sw)

$$\text{takesCourse}(\text{john}, \text{sw})$$

DL Syntax

concepts (classes)

$$C \quad ::= \quad \underbrace{A_i}_{\text{concept name}} \quad | \quad \underbrace{\top}_{\text{owl:Thing}} \quad | \quad \underbrace{\bot}_{\text{owl:Nothing}} \quad |$$

$$\underbrace{\neg C}_{\text{ObjectComplementOf}(C)} \quad | \quad \underbrace{C_1 \sqcap C_2}_{\text{ObjectIntersectionOf}(C_1,C_2)} \quad | \quad \underbrace{C_1 \sqcup C_2}_{\text{ObjectUnionOf}(C_1,C_2)} \quad |$$

$$\underbrace{\exists R.C}_{\text{ObjectSomeValuesFrom}(R,C)} \quad | \quad \underbrace{\forall R.C}_{\text{ObjectAllValuesFrom}(R,C)}$$

roles (object properties)

$$R \quad ::= \quad \underbrace{P_i}_{\text{role name}} \quad | \; P_i^-$$

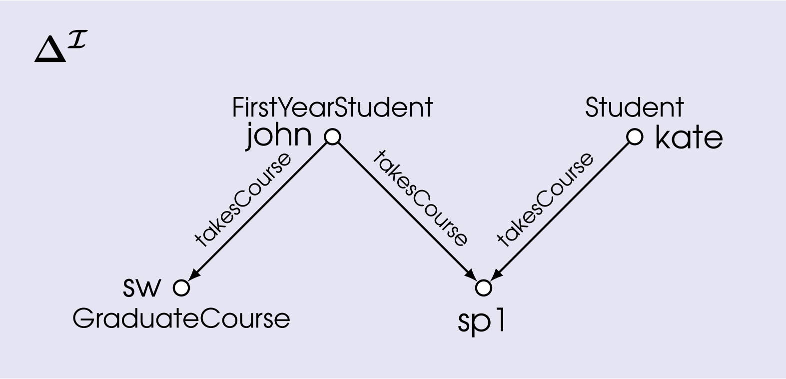
TBox $\mathcal{T}$ $\quad \underbrace{C_1 \sqsubseteq C_2}_{\text{SubClassOf}(C_1,C_2)}$ and $\underbrace{R_1 \sqsubseteq R_2}_{\text{SubObjectPropertyOf}(R_1,R_2)}$

ABox $\mathcal{A}$ $\quad C(a)$ and $R(a, b)$

$\boxed{\textbf{knowledge base } \mathcal{K} = (\mathcal{T}, \mathcal{A}) \qquad \text{(ontology)}}$

DL Semantics

interpretation $\mathcal{I} = (\underbrace{\Delta^{\mathcal{I}}}_{\text{domain}}, \cdot^{\mathcal{I}})$



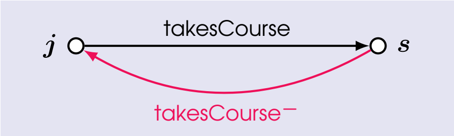
$\cdot^{\mathcal{I}}$ (interpretation function)

individuals a_i $\longrightarrow$ elements $a_i^{\mathcal{I}} \in \Delta^{\mathcal{I}}$

concept names A_i $\longrightarrow$ sets $A_i^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$

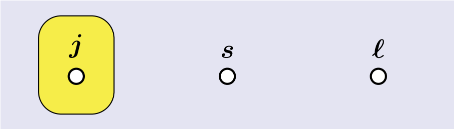
role names P_i $\longrightarrow$ binary relations $P_i^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$

DL Semantics (2)



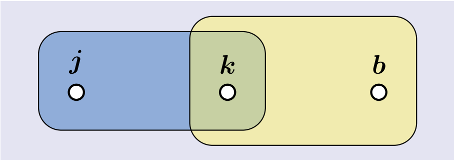
$$(P^-)^\mathcal{I} \;=\; \{(v,u) \mid (u,v) \in P^\mathcal{I}\}$$

$\top^\mathcal{I} \;=\; \Delta^\mathcal{I}$ and $\bot^\mathcal{I} \;=\; \emptyset$



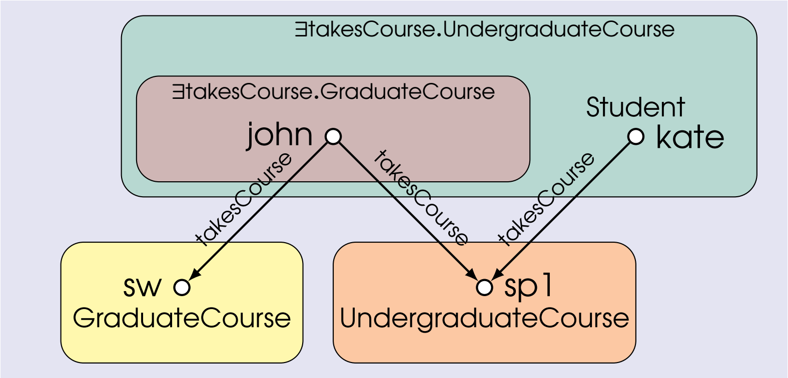
$$(\neg C)^\mathcal{I} \;=\; \Delta^\mathcal{I} \setminus C^\mathcal{I}$$

$(C_1 \sqcap C_2)^\mathcal{I} \;=\; C_1^\mathcal{I} \cap C_2^\mathcal{I}$ and $(C_1 \sqcup C_2)^\mathcal{I} \;=\; C_1^\mathcal{I} \cup C_2^\mathcal{I}$



DL Semantics (3)

$(\exists R.C)^{\mathcal{I}} \;=\; \{\, u \mid$ there is $v \in C^{\mathcal{I}}$ such that $(u, v) \in R^{\mathcal{I}} \,\}$



$(\forall R.C)^{\mathcal{I}} \;=\; \{\, u \mid v \in C^{\mathcal{I}},$ for all v with $(u, v) \in R^{\mathcal{I}} \,\}$

$\forall R.C = \neg\exists R.\neg C$

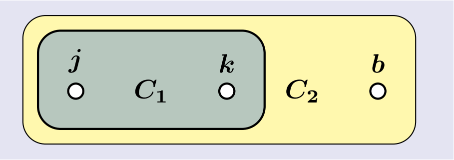
NB. "for all" is true when there are no v with $(u, v) \in R^{\mathcal{I}}$

e.g., sp1 $\in$ ($\forall$takesCourse.UndergraduateCourse)$^{\mathcal{I}}$

sp1 $\in$ ($\forall$takesCourse.$\bot$)$^{\mathcal{I}}$

DL Semantics (4)

$$\mathcal{I} \models C_1 \sqsubseteq C_2 \quad \iff \quad C_1^{\mathcal{I}} \subseteq C_2^{\mathcal{I}}$$



$$\mathcal{I} \models R_1 \sqsubseteq R_2 \quad \iff \quad R_1^{\mathcal{I}} \subseteq R_2^{\mathcal{I}}$$

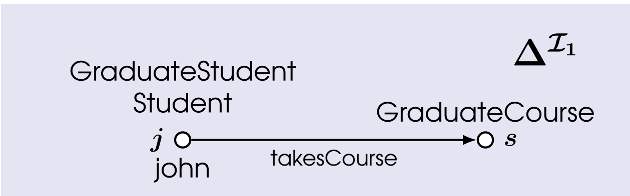
$$\mathcal{I} \models C(a) \quad \iff \quad a^{\mathcal{I}} \in C^{\mathcal{I}}$$

$$\mathcal{I} \models R(a, b) \quad \iff \quad (a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$$

$\mathcal{I}$ is a **model** of $(\mathcal{T}, \mathcal{A})$ if $\mathcal{I} \models \alpha$, for all inclusions α in $\mathcal{T}$ and assertions α in $\mathcal{A}$

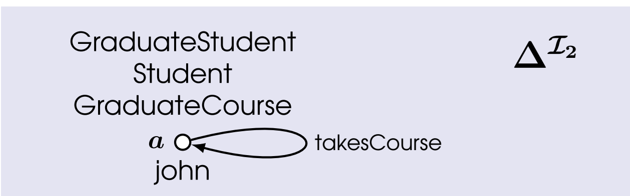
Open World Assumption

$$\begin{aligned}\mathcal{T} &= \{\ \text{GraduateStudent} \sqsubseteq \text{Student}\\ &\qquad \text{GraduateStudent} \sqsubseteq \exists \text{takesCourse}.\text{GraduateCourse}\ \}\\ \mathcal{A} &= \{\ \text{GraduateStudent(john)}\ \}\end{aligned}$$



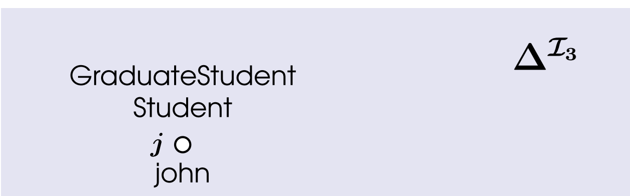
$\text{john}^{\mathcal{I}_1} = j$
$\text{GraduateStudent}^{\mathcal{I}_1} = \{j\}$
$\text{Student}^{\mathcal{I}_1} = \{j\}$
$\text{GraduateCourse}^{\mathcal{I}_1} = \{s\}$
$\text{takesCourse}^{\mathcal{I}_1} = \{(j, s)\}$

is a **model**



$\text{john}^{\mathcal{I}_2} = a$
$\text{GraduateStudent}^{\mathcal{I}_2} = \{a\}$
$\text{Student}^{\mathcal{I}_2} = \{a\}$
$\text{GraduateCourse}^{\mathcal{I}_2} = \{a\}$
$\text{takesCourse}^{\mathcal{I}_2} = \{(a, a)\}$

is a **model**



$\text{john}^{\mathcal{I}_3} = j$
$\text{GraduateStudent}^{\mathcal{I}_3} = \{j\}$
$\text{Student}^{\mathcal{I}_3} = \{j\}$
$\text{GraduateCourse}^{\mathcal{I}_3} = \emptyset$
$\text{takesCourse}^{\mathcal{I}_3} = \emptyset$

is **not a model**

Reasoning: Consistency

a knowledge base $\mathcal{K}$ is **satisfiable** (or **consistent**)

if there exists <u>at least one</u> model of $\mathcal{K}$

(in other words, $\mathcal{K}$ implies **no contradictions**)

Example

$\mathcal{T}$:

$\mathsf{UndergraduateStudent} \sqsubseteq \forall \mathsf{takesCourse}.\mathsf{UndergraduateCourse}$

$\mathsf{UndergraduateCourse} \sqcap \mathsf{GraduateCourse} \sqsubseteq \bot$

$\mathcal{A}$:

$\mathsf{UndergraduateStudent}(\mathsf{john})$

$\mathsf{takesCourse}(\mathsf{john}, \mathsf{sw})$

$\mathsf{GraduateCourse}(\mathsf{sw})$

$(\mathcal{T}, \mathcal{A})$ is **inconsistent**:

John (as an undergraduate student) can take only undergraduate courses.
We know, however, that he takes a graduate course,
which cannot be an undergraduate one.

