Ontology-based Data Access: Theory and Practice

Guohui Xiao

KRDB Research Centre

Free University of Bozen-Bolzano

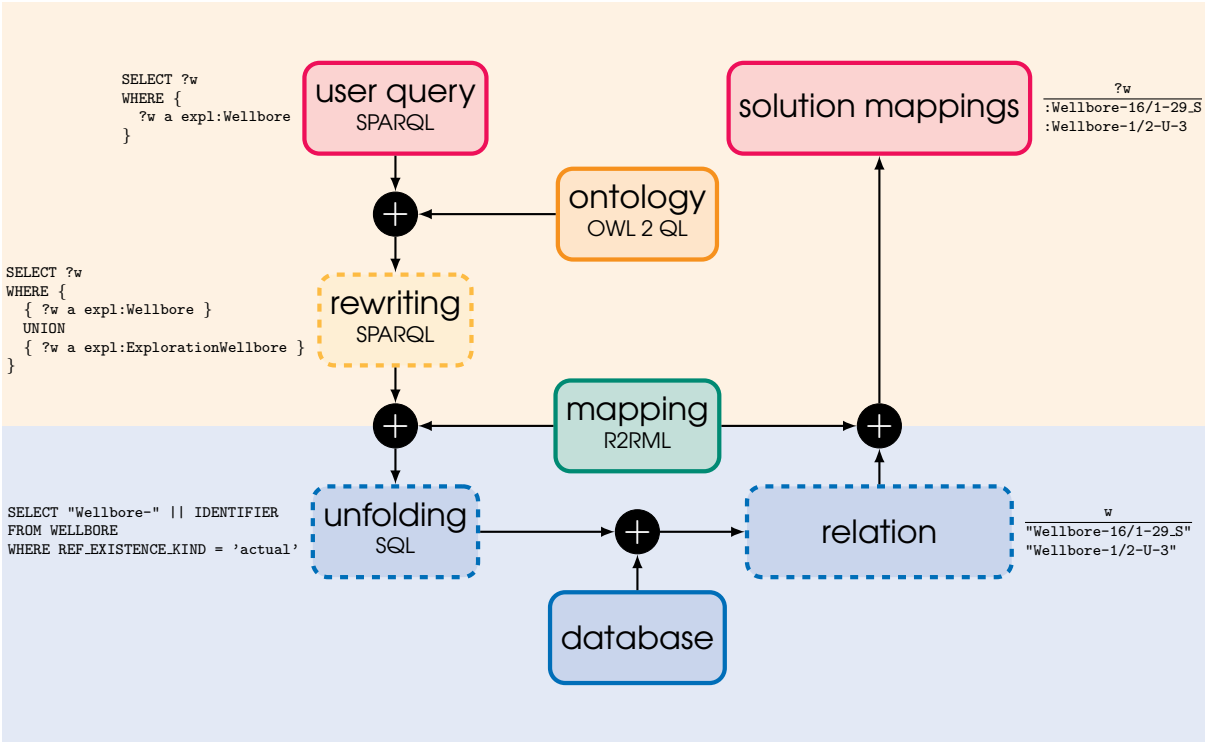
Roman Kontchakov

Department of Computer Science & Inf. Systems

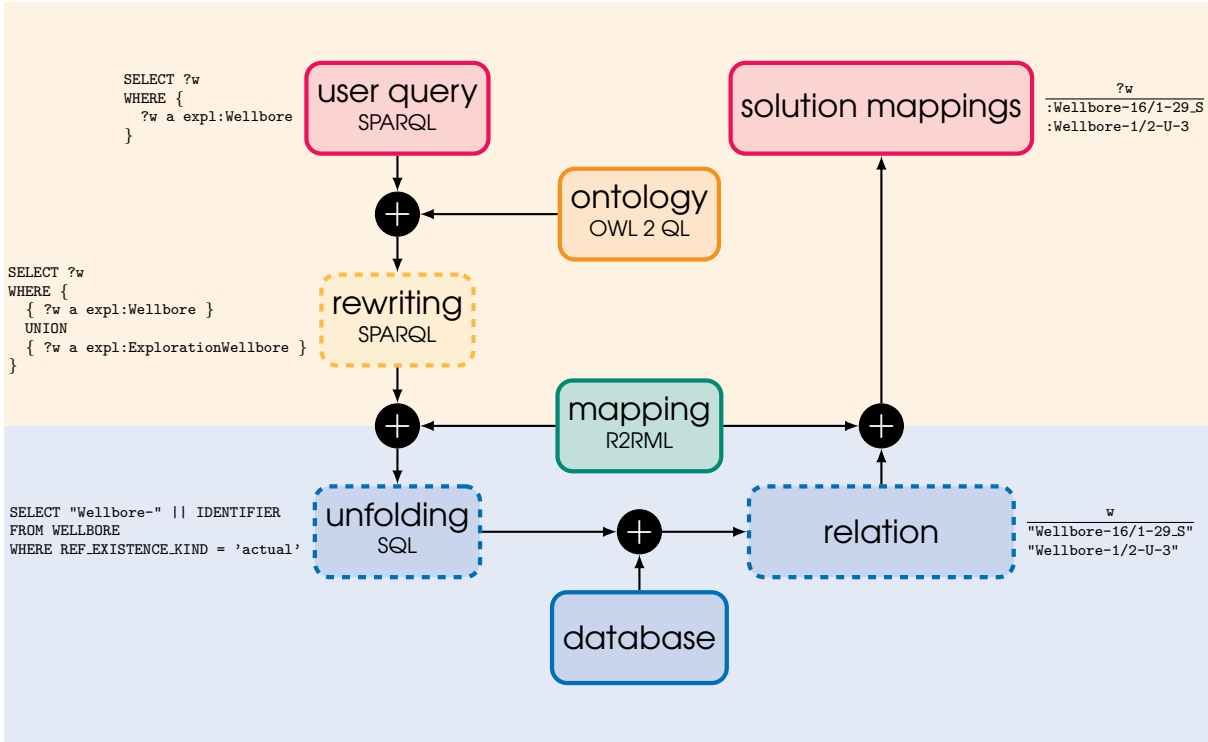
Birkbeck, University of London

http://ontop.inf.unibz.it/ijcai-2018-tutorial

Query Answering in OBDA



Query Answering in OBDA



no ETL (Extract Transform Load)

Formalising OBDA

an **OBDA specification** $\mathcal{P} = (\mathcal{O}, \mathcal{M}, \mathcal{S})$

- $\mathcal{O}$ an ontology (class and property inclusions / tgds)
- $\mathcal{M}$ a mapping (assertions of the form $\varphi(\boldsymbol{x}) \rightsquigarrow \psi(\boldsymbol{x})$)
- $\mathcal{S}$ a data source schema with integrity constraints

Formalising OBDA

an **OBDA specification** $\mathcal{P} = (\mathcal{O}, \mathcal{M}, \mathcal{S})$

- $\mathcal{O}$ an ontology (class and property inclusions / tgds)
- $\mathcal{M}$ a mapping (assertions of the form $\varphi(\boldsymbol{x}) \rightsquigarrow \psi(\boldsymbol{x})$)
- $\mathcal{S}$ a data source schema with integrity constraints

a **source database** $\mathcal{D}$ (relations conforming to $\mathcal{S}$)

an **OBDA instance** $(\mathcal{P}, \mathcal{D})$

Formalising OBDA

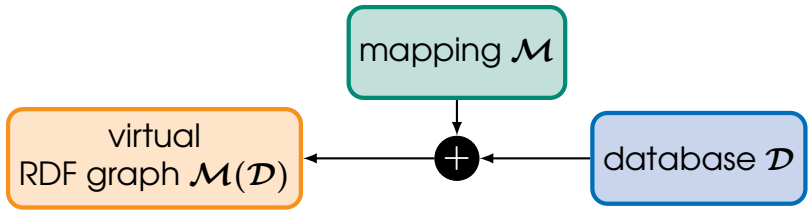