Reasoning: Entailment

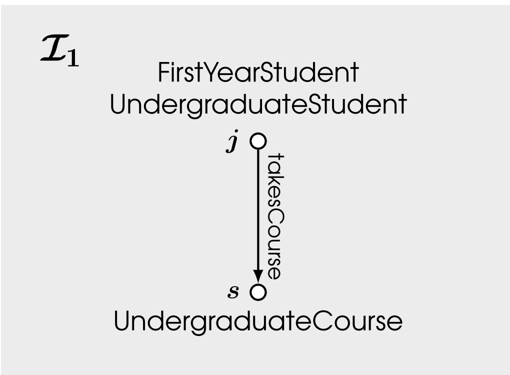
$C_1 \sqsubseteq C_2$ is **entailed by** $\mathcal{K}$ $\qquad$ $\boxed{\mathcal{K} \models C_1 \sqsubseteq C_2}$

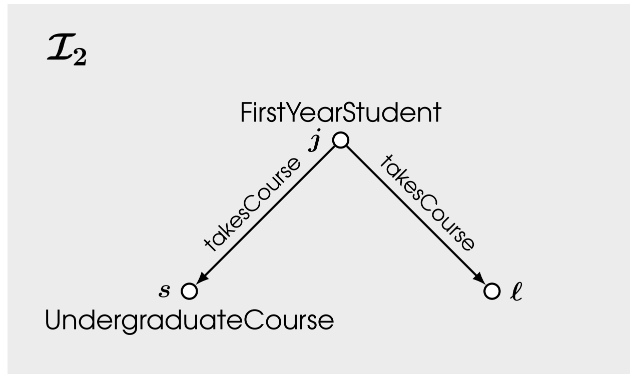
if $\mathcal{I} \models C_1 \sqsubseteq C_2$ for all models $\mathcal{I}$ of $\mathcal{K}$

(entailment for role inclusions and concept and role assertions is defined similarly)

$\mathcal{T}$: $\quad \forall$takesCourse.UndergraduateCourse $\sqsubseteq$ UndergraduateStudent

FirstYearStudent $\sqsubseteq$ $\exists$takesCourse.UndergraduateCourse.





$\mathcal{I}_1 \models \mathcal{T}$

$\mathcal{I}_1 \models$ FirstYearStudent $\sqsubseteq$ UndergraduateStudent

$\mathcal{I}_2 \models \mathcal{T}$

$\mathcal{I}_2 \models$ FirstYearStudent $\not\sqsubseteq$ UndergraduateStudent

Reasoning: Entailment (2)

C_1 is **subsumed** by C_2 with respect to $\mathcal{K}$ if $\mathcal{K} \models C_1 \sqsubseteq C_2$

Proposition $(\mathcal{T}, \mathcal{A}) \models C_1 \sqsubseteq C_2$ iff $(\mathcal{T}, \mathcal{A} \cup \{C_1(a), \neg C_2(a)\})$ is not satisfiable
for a **fresh** a

Proof iff = if and only if

($\Rightarrow$, only if) Let $(\mathcal{T}, \mathcal{A}) \models C_1 \sqsubseteq C_2$.
Assume, for the sake of contradiction, that $(\mathcal{T}, \mathcal{A} \cup \{C_1(a), \neg C_2(a)\})$ is satisfiable.
Then there is an interpretation $\mathcal{I}$ such that $\mathcal{I} \models (\mathcal{T}, \mathcal{A})$ (a model of $(\mathcal{T}, \mathcal{A})$),
$a^{\mathcal{I}} \in C_1^{\mathcal{I}}$ and $a^{\mathcal{I}} \notin C_2^{\mathcal{I}}$, contrary to $(\mathcal{T}, \mathcal{A}) \models C_1 \sqsubseteq C_2$

($\Leftarrow$, if) Let $(\mathcal{T}, \mathcal{A} \cup \{C_1(a), \neg C_2(a)\})$ be not satisfiable.
Assume, for the sake of contradiction, that $(\mathcal{T}, \mathcal{A}) \not\models C_1 \sqsubseteq C_2$.
Then there is a model of $(\mathcal{T}, \mathcal{A})$ and some $u \in C_1^{\mathcal{I}}$ such that $u \notin C_2^{\mathcal{I}}$.
Since a is fresh, we can redefine $a^{\mathcal{I}}$ by taking $a^{\mathcal{I}} = u$,
which will contradict inconsistency of $(\mathcal{T}, \mathcal{A} \cup \{C_1(a), \neg C_2(a)\})$.

a is an **instance of** C with respect to $\mathcal{K}$ if $\mathcal{K} \models C(a)$

Proposition $(\mathcal{T}, \mathcal{A}) \models C(a)$ iff $(\mathcal{T}, \mathcal{A} \cup \{\neg C(a)\})$ is not satisfiable

Conjunctive Queries (CQs)

conjunctive query

$$\boxed{q(\underbrace{\vec{x}}_{\text{answer variables}}) \quad = \quad \exists \underbrace{\vec{y}}_{\text{existentially quantified variables}} \varphi(\vec{x}, \vec{y})}$$

$\varphi(\vec{x}, \vec{y})$ is a conjunction of atoms such as $\underbrace{A}_{\text{concept name}}(z)$ and $\underbrace{P}_{\text{role name}}(z_1, z_2)$

z, z_1 and z_2 are **terms** = an individual name or a variable from $\vec{x}$ or $\vec{y}$

Example

$$q(x) = \exists y, z \; (\text{friend}(x, y) \land \text{Female}(y) \land \text{loves}(y, z) \land \text{Male}(z))$$

```
SELECT ?x
WHERE {
      ?x :friend ?y.
      ?y a :Female.
      ?y :loves ?z.
      ?z a :Male.
}
```

Certain Answers to CQs

$q(\vec{x}) = \exists \vec{y}\, \varphi(\vec{x}, \vec{y})$ is a CQ with $\vec{x} = (x_1, \dots, x_n)$

$\vec{a} = (a_1, \dots, a_n)$ is a tuple of individual names from $\mathcal{A}$

$q(\vec{a})$ is the result of replacing each x_i in $\exists \vec{y}\, \varphi(\vec{x}, \vec{y})$ with a_i

$\vec{a}$ is a **certain answer** to $q(\vec{x})$ over $(\mathcal{T}, \mathcal{A})$ $\qquad \mathcal{I} \models q(\vec{a})$

if, for any model $\mathcal{I}$ of $(\mathcal{T}, \mathcal{A})$, the sentence $q(\vec{a})$ is true in $\mathcal{I}$

q without answer variables is **Boolean**

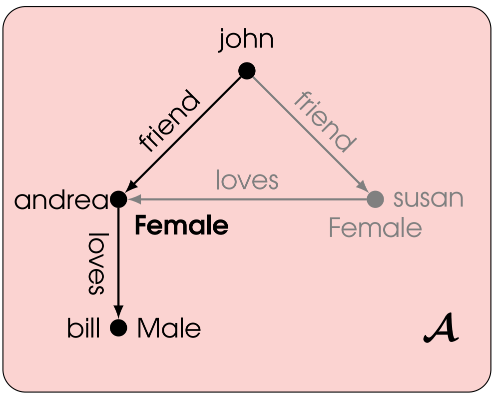
a certain answer to q over $(\mathcal{T}, \mathcal{A})$ is 'yes' if $(\mathcal{T}, \mathcal{A}) \models q$ and 'no' otherwise

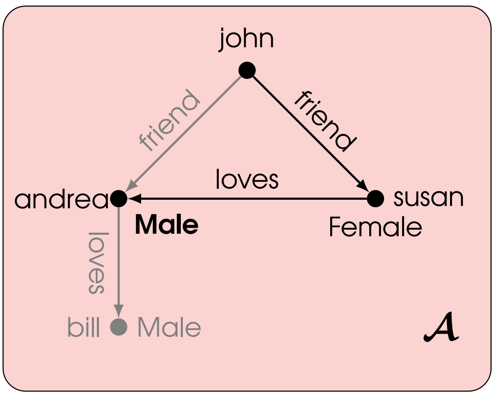
Andrea's Example

$\mathcal{T}$: $\top \sqsubseteq \mathsf{Male} \sqcup \mathsf{Female}, \quad \mathsf{Male} \sqcap \mathsf{Female} \sqsubseteq \bot$

$\mathcal{A}$: friend(john, susan), friend(john, andrea), Female(susan)
loves(susan, andrea), loves(andrea, bill), Male(bill)

$$q(x) = \exists y, z\ (\mathsf{friend}(x, y) \wedge \mathsf{Female}(y) \wedge \mathsf{loves}(y, z) \wedge \mathsf{Male}(z))$$





NB: the same as finding instances of $\exists$friend.(Female $\sqcap$ $\exists$loves.Male)

From OWL to DL

ObjectPropertyDomain(takesCourse, Student)

$$\exists \text{takesCourse}.\top \sqsubseteq \text{Student}$$

ObjectPropertyRange(takesCourse, Course)

$$\exists \text{takesCourse}^{-}.\top \sqsubseteq \text{Course}$$

or

$$\top \sqsubseteq \forall \text{takesCourse}.\text{Course}$$

Proposition The following are equivalent (have the same models)

$C_1 \sqsubseteq \forall P.C_2$	and	$\exists P^{-}.C_1 \sqsubseteq C_2$
$C_1 \sqcup C_2 \sqsubseteq C$	and	$\{C_1 \sqsubseteq C, \ C_2 \sqsubseteq C\}$
$C \sqsubseteq C_1 \sqcap C_2$	and	$\{C \sqsubseteq C_1, \ C \sqsubseteq C_2\}$
$C_1 \sqsubseteq \neg C_2$	and	$C_1 \sqcap C_2 \sqsubseteq \bot$

From OWL to DL (2)

EquivalentClasses$(C_1, C_2, \dots, C_n)$

$$C_1 \sqsubseteq C_2, \quad C_2 \sqsubseteq C_3, \quad \dots, \quad C_{n-1} \sqsubseteq C_n, \quad C_n \sqsubseteq C_1$$

NB: $A \equiv B$ stands for '$A \sqsubseteq B$ and $B \sqsubseteq A$'

DisjointClasses$(C_1, C_2, \dots, C_n)$

$$C_i \sqcap C_j \sqsubseteq \bot, \quad \text{for all } i, j \text{ with } 1 \leq i < j \leq n$$

DisjointUnion$(C, C_1, C_2, \dots, C_n)$,

$$C \equiv C_1 \sqcup \cdots \sqcup C_n$$

SymmetricObjectProperty(P)

$$P^- \sqsubseteq P$$

DL Zoo

$\mathcal{ALCHI}$

$\mathcal{AL}$ – attributive language

$\mathcal{C}$ – complement $\neg C$

$\mathcal{I}$ – role inverses P^-

$\mathcal{H}$ – role inclusions $R_1 \sqsubseteq R_2$

$\mathcal{S}$ – $\mathcal{ALC}$ + transitive roles

$\mathcal{N}$ – unqualified number restrictions $\geq q\, R.\top$

$\mathcal{O}$ – nominals $\{a\}$

$\mathcal{SHOIN} \approx$ OWL 1.0

$\mathcal{Q}$ – qualified number restrictions $\geq q\, R.C$

$\mathcal{F}$ – functionality constraints $\geq 2\, R.\top \sqsubseteq \bot$

$\mathcal{SHIF} \approx$ OWL Lite

$\mathcal{R}$ – role chains and $\exists R.$Self

$\mathcal{SROIQ} \approx$ OWL 2

Complexity of Reasoning

The satisfiability problem is **ExpTime**-complete for $\mathcal{ALCHI}$ KBs
and **N2ExpTime**-complete for $\mathcal{SROIQ}$ KBs

Concept and role subsumption and instance checking are **ExpTime**- and **coN2ExpTime**-complete for, respectively, $\mathcal{ALCHI}$ and $\mathcal{SROIQ}$ KBs

CQ entailment over $\mathcal{ALCHI}$ KBs is **2ExpTime**-complete

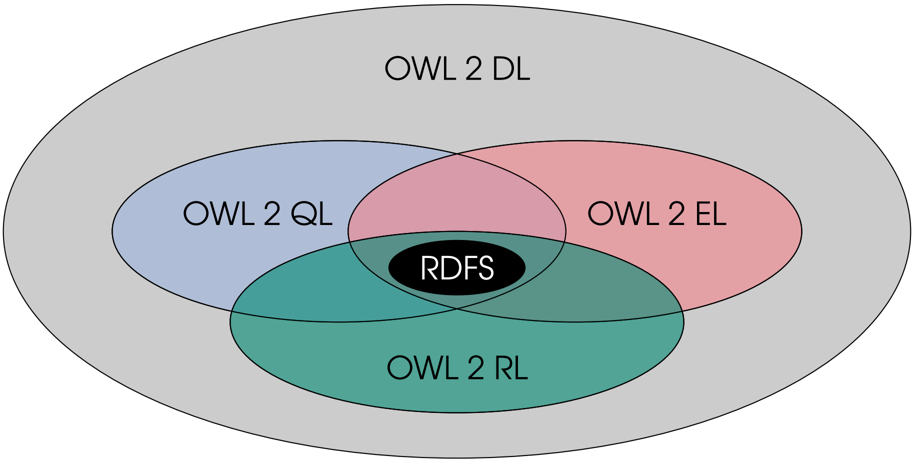
CQ entailment over $\mathcal{SROIQ}$ is not even known to be decidable

DL Complexity Navigator: `www.cs.man.ac.uk/~ezolin/dl`

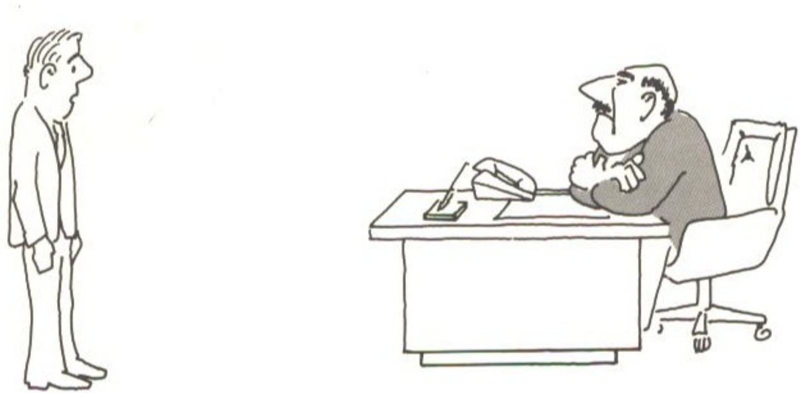
practical reasoners for OWL 2 DL: FaCT++, HermiT, Pellet

Part 2

OWL 2 Profiles



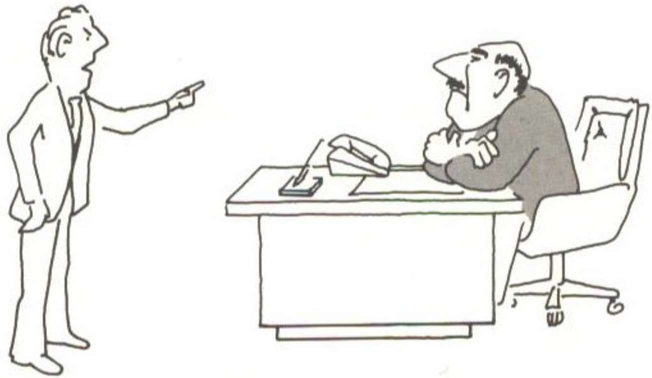
(In)tractable Reasoning



"I can't find an efficient algorithm, I guess I'm just too dumb."

Garey & Johnson, 1979
Computers and Intractability: A Guide to the Theory of NP-Completeness

(In)tractable Reasoning



"I can't find an efficient algorithm, because no such algorithm is possible!"

Garey & Johnson, 1979
Computers and Intractability: A Guide to the Theory of NP-Completeness

(In)tractable Reasoning

"I can't find an efficient algorithm, but neither can all these famous people."

Garey & Johnson, 1979
Computers and Intractability: A Guide to the Theory of NP-Completeness

OWL 2 Profiles: No Disjunction

graph 3-colourability problem is **NP**-complete

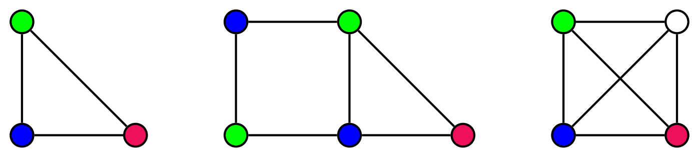
given an (undirected) graph $G = (V, E)$,
decide whether each of its vertices can be painted in one of the three colours
in such a way that no pair of adjacent vertices has the same colour

$\mathcal{A}_G$: edge(v_1, v_2), for each $\{v_1, v_2\} \in E$

$\mathcal{T}$:

$$\boxed{\begin{array}{ll} \top \sqsubseteq C_1 \sqcup C_2 \sqcup C_3 & \\ C_i \sqcap C_j \sqsubseteq \bot, & 1 \leq i < j \leq 3, \\ C_i \sqcap \exists \text{edge}.C_i \sqsubseteq \bot, & 1 \leq i \leq 3 \end{array}}$$

models of $(\mathcal{T}, \mathcal{A}_G)$ $\sim$
3-colouring of G



$(\mathcal{T}, \mathcal{A}_G)$ is satisfiable $\iff$ G is 3-colourable

the satisfiability problem for KBs in any DL able to express this TBox
is **NP**-hard (intractable)

OWL 2 RL

Description Logic Programs (Grosof, Horrocks, Volz, Decker ’03) and pD* (ter Horst ’05)

supported by Oracle Database 11g, OWLIM, BaseVISor, ELLY, Jena and RDFox

RL TBox:

$$\boldsymbol{B \sqsubseteq A}, \qquad \boldsymbol{R_1 \sqsubseteq R_2} \quad \text{and} \quad \boldsymbol{B \sqsubseteq \bot}$$

only concept names on the right

$$\boldsymbol{B} \quad ::= \quad \boldsymbol{A} \quad | \quad \exists \boldsymbol{R}.\top \quad | \quad \exists \boldsymbol{R}.\boldsymbol{B} \quad | \quad \boldsymbol{B_1} \sqcap \boldsymbol{B_2}$$

simple ABox:

$\boldsymbol{A(a)}$ and $\boldsymbol{P(a, b)}$

NB: this is a simplified version — see the equivalences

RL and Datalog

UndergraduateStudent $\sqsubseteq$ Student

$$\forall x\,(\mathsf{UndergraduateStudent}(x) \to \mathsf{Student}(x))$$

$\exists$takesCourse.UndergraduateCourse $\sqsubseteq$ UndergraduateStudent

$$\forall x \forall y\,(\mathsf{takesCourse}(x, y) \land \mathsf{UndergraduateCourse}(y) \\ \to \mathsf{UndergraduateStudent}(x))$$

standard translation $\quad ST_x(A) = A(x)$ and $ST_{x,y}(P) = P(x, y)$

$$ST_{x,y}(P^-) = P(y, x) \qquad ST_x(B_1 \sqcap B_2) = ST_x(B_1) \land ST_x(B_2)$$
$$ST_x(\exists R.\top) = \exists y\, ST_{x,y}(R) \qquad ST_x(\exists R.B) = \exists y\,(ST_{x,y}(R) \land ST_y(B))$$

$$(B \sqsubseteq A)^* = \forall x\,(ST_x(B) \to ST_x(A))$$
$$(R_1 \sqsubseteq R_2)^* = \forall x \forall y\,(ST_{x,y}(R_1) \to ST_{x,y}(R_2))$$
$$(B \sqsubseteq \bot)^* = \forall x\,(ST_x(B) \to \bot)$$

$$\forall \vec{y}\,(\gamma_1(\vec{y}) \land \cdots \land \gamma_k(\vec{y}) \to \gamma_0(\vec{y}))$$
$$\forall \vec{y}\,(\gamma_1(\vec{y}) \land \cdots \land \gamma_k(\vec{y}) \to \bot)$$

Horn clauses (datalog programs)

Chase: Example

$\mathcal{T}$: $\exists$takesCourse.UndergraduateCourse $\sqsubseteq$ UndergraduateStudent
UndergraduateStudent $\sqsubseteq$ Student

$\mathcal{A}$: takesCourse(john, sp1), UndergraduateCourse(sp1)

step 0: $\mathcal{I}_0 \approx \mathcal{A}$

$$\begin{aligned}
&\Delta^{\mathcal{I}_0} = \{\text{john}, \text{sp1}\}\\
&\text{takesCourse}^{\mathcal{I}_0} = \{(\text{john}, \text{sp1})\}\\
&\text{UndergraduateCourse}^{\mathcal{I}_0} = \{\text{sp1}\}\\
&\text{UndergraduateStudent}^{\mathcal{I}_0} = \emptyset\\
&\text{Student}^{\mathcal{I}_0} = \emptyset
\end{aligned}$$

$\mathcal{I}_0 \not\models \mathcal{T}$

step 1: $\Delta^{\mathcal{I}_1} = \Delta^{\mathcal{I}_0}$ and $\cdot^{\mathcal{I}_1}$ expands $\cdot^{\mathcal{I}_0}$ with

$$\text{UndergraduateStudent}^{\mathcal{I}_1} = \{\text{john}\}$$

$\mathcal{I}_1 \not\models \mathcal{T}$

step 2: $\Delta^{\mathcal{I}_2} = \Delta^{\mathcal{I}_1}$ and $\cdot^{\mathcal{I}_1}$ expands $\cdot^{\mathcal{I}_1}$ with

$$\text{Student}^{\mathcal{I}_2} = \{\text{john}\}$$

$\mathcal{I}_2 \models \mathcal{T}$

stop: $\mathcal{I}_2$ is a model of $(\mathcal{T}, \mathcal{A})$

Chase for RL

the **standard model** $\mathcal{I}_\mathcal{A}$ of a simple ABox $\mathcal{A}$ is

$$
\begin{aligned}
\Delta^{\mathcal{I}_\mathcal{A}} &= \text{ind}(\mathcal{A})\\
a^{\mathcal{I}_\mathcal{A}} &= a, \ \text{ for } a \in \text{ind}(\mathcal{A})\\
A^{\mathcal{I}_\mathcal{A}} &= \{a \mid A(a) \in \mathcal{A}\}, \ \text{ for concept name } A\\
P^{\mathcal{I}_\mathcal{A}} &= \{(a, b) \mid P(a, b) \in \mathcal{A}\}, \ \text{ for role name } P
\end{aligned}
$$

chase rules: (**forward chaining**)

- **(c)** if $a \in B^{\mathcal{I}_k}$, $B \sqsubseteq A \in \mathcal{T}$ but $a \notin A^{\mathcal{I}_k}$, then add a to $A^{\mathcal{I}_{k+1}}$
- **(r)** if $(a, b) \in R_1^{\mathcal{I}_k}$, $R_1 \sqsubseteq R_2 \in \mathcal{T}$ but $(a, b) \notin R_2^{\mathcal{I}_k}$, then add (a, b) to $R_2^{\mathcal{I}_{k+1}}$
- **(b)** if $a \in B^{\mathcal{I}_k}$ and $B \sqsubseteq \bot \in \mathcal{T}$, then the process terminates

the domains of the $\mathcal{I}_k$ are finite and all coincide with ind$(\mathcal{A})$,
$\Rightarrow$ the process **terminates** after a finite number of steps

– if **(b)** is not applicable then the result is a model of $(\mathcal{T}, \mathcal{A})$,
the **canonical model** $\mathcal{C}_{\mathcal{T},\mathcal{A}}$ of $(\mathcal{T}, A)$

– otherwise, $(\mathcal{T}, \mathcal{A})$ is inconsistent

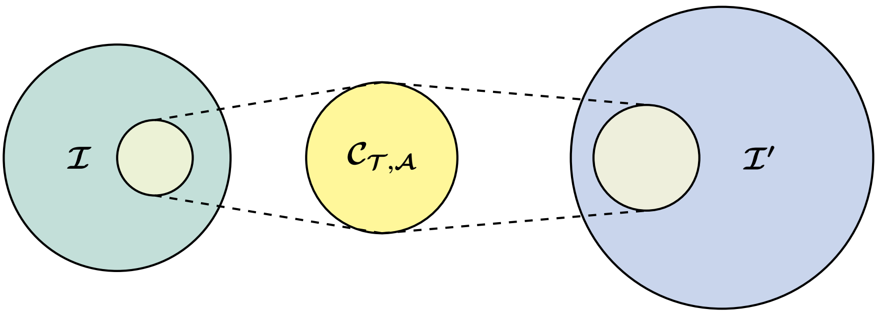
Canonical Model is Universal

a **homomorphism** h from an interpretation $\mathcal{I}_1$ to an interpretation $\mathcal{I}_2$ is a map from $\Delta^{\mathcal{I}_1}$ to $\Delta^{\mathcal{I}_2}$ such that

- $h(a^{\mathcal{I}_1}) = a^{\mathcal{I}_2}$, for each individual name a
- $h(u) \in A^{\mathcal{I}_2}$, for any $u \in A^{\mathcal{I}_1}$ and any concept name A
- $(h(u), h(v)) \in P^{\mathcal{I}_2}$, for any $(u, v) \in P^{\mathcal{I}_1}$ and any role name P

Lemma $\mathcal{C}_{\mathcal{T},\mathcal{A}}$ is **universal**

$\mathcal{C}_{\mathcal{T},\mathcal{A}}$ can be homomorphically mapped into any other model of $(\mathcal{T}, \mathcal{A})$



Complexity of Reasoning in RL

Theorem Let $(\mathcal{T}, \mathcal{A})$ be a consistent RL KB. Then $\mathcal{C}_{\mathcal{T},\mathcal{A}} \models (\mathcal{T}, \mathcal{A})$. In addition,

$$(\mathcal{T}, \mathcal{A}) \models B \sqsubseteq A \iff \mathcal{C}_{\mathcal{T},\mathcal{A}} \models B \sqsubseteq A$$

$$(\mathcal{T}, \mathcal{A}) \models R_1 \sqsubseteq R_2 \iff \mathcal{C}_{\mathcal{T},\mathcal{A}} \models R_1 \sqsubseteq R_2$$

$$(\mathcal{T}, \mathcal{A}) \models q(\vec{a}) \iff \mathcal{C}_{\mathcal{T},\mathcal{A}} \models q(\vec{a})$$

Theorem The problems of KB consistency, concept and role subsumption and instance checking are **P**-complete in RL

The problem of CQ entailment in RL is **NP**-complete

Proof Construct the canonical model (in a polynomial number of steps).

Check the condition in the canonical model.

OWL 2 EL

SNOMED CT (Spackman '00, Baader, Brandt & Lutz '05)

Pericardium $\sqsubseteq$ Tissue $\sqcap$ containedIn.Heart
Pericarditis $\sqsubseteq$ Inflammation $\sqcap$ hasLocation.Pericardium
Inflammation $\sqsubseteq$ Disease $\sqcap$ actsOn.Tissue
Disease $\sqcap$ hasLocation.containedIn.Heart $\sqsubseteq$ HeartDisease $\sqcap$ NeedsTreatment

EL TBox:

$C_1 \sqsubseteq C_2$ and $P_1 \sqsubseteq P_2$

$$C \quad ::= \quad A \quad | \quad \exists P.\top \quad | \quad \exists P.C \quad | \quad C_1 \sqcap C_2$$

no inverses

simple ABox:

$A(a)$ and $P(a, b)$

EL and Existential Rules

Person $\sqsubseteq$ $\exists$parent.Person

$$\forall x\,(\text{Person}(x) \to \exists y\,(\text{parent}(x, y) \land \text{Person}(y)))$$

the standard translation can be extended to EL

$$\forall \vec{y}\,(\gamma_1(\vec{y}) \land \dots \land \gamma_k(\vec{y}) \to \exists \vec{x}\,\gamma_0(\vec{x}, \vec{y}))$$

existential rules

chase rules: (Johnson & Klug '84)

(c′) if $d \in C^{\mathcal{I}_k}$ and $C \sqsubseteq A \in \mathcal{T}$, then add d to $A^{\mathcal{I}_{k+1}}$

(r′) if $(d, d') \in P_1^{\mathcal{I}_k}$ and $P_1 \sqsubseteq P_2 \in \mathcal{T}$, then we add (d, d') to $P_2^{\mathcal{I}_{k+1}}$

(e) if $d \in C^{\mathcal{I}_k}$ and $C \sqsubseteq \exists P.D \in \mathcal{T}$, where D is a concept name or $\top$,
then take a **fresh labelled null**, d', and add d' to $D^{\mathcal{I}_{k+1}}$ and (d, d') to $P^{\mathcal{I}_{k+1}}$

NB: oblivious chase — no check whether there is such an element

EL Chase: Example

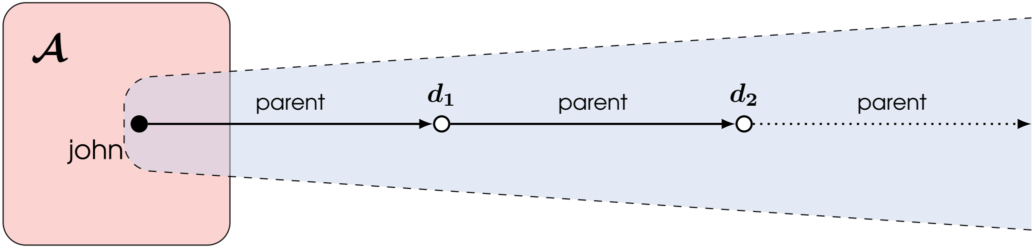
$\mathcal{T}$: Person $\sqsubseteq$ $\exists$parent.Person

$\mathcal{A}$: Person(john)

step 0: $\Delta^{\mathcal{I}_0} = \{\text{john}\}$, $\text{Person}^{\mathcal{I}_0} = \{\text{john}\}$, $\text{parent}^{\mathcal{I}_0} = \emptyset$ $\quad \mathcal{I}_0 = \mathcal{I}_\mathcal{A}$

step 1: $\Delta^{\mathcal{I}_1} = \{\text{john}, d_1\}$, $\text{Person}^{\mathcal{I}_1} = \{\text{john}, d_1\}$, $\text{parent}^{\mathcal{I}_1} = \{(\text{john}, d_1)\}$

step 2: $\Delta^{\mathcal{I}_2} = \{\text{john}, d_1, d_2\}$, $\text{Person}^{\mathcal{I}_2} = \{\text{john}, d_1, d_2\}$,
$\text{parent}^{\mathcal{I}_2} = \{(\text{john}, d_1), (d_1, d_2)\}$



1. this canonical model is infinite
2. there is no finite universal model
3. there are many universal models (each is infinite)

Normal Form in EL

normal form of concept inclusions

$$\boxed{A_1 \sqcap A_2 \sqsubseteq A, \qquad \exists P.D \sqsubseteq A \quad \text{or} \quad A \sqsubseteq \exists P.D}$$

P is a role name, A, A_1 and A_2 are concept names and D is either a concept name or $\top$

Example: $\exists P.A \sqcap B \sqsubseteq \exists R.\exists P.A$

introduce abbreviations: C is $\exists P.A$ and D is $\exists R.C$

result:

$$C \sqcap B \sqsubseteq D, \qquad \exists P.A \sqsubseteq C, \qquad C \sqsubseteq \exists P.A, \qquad D \sqsubseteq \exists R.C, \qquad \exists R.C \sqsubseteq D$$

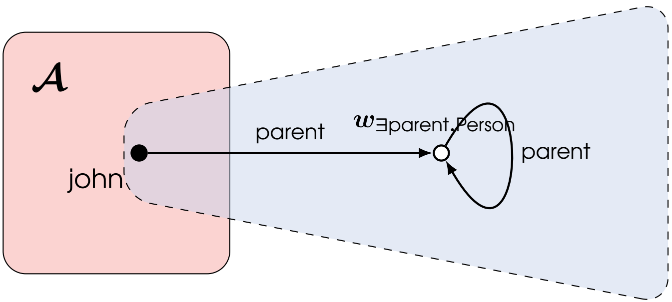
Reasoning in EL

do not construct the infinite chase, **re-use** the labelled nulls

for each positive $\exists S.D$, take a fixed **witness** $w_{\exists S.D}$

(e-c) if $d \in C^{\mathcal{I}_k}$ and $C \sqsubseteq \exists P.D \in \mathcal{T}$, where D is a concept name or $\top$,

then add $w_{\exists S.D}$ to $D^{\mathcal{I}_{k+1}}$ and (d, d') to $P^{\mathcal{I}_{k+1}}$



the **generating model** $\mathcal{G}_{\mathcal{T},\mathcal{A}}$ of $(\mathcal{T}, \mathcal{A})$

NB: alternatively,

take the canonical model and identify all labelled nulls introduced for the same $\exists S.D$

Reasoning in EL (2)

Theorem Let $(\mathcal{T},\mathcal{A})$ be an EL KB. Then $\mathcal{G}_{\mathcal{T},\mathcal{A}} \models (\mathcal{T},\mathcal{A})$. In addition,

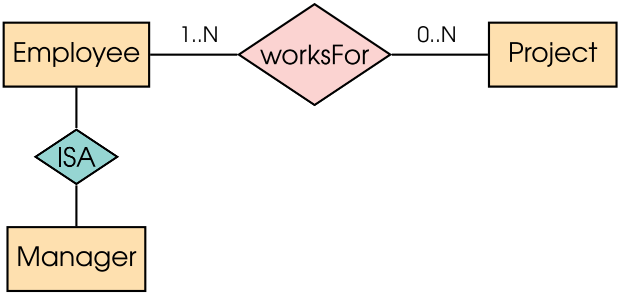
$$(\mathcal{T},\mathcal{A}) \models C_1 \sqsubseteq C_2 \iff \mathcal{G}_{\mathcal{T},\mathcal{A}} \models C_1 \sqsubseteq C_2$$
$$(\mathcal{T},\mathcal{A}) \models P_1 \sqsubseteq P_2 \iff \mathcal{G}_{\mathcal{T},\mathcal{A}} \models P_1 \sqsubseteq P_2$$
$$(\mathcal{T},\mathcal{A}) \models C(a) \iff \mathcal{G}_{\mathcal{T},\mathcal{A}} \models C(a)$$
$$(\mathcal{T},\mathcal{A}) \models P(a,b) \iff \mathcal{G}_{\mathcal{T},\mathcal{A}} \models P(a,b)$$

Theorem The problems of concept and role subsumption and instance checking are **P**-complete in EL

NB: $\mathcal{G}_{\mathcal{T},\mathcal{A}}$ does **not** give correct answers to queries

$$q = \exists x\, \text{parent}(x,x)$$

OWL 2 QL



(Calvanese et al. '05)

$\exists \text{worksFor}.\top \sqsubseteq \text{Employee}$
$\exists \text{worksFor}^-.\top \sqsubseteq \text{Project}$
$\text{Manager} \sqsubseteq \text{Employee}$
$\text{Project} \sqsubseteq \exists \text{worksFor}^-.\top$

QL TBox:

$\boldsymbol{B} \sqsubseteq \boldsymbol{C}$ and $\boldsymbol{R_1} \sqsubseteq \boldsymbol{R_2}$

$$\begin{array}{lllllll} \boldsymbol{B} & ::= & \boldsymbol{A} & \mid & \exists \boldsymbol{R}.\top & & \\ \boldsymbol{C} & ::= & \boldsymbol{A} & \mid & \exists \boldsymbol{R}.\top & \mid & \exists \boldsymbol{R}.\boldsymbol{C} \end{array}$$

no $\exists R.C$ on the left

simple ABox:

$\boldsymbol{A}(\boldsymbol{a})$ and $\boldsymbol{P}(\boldsymbol{a}, \boldsymbol{b})$

Chase for QL

normal form of concept inclusions

$$\boxed{A' \sqsubseteq A, \qquad \exists R \sqsubseteq A \quad \text{or} \quad A \sqsubseteq \exists R.D}$$

R is a role, A and A' are concept names and D is either a concept name or $\top$

chase rules:

(c′) if $d \in B^{\mathcal{I}_k}$ and $B \sqsubseteq A \in \mathcal{T}$, then add d to $A^{\mathcal{I}_{k+1}}$

(r′) if $(d, d') \in R_1^{\mathcal{I}_k}$ and $R_1 \sqsubseteq R_2 \in \mathcal{T}$, then add (d, d') to $R_2^{\mathcal{I}_{k+1}}$

(e) if $d \in B^{\mathcal{I}_k}$ and $B \sqsubseteq \exists R.D \in \mathcal{T}$, where D is a concept name or $\top$,
then take a **fresh labelled null**, d', and add d' to $D^{\mathcal{I}_{k+1}}$ and (d, d') to $R^{\mathcal{I}_{k+1}}$

Unravelling of the Generating Structure

for each $\exists R.D$ that occurs positively in $\mathcal{T}$, take a **witness** $w_{\exists R.D}$

generating relation $\leadsto_{\mathcal{T},\mathcal{A}}$:

$a \leadsto_{\mathcal{T},\mathcal{A}} w_{\exists R.D}$	if	$a \in \mathsf{ind}(\mathcal{A})$, $\mathcal{I}_{\mathcal{A}} \models B(a)$ and $\mathcal{T} \models B \sqsubseteq \exists R.D$
$w_{\exists S.B} \leadsto_{\mathcal{T},\mathcal{A}} w_{\exists R.D}$	if	$\mathcal{T} \models \exists S^{-} \sqsubseteq \exists R.D$ or $\mathcal{T} \models B \sqsubseteq \exists R.D$

$\leadsto_{\mathcal{T},\mathcal{A}}$**-path** σ is any $a w_{\exists R_1.D_1} \cdots w_{\exists R_n.D_n}$

– $a \in \mathsf{ind}(\mathcal{A})$ and,

– if $n > 0$ then $a \leadsto_{\mathcal{T},\mathcal{A}} w_{\exists R_1.D_1}$ and $w_{\exists R_i.D_i} \leadsto_{\mathcal{T},\mathcal{A}} w_{\exists R_{i+1}.D_{i+1}}$, for $i < n$

Canonical Model: Example

$\mathcal{T}$: RA $\sqsubseteq \exists$worksOn.Project   worksOn$^- \sqsubseteq$ involves

Project $\sqsubseteq \exists$isManagedBy.Prof   isManagedBy $\sqsubseteq$ involves

$\mathcal{A}$: RA(chris), worksOn(chris, dyn), Project(dyn), Lecturer(dave), worksOn(dave, dyn)

$w_1 = w_{\exists \text{worksOn.Project}}$ and $w_2 = w_{\exists \text{isManagedBy.Prof}}$

generating relation

chris $\leadsto_{\mathcal{T},\mathcal{A}} w_1$,   dyn $\leadsto_{\mathcal{T},\mathcal{A}} w_2$, $w_1 \leadsto_{\mathcal{T},\mathcal{A}} w_2$

$\leadsto_{\mathcal{T},\mathcal{A}}$-paths

chris,   chris w_1,   chris w_1 w_2,   dyn,   dyn w_2 and dave

Canonical Model: Example (2)

$\mathcal{T}$:

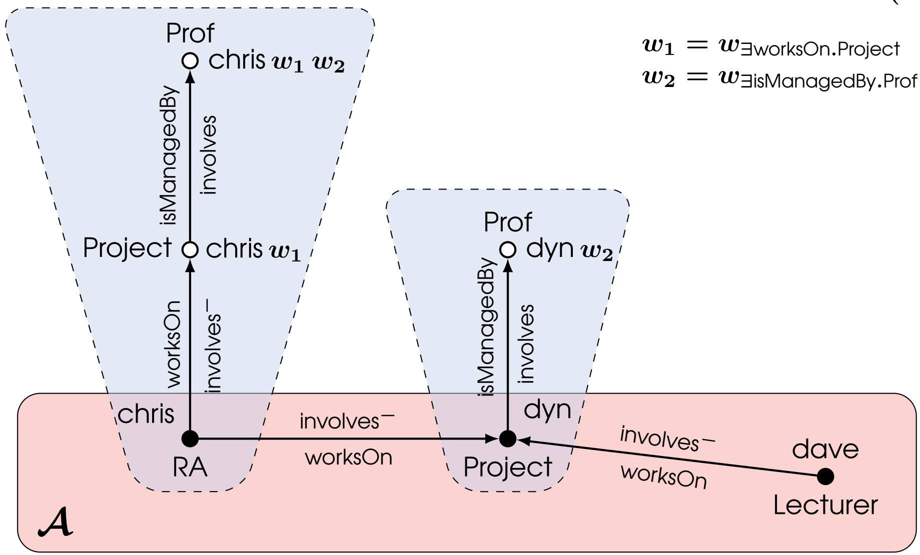
$\mathsf{RA} \sqsubseteq \exists \mathsf{worksOn}.\mathsf{Project}$ $\qquad$ $\mathsf{worksOn}^- \sqsubseteq \mathsf{involves}$

$\mathsf{Project} \sqsubseteq \exists \mathsf{isManagedBy}.\mathsf{Prof}$ $\qquad$ $\mathsf{isManagedBy} \sqsubseteq \mathsf{involves}$

$\mathcal{A}$: RA(chris), worksOn(chris, dyn), Project(dyn), Lecturer(dave), worksOn(dave, dyn)

$w_1 = w_{\exists \mathsf{worksOn}.\mathsf{Project}}$

$w_2 = w_{\exists \mathsf{isManagedBy}.\mathsf{Prof}}$



Comparing Profiles

	RL	EL	QL
inverse roles P^-	+	–	+
$\exists R.C$ on the right	–	+	+
$\exists R.C$ on the left	+	+	–

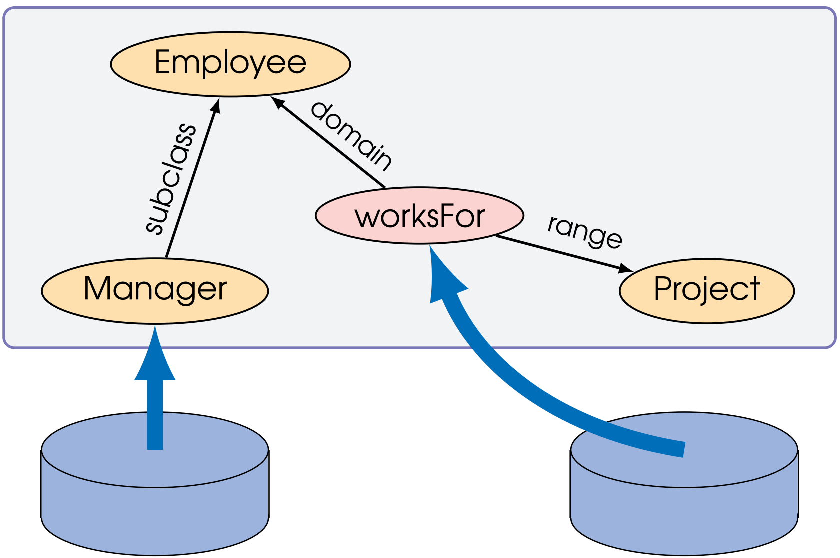
Part 3

OBDA and Query Rewriting

Ontology-Based Data Access

Aim: to achieve **logical transparency** in accessing data

- hide from the user where and how data is stored
- present only a **conceptual view** of the data
- **query** the data sources through the **conceptual model** using **RDBMSs**



ontology
intensional level,
conceptual structure of
the domain,
TBox

(GAV) mappings

data sources
extensional level,
instances of conceptual
elements,
ABox

Databases: Data and the Closed World Assumption

data is **completely** specified (**closed world assumption**) and is typically **large**

what is specified is true, everything else is false

data: $\mathsf{Employee} = \{\mathsf{john}, \mathsf{mary}, \mathsf{nick}\}$
$\mathsf{Manager} = \{\mathsf{john}, \mathsf{nick}\}$
$\mathsf{Project} = \{\mathsf{A}, \mathsf{B}\}$
$\mathsf{worksFor} = \{(\mathsf{john}, \mathsf{A}), (\mathsf{mary}, \mathsf{B})\}$

query: $q(x) \leftarrow \mathsf{Employee}(x)$

answer: $\{\mathsf{john}, \mathsf{mary}, \mathsf{nick}\}$

NB: not having nick in Employee would violate the integrity constraint

$$\forall x\ (\mathsf{Manager}(x) \rightarrow \mathsf{Employee}(x))$$

Why do Databases Work?

query answering problem (as a recognition problem):

given a finite data $\mathcal{D}$, a query $q(\vec{x})$ and a tuple $\vec{a}$,
decide whether $\mathcal{I}_{\mathcal{D}} \models q(\vec{a})$

$\mathcal{I}_{\mathcal{D}}$ makes the facts in $\mathcal{D}$ true (and only them)

what is the complexity of CQ answering?

naive algorithm:

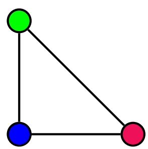
guess values for all existential variables and then
evaluate the query in polynomial time — in **NP**

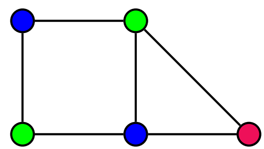
can it be done better?

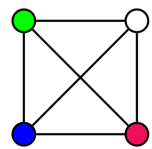
Why Do Databases Work? (2)

no, by reduction of the graph 3-colourability problem, which is **NP-complete**:

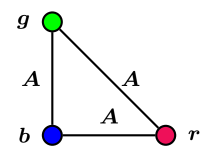
'given an undirected graph $G = (V, E)$,
decide whether it possible to colour it (using r, g, b)
so that no edge has the same colour at both ends?'







$$
\begin{aligned}
\mathcal{D} &= \{A(r,g), A(g,b), A(b,r), A(g,r), A(r,b), A(b,g)\}\\
q_G &= \exists v_1, \dots, v_n \bigwedge_{(v_i,v_j)\in E} A(v_i, v_j)
\end{aligned}
$$



$$\boxed{\mathcal{D} \models q_G \quad \text{iff} \quad G \text{ is 3-colourable}}$$

in fact, the query answering algorithm runs in $O(|\mathcal{D}|^{|q|})$ **data is large, query is short**

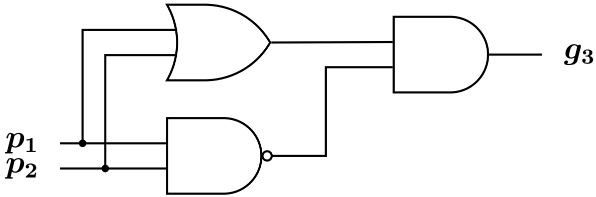
data complexity: only data $\mathcal{D}$ are counted as **input** (q is constant)

(Vardi, 1982): query answering is in AC^0 for data complexity

Circuits and AC^0

a **circuit** is an acyclic graph of AND-, OR- and NOT-gates

(with n inputs and a single output, **sink**)



database instances $\mathcal{D}$ can be encoded on inputs

(one input for each possible ground atom)

FO-query is a circuit: $\land$, $\lor$ and $\neg$ are AND-, OR- and NOT-gates, respectively

$\forall$ and $\exists$ are AND- and OR-gates with unbounded fan-in

AC^0 = circuits of constant depth with AND- and OR-nodes of unbounded fan-in

constant time by a polynomial number of processors (high degree of parallelism)

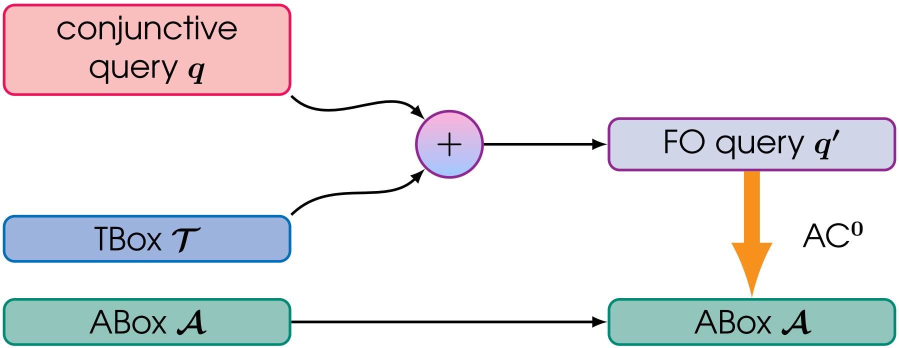
the depth of this circuit does not depend on $\mathcal{D}$ $\longrightarrow$ Vardi's theorem

NB: AC^0 is a **proper** subclass of LogSpace $\subseteq$ P (Parity does not belong to AC^0)

given a word w, decide whether its length is even

Query Rewriting Approach

use off-the-shelf RDBMS (Calvanese et al. ’05)



given a CQ $q(\vec{x})$ over $\mathcal{T}$, rewrite $q(\vec{x})$ into an FO query $q'(\vec{x})$ such that

for all $\mathcal{A}$ and $\vec{a}$, $\mathcal{T}, \mathcal{A} \models q(\vec{a})$ iff $\mathcal{A} \models q'(\vec{a})$

FO-rewritability: only possible in DL with query answering in **FO** (=AC^0) for data complexity:

OWL 2 QL

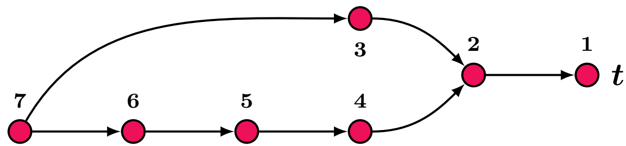
Can We Use $\exists R.A \sqsubseteq B$ in QL?

reachability problem for directed graphs is **NLogSpace-complete**:

'given a directed graph $G = (V, E)$ and $s, t \in V$, decide whether there is a directed path from s to t in G'

ABox: $\mathcal{A}_{G,t} = \{\, \mathsf{edge}(v_1, v_2) \mid (v_1, v_2) \in E \,\} \cup \{\mathsf{ReachableFromTarget}(t)\}$

TBox: $\mathcal{T} = \{\, \exists \mathsf{edge}.\mathsf{ReachableFromTarget} \sqsubseteq \mathsf{ReachableFromTarget} \,\}$



CQ: $q \leftarrow \mathsf{ReachableFromTarget}(s)$

$(\mathcal{T}, \mathcal{A}_{G,t}) \models q$ iff there is a path from s to t in G

$\mathcal{T}$ and q do not depend on G, s, t

→ '$(\mathcal{T}, \mathcal{A}_{G,t}) \models q$?' is **NLogSpace**-hard for data complexity

→ q and $\mathcal{T}$ are not FO-rewritable

Practical Query Answering in OWL 2 QL

systems

- QuOnto (Rome, 2005)
- REQUIEM (Oxford, 2009)
- Presto (Rome, 2010)
- IQAROS (Athens, 2011)
- Rapid (Athens-Oxford, 2011)
- Nyaya (Milan-Oxford, 2010) for TGDs
- Clipper (Vienna, 2012) for Horn-SHIQ
- (Montpellier, 2013) for TGDs
- Quest/ontop (Bolzano, 2011)

not so smoothly: the size of implemented rewritings q' is $O\big((|q| \cdot |\mathcal{T}|)^{|q|}\big)$

(can't say 'query is small or fixed' any longer)

Does a Rewriting Have to be Exponential?

TBox $\boxed{\text{mother} \sqsubseteq \text{parent} \quad \text{and} \quad \text{father} \sqsubseteq \text{parent}}$

query $\boxed{\text{grandparent}(x, z) \leftarrow \text{parent}(x, y) \wedge \text{parent}(y, z)}$

UCQ-rewritings (unions of CQs) are **exponential** in the worst case

$\text{grandparent}(x, z) \leftarrow \text{parent}(x, y) \wedge \text{parent}(y, z)$
$\text{grandparent}(x, z) \leftarrow \text{father}(x, y) \wedge \text{father}(y, z)$
$\text{grandparent}(x, z) \leftarrow \text{mother}(x, y) \wedge \text{father}(y, z)$
$\dots$

PE-rewritings (positive existential queries ≈ select-project-join-union) $\boxed{\exists \ \vee \ \wedge}$

$$\text{grandparent}(x, z) \leftarrow (\text{parent}(x, y) \vee \text{father}(x, y) \vee \text{mother}(x, y)) \wedge \\ (\text{parent}(y, z) \vee \text{father}(y, z) \vee \text{mother}(y, z))$$

NDL-rewriting (non-recursive Datalog ≈ SQL with views) $\boxed{\exists \ \vee \ \wedge + \text{structure sharing}}$

$\text{grandparent}(x, z) \leftarrow \text{ext-parent}(x, y) \wedge \text{ext-parent}(y, z)$
$\text{ext-parent}(x, y) \leftarrow \text{parent}(x, y)$
$\text{ext-parent}(x, y) \leftarrow \text{father}(x, y)$
$\text{ext-parent}(x, y) \leftarrow \text{mother}(x, y)$

FO-rewriting (first-order queries ≈ SQL) $\boxed{\exists \forall \ \vee \ \wedge \neg}$

Case 1: Flat QL TBoxes

a TBox $\mathcal{T}$ is **flat** if it does not contain **generating axioms** $B' \sqsubseteq \exists R.B$

$\approx$ RDF Schema

$q(\vec{x})$ and a flat $\mathcal{T}$ $\longrightarrow$ $q_{\text{ext}}(\vec{x})$ by replacing

$$A(u) \longrightarrow \bigvee_{\mathcal{T} \models A' \sqsubseteq A} A'(u) \lor \bigvee_{\mathcal{T} \models \exists R \sqsubseteq A} \exists v\, R(u,v)$$

$$P(u,v) \longrightarrow \bigvee_{\mathcal{T} \models R \sqsubseteq P} R(u,v)$$

for any CQ $q(\vec{x})$ and any flat OWL 2 QL TBox $\mathcal{T}$,

$q_{\text{ext}}(\vec{x})$ is a PE-rewriting of q and $\mathcal{T}$ of size $O(|q| \cdot |\mathcal{T}|)$

easy in theory, not so in practice

Who Works with Professors?

TBox:

$$\mathsf{worksOn}^- \sqsubseteq \mathsf{involves}$$
$$\mathsf{isManagedBy} \sqsubseteq \mathsf{involves}$$

in English: find those who work with professors

query: $q(x) \leftarrow \mathsf{worksOn}(x, y) \wedge \mathsf{involves}(y, z) \wedge \mathsf{Professor}(z)$

$\mathsf{worksOn}(z, y) \vee \mathsf{isManagedBy}(y, z) \vee \mathsf{involves}(y, z)$

Rewriting over H-complete ABoxes

an ABox $\mathcal{A}$ is **H-complete with respect to $\mathcal{T}$** if

- $A(a) \in \mathcal{A}$ whenever $A'(a) \in \mathcal{A}$ and $\mathcal{T} \models A' \sqsubseteq A$
- $A(a) \in \mathcal{A}$ whenever $R(a, b) \in \mathcal{A}$ and $\mathcal{T} \models \exists R \sqsubseteq A$
- $P(a, b) \in \mathcal{A}$ whenever $R(a, b) \in \mathcal{A}$ and $\mathcal{T} \models R \sqsubseteq P$

an FO-query $q'(\vec{x})$ is an **FO-rewriting of $q(\vec{x})$ and $\mathcal{T}$ over H-complete ABoxes** if,

for any H-complete (w.r.t. $\mathcal{T}$) ABox $\mathcal{A}$ and any $\vec{a}$,

$(\mathcal{T}, \mathcal{A}) \models q(\vec{a})$ iff $\mathcal{A} \models q'(\vec{a})$

(thus we ignore the axioms considered in the flat rewriting)

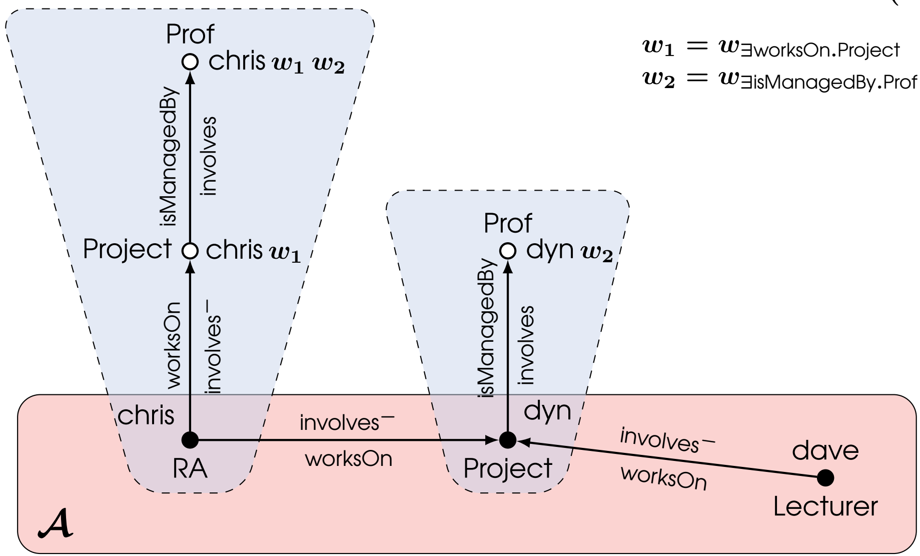
Case 2: Who Works with Professors (2)?

$\mathcal{T}$:

$\mathsf{RA} \sqsubseteq \exists \mathsf{worksOn}.\mathsf{Project}$ $\qquad$ $\mathsf{worksOn}^- \sqsubseteq \mathsf{involves}$

$\mathsf{Project} \sqsubseteq \exists \mathsf{isManagedBy}.\mathsf{Prof}$ $\qquad$ $\mathsf{isManagedBy} \sqsubseteq \mathsf{involves}$

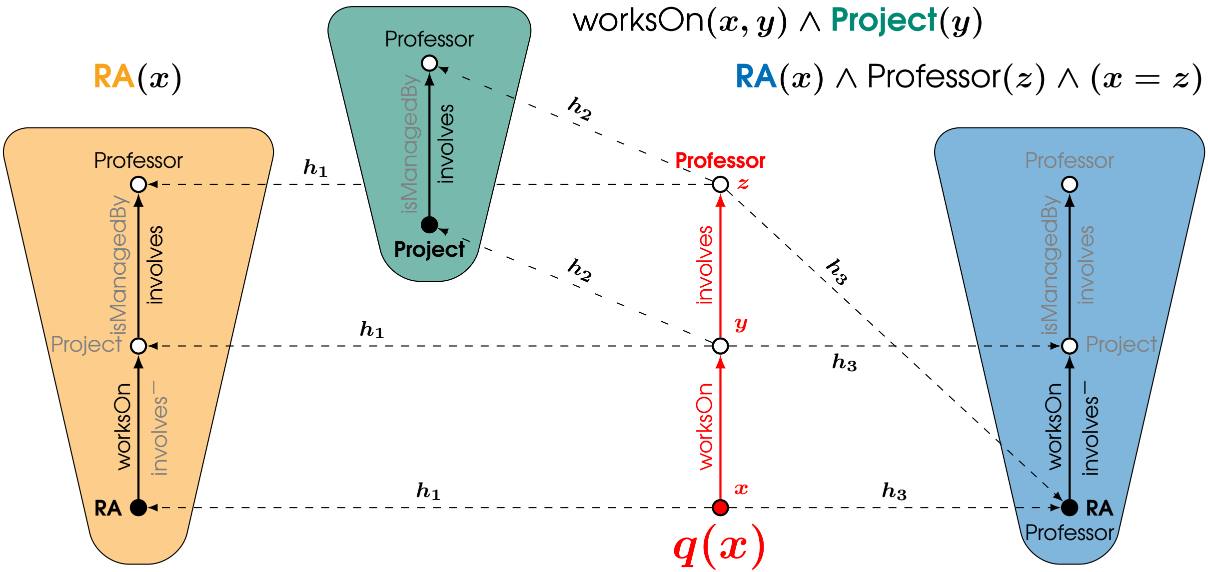
$\mathcal{A}$: RA(chris), worksOn(chris, dyn), Project(dyn), Lecturer(dave), worksOn(dave, dyn)



$w_1 = w_{\exists \mathsf{worksOn}.\mathsf{Project}}$

$w_2 = w_{\exists \mathsf{isManagedBy}.\mathsf{Prof}}$

Case 2: Rewriting the Labelled Nulls



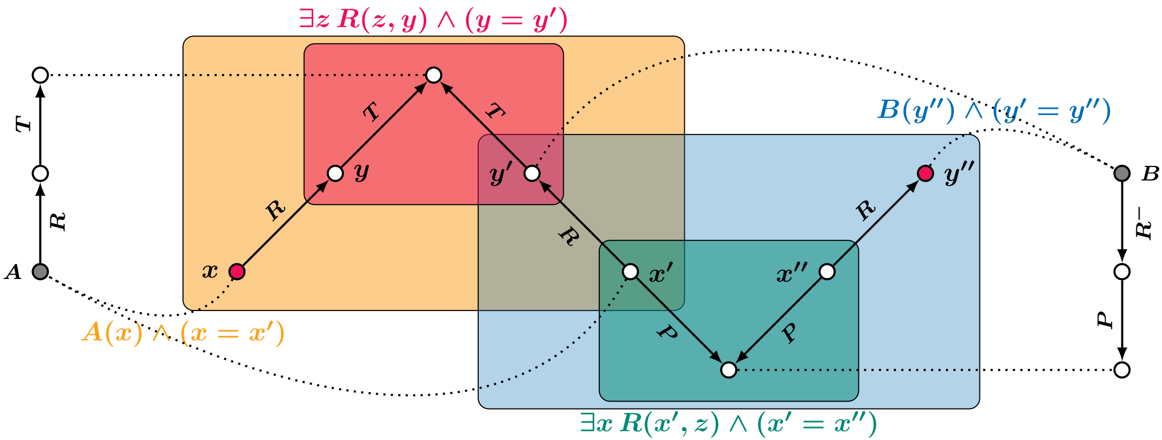
(x and z are the **roots** of the tree witness)

PE-rewriting (over H-complete ABoxes):

$$q'(x) \;\leftarrow\; \mathsf{RA}(x) \;\lor\; (\mathsf{worksOn}(x,y) \land \mathsf{Project}(y)) \;\lor$$
$$(\mathsf{RA}(x) \land \mathsf{Professor}(z) \land (x = z)) \lor$$
$$(\mathsf{worksOn}(x,y) \land \mathsf{involves}(y,z) \land \mathsf{Professor}(z))$$

Tree-Witness Rewriting

TBox $\mathcal{T}$: $\quad A \sqsubseteq \exists R, \quad \exists R^- \sqsubseteq \exists T, \quad B \sqsubseteq \exists R^-, \quad \exists R \sqsubseteq \exists S$



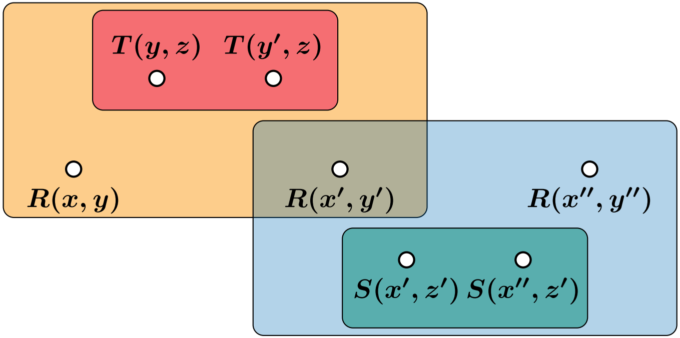
$$q_{\mathsf{tw}}(\vec{x}) \;=\; \bigvee_{\substack{\Theta \text{ independent set}\\ \text{of tree witnesses}}} \exists \vec{y}\Big(\bigwedge_{S(\vec{z}) \in q \setminus q_\Theta} S(\vec{z}) \;\wedge\; \bigwedge_{\mathfrak{t} \in \Theta} \mathsf{tw}_{\mathfrak{t}} \Big)$$

Rewritings as Boolean functions

hypergraph H:

vertices = atoms of the query

hyperedges = tree witnesses



hypergraph function of $H = (V, E)$:

$$f_H = \bigvee_{\substack{X \subseteq E \\ X \text{ independent}}} \Big(\bigwedge_{v \in V \setminus V_X} p_v \wedge \bigwedge_{e \in X} p_e \Big)$$

lower bounds from circuit complexity:

exponential non-recursive datalog (and positive existential) rewritings

superpolynomial first-order rewritings (unless NP $\subseteq$ P/poly)

Short Rewritings in Theory

if $\boldsymbol{q}_{\mathfrak{t}_1} \cap \boldsymbol{q}_{\mathfrak{t}_2} = \emptyset$ or $\boldsymbol{q}_{\mathfrak{t}_1} \subseteq \boldsymbol{q}_{\mathfrak{t}_2}$ or $\boldsymbol{q}_{\mathfrak{t}_2} \subseteq \boldsymbol{q}_{\mathfrak{t}_1}$, for each pair $\underbrace{\mathfrak{t}_1 \text{ and } \mathfrak{t}_2}_{\text{compatible}}$, then

$$q'_{\mathsf{tw}}(\vec{x}) = \bigwedge_{S(\vec{z}) \in q} \Big(S(\vec{z}) \vee \bigvee_{\mathfrak{t}\colon S(\vec{z}) \in q_{\mathfrak{t}}} \mathsf{tw}_{\mathfrak{t}} \Big)$$

is a **rewriting** (over H-complete ABoxes)

EL: add (recursive) datalog rules that 'complete' ABox

$\Longrightarrow$ **polynomial datalog rewriting**

(tree witnesses in $\mathcal{EL}$ are compatible)

QL: replace $S(\vec{z})$ with $\displaystyle\bigvee_{\mathcal{T} \models S' \sqsubseteq S} S'(\vec{z})$

$\Longrightarrow$ **polynomial positive existential rewriting**

provided that the number of tree witnesses is **polynomial** and they are **compatible**

not the case in general!

Part 4

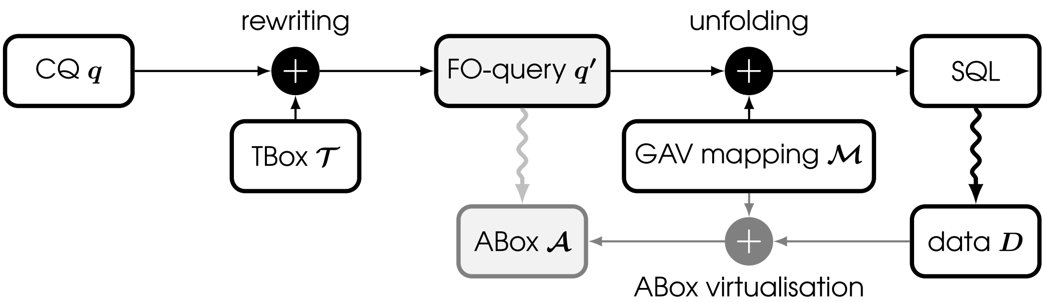
Practical OBDA with Ontop

OBDA system Ontop

http://ontop.inf.unibz.it

- implemented at the Free University of Bozen-Bolzano
  (Mariano Rodríguez-Muro, Martin Rezk, Guohui Xiao)
- open-source
- available as a plugin for Protégé 4, SPARQL end-point,
  OWLAPI and Sesame libraries

OBDA with Databases



Why SQL rewritings are large:

very few for real-world CQs/ontologies

(1) a large number of tree witnesses

(2) large concept/role hierarchies in $\mathcal{T}$

many inclusions in $\mathcal{T}$ follow from Σ and $\mathcal{M}$

(3) multiple definitions of the ontology terms in $\mathcal{M}$

Ontop Example

IMDb (simplified): http://www.imdb.com/interfaces

- database

title

movie ID	title	production year
728	'Django Unchained'	2012
...	...	...

castinfo

person ID	movie ID	person role
n37	728	1
n38	728	1
...	...	...

- dependencies

$$\forall m\ (\exists p, r\ \mathit{castinfo}(p, m, r) \rightarrow \exists t, y\ \mathit{title}(m, t, y)) \qquad \text{(FK)}$$
$$\forall m \forall t_1 \forall t_2\ (\exists y\ \mathit{title}(m, t_1, y) \wedge \exists y\ \mathit{title}(m, t_2, y) \rightarrow (t_1 = t_2)) \qquad (\text{PK}_1)$$
$$\forall m \forall y_1 \forall y_2\ (\exists t\ \mathit{title}(m, t, y_1) \wedge \exists t\ \mathit{title}(m, t, y_2) \rightarrow (y_1 = y_2)) \qquad (\text{PK}_2)$$

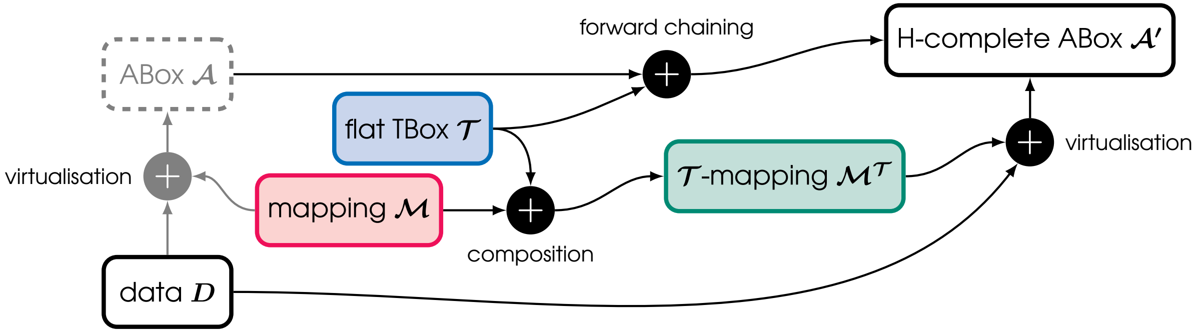
Movie Ontology MO http://www.movieontology.org

$$\mathit{mo{:}Movie} \equiv \exists \mathit{mo{:}title}, \qquad \mathit{mo{:}Movie} \sqsubseteq \exists \mathit{mo{:}year},$$
$$\mathit{mo{:}Movie} \equiv \exists \mathit{mo{:}cast}, \qquad \exists \mathit{mo{:}cast}^- \sqsubseteq \mathit{mo{:}Person}, \ldots$$

Mapping (created by the Ontop development team)

$$\mathit{mo{:}Movie}(m), \mathit{mo{:}title}(m, t), \mathit{mo{:}year}(m, y) \leftarrow \mathit{title}(m, t, y) \qquad (\text{M}_1)$$
$$\mathit{mo{:}cast}(m, p), \mathit{mo{:}Person}(p) \leftarrow \mathit{castinfo}(p, m, r) \qquad (\text{M}_2)$$

Ontop: T-mappings



$\mathcal{T}$	$mo{:}Movie \equiv \exists mo{:}title$, $\quad mo{:}Movie \sqsubseteq \exists mo{:}year$, $mo{:}Movie \equiv \exists mo{:}cast$, $\quad \exists mo{:}cast^- \sqsubseteq mo{:}Person$
$\mathcal{M}$	$mo{:}Movie(\boldsymbol{m}), mo{:}title(\boldsymbol{m}, \boldsymbol{t}), mo{:}year(\boldsymbol{m}, \boldsymbol{y}) \leftarrow title(\boldsymbol{m}, \boldsymbol{t}, \boldsymbol{y})$ (M$_1$) $mo{:}cast(\boldsymbol{m}, \boldsymbol{p}), mo{:}Person(\boldsymbol{p}) \leftarrow castinfo(\boldsymbol{p}, \boldsymbol{m}, \boldsymbol{r})$ (M$_2$)
$\mathcal{M}^{\mathcal{T}}$	$mo{:}Movie(\boldsymbol{m}) \leftarrow title(\boldsymbol{m}, \boldsymbol{t}, \boldsymbol{y})$ by (M$_1$) $\cancel{mo{:}Movie(\boldsymbol{m}) \leftarrow castinfo(\boldsymbol{p}, \boldsymbol{m}, \boldsymbol{r})}$ by (M$_2$) + $\exists mo{:}cast \sqsubseteq mo{:}Movie$

redundant by (FK)

$\forall m\ (\exists p, r\ \textbf{castinfo}(p, m, r) \rightarrow \exists t, y\ \textbf{title}(m, t, y))$

Optimising T-mappings

- using foreign keys (inclusion dependencies)

- using disjunction

$\mathcal{T}$

$$\textit{mo:Actor} \sqsubseteq \textit{mo:Artist}, \qquad \textit{mo:Artist} \sqsubseteq \textit{mo:Person},$$
$$\textit{mo:Director} \sqsubseteq \textit{mo:Person}, \qquad \textit{mo:Editor} \sqsubseteq \textit{mo:Person}, \dots$$

$\mathcal{M}$

$$\textit{mo:Actor}(\boldsymbol{p}) \leftarrow \textit{castinfo}(\boldsymbol{p}, \boldsymbol{m}, \boldsymbol{r}), (\boldsymbol{r} = \mathbf{1}) \qquad (\mathrm{M}_1)$$
$$\dots$$
$$\textit{mo:Editor}(\boldsymbol{p}) \leftarrow \textit{castinfo}(\boldsymbol{p}, \boldsymbol{m}, \boldsymbol{r}), (\boldsymbol{r} = \mathbf{6}) \qquad (\mathrm{M}_6)$$

$\mathcal{M}^{\mathcal{T}}$

$$\textit{mo:Person}(\boldsymbol{p}) \leftarrow \textit{castinfo}(\boldsymbol{p}, \boldsymbol{m}, \boldsymbol{r}), ((\boldsymbol{r} = \mathbf{1}) \lor \cdots \lor (\boldsymbol{r} = \mathbf{6}))$$

Unfolding with Semantic Query Optimisation

Query

$$q(t, y) \leftarrow \textit{mo:Movie}(m), \textit{mo:title}(m, t), \textit{mo:year}(m, y), (y > 2010)$$

Rewriting

$$q'(t, y) \leftarrow \textit{mo:Movie}(m), \textit{mo:title}(m, t), \textit{mo:year}(m, y), (y > 2010)$$

$\mathcal{M}$

$\textit{mo:Movie}(m) \leftarrow \textit{title}(m, t, y)$	(M_1)
$\textit{mo:title}(m, t) \leftarrow \textit{title}(m, t, y)$	(M_2)
$\textit{mo:year}(m, y) \leftarrow \textit{title}(m, t, y)$	(M_3)

Unfolding

$$q^*(t, y) \leftarrow \textit{title}(m, t_0, y_0), \textit{title}(m, t, y_1), \textit{title}(m, t_2, y), (y > 2010)$$

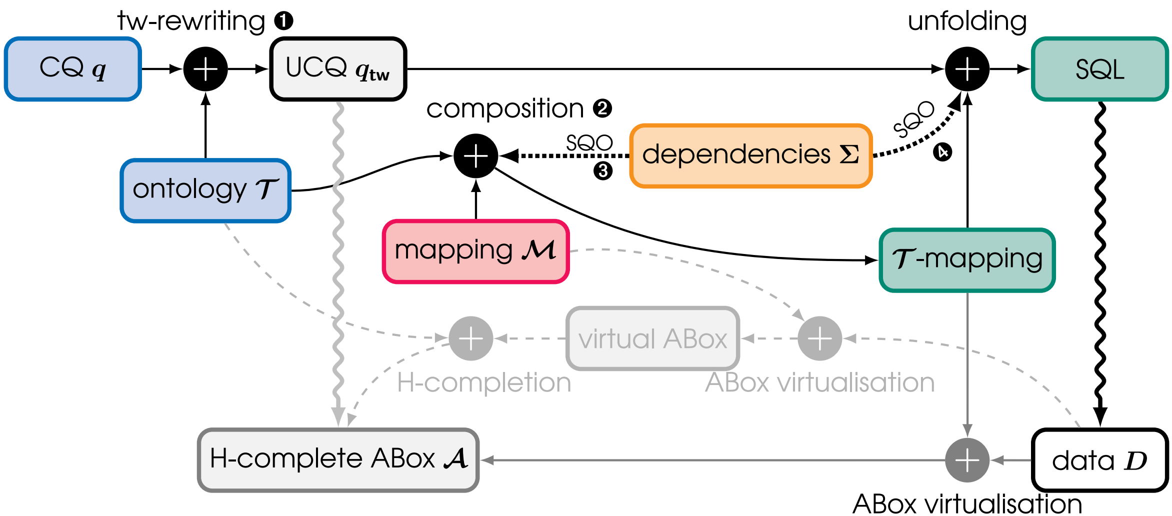
primary keys

$\forall m \forall t_1 \forall t_2 \left(\exists y\, \textit{title}(m, t_1, y) \wedge \exists y\, \textit{title}(m, t_2, y) \rightarrow (t_1 = t_2)\right)$	(PK_1)
$\forall m \forall y_1 \forall y_2 \left(\exists t\, \textit{title}(m, t, y_1) \wedge \exists t\, \textit{title}(m, t, y_2) \rightarrow (y_1 = y_2)\right)$	(PK_2)

Semantic Query Optimisation

$$q^\dagger(t, y) \leftarrow \textit{title}(m, t, y), (y > 2010)$$

Practical OBDA with Ontop



❶ tree-witness rewriting q_{tw} over H-complete ABoxes (no concept/role hierarchies)

❷ $\mathcal{T}$-mapping $=$ system mapping $\mathcal{M} + \mathcal{T}$ makes virtual ABoxes H-complete

❸ $\mathcal{T}$-mapping is simplified using SQO and SQL features

constructed and optimised for $\mathcal{T}$ and Σ only once

❹ unfolding uses SQO to produce small and efficient SQL queries

Recommended Reading (1)

A. Poggi, D. Lembo, D. Calvanese, G. De Giacomo, M. Lenzerini and R. Rosati. Linking Data to Ontologies. J. Data Semantics 10: 133–173 (2008)

A. Artale, D. Calvanese, R. Kontchakov and M. Zakharyaschev: The DL-Lite Family and Relations. J. Artif. Intell. Res. (JAIR) 36:1–69 (2009)

A. Calì, G. Gottlob and A. Pieris. Query Rewriting under Non-Guarded Rules. In Proc. AMW 2010

M. Vardi. The Complexity of Relational Query Languages (Extended Abstract). In Proc. STOC 1982: 137–146

D. Maier, A. O. Mendelzon, and Y. Sagiv. Testing implications of data dependencies. ACM Trans. Database Syst., 4(4):455–469, 1979

D. S. Johnson and A. C. Klug. Testing containment of conjunctive queries under functional and inclusion dependencies. J. Comput. Syst. Sci., 28(1):167–189, 1984

A. Deutsch, A. Nash, and J. B. Remmel. The chase revisited. Proc. PODS 2008: 149–158

M. Rodriguez-Muro and D. Calvanese. Semantic Index: Scalable Query Answering without Forward Chaining or Exponential Rewritings Posters of ISWC 2011

M. Rodriguez-Muro and D. Calvanese. Dependencies: Making Ontology Based Data Access Work. Proc. AMW 2011

G. Gottlob and T. Schwentick. Rewriting Ontological Queries into Small Nonrecursive Datalog Programs. Proc. KR 2012

Recommended Reading (2)

S. Kikot, R. Kontchakov, V. Podolskii and M. Zakharyaschev: Exponential Lower Bounds and Separation for Query Rewriting. Proc. ICALP (2) 2012: 263–274

F. Baader, S. Brandt and C. Lutz. Pushing the EL envelope. Proc. IJCAI 2005

B. N. Grosof, I. Horrocks, R. Volz and S. Decker. Description logic programs: Combining logic programs with description logic. Proc. WWW 2003

A. Calì, G. Gottlob and A. Pieris. Advanced Processing for Ontological Queries. PVLDB 3(1): 554–565 (2010)

D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini and R. Rosati. DL-Lite: Tractable Description Logics for Ontologies. Proc. AAAI 2005: 602–607

A. Chortaras, D. Trivela and G. Stamou. Optimized query rewriting for OWL 2 QL. Proc. CADE 2011

T. Eiter, Ortiz, M. Šimkus, T.-K. Tran and G. Xiao. Query rewriting for Horn-SHIQ plus rules. Proc. AAAI 2012

M. König, M. Leclère, M.-L. Mugnier and M. Thomazo: A sound and complete backward chaining algorithm for existential rules. Proc. RR 2012

M. Rodríguez-Muro, R. Kontchakov and M. Zakharyaschev. Ontology-Based Data Access: Ontop of Databases. Proc. ISWC 2013

R. Kontchakov, M. Rezk, M. Rodríguez-Muro, G. Xiao and M. Zakharyaschev. Answering SPARQL Queries under the OWL 2 QL Entailment Regime with Databases. Proc. ISWC 2014