an **OBDA specification** $\mathcal{P} = (\mathcal{O}, \mathcal{M}, \mathcal{S})$

- $\mathcal{O}$ an ontology (class and property inclusions / tgds)
- $\mathcal{M}$ a mapping (assertions of the form $\varphi(\boldsymbol{x}) \rightsquigarrow \psi(\boldsymbol{x})$)
- $\mathcal{S}$ a data source schema with integrity constraints

a **source database** $\mathcal{D}$ (relations conforming to $\mathcal{S}$)

an **OBDA instance** $(\mathcal{P}, \mathcal{D})$

$\mathcal{M}(\mathcal{D})$ is a set of atoms in the signature of $\mathcal{O}$



Formalising OBDA

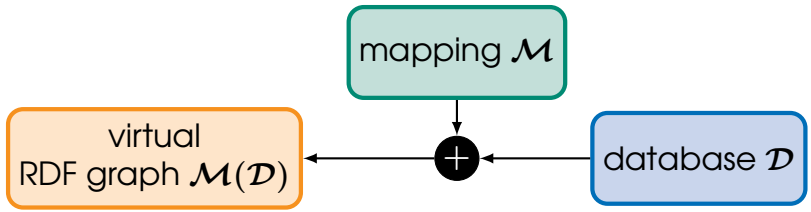
an **OBDA specification** $\mathcal{P} = (\mathcal{O}, \mathcal{M}, \mathcal{S})$

- $\mathcal{O}$ an ontology (class and property inclusions / tgds)
- $\mathcal{M}$ a mapping (assertions of the form $\varphi(\boldsymbol{x}) \rightsquigarrow \psi(\boldsymbol{x})$)
- $\mathcal{S}$ a data source schema with integrity constraints

a **source database** $\mathcal{D}$ (relations conforming to $\mathcal{S}$)

an **OBDA instance** $(\mathcal{P}, \mathcal{D})$

$\mathcal{M}(\mathcal{D})$ is a set of atoms in the signature of $\mathcal{O}$



$\mathcal{I}$ is a model of $(\mathcal{P}, \mathcal{D})$ if it satisfies all axioms in $\mathcal{O}$ and contains all $\mathcal{M}(\mathcal{D})$

Formalising OBDA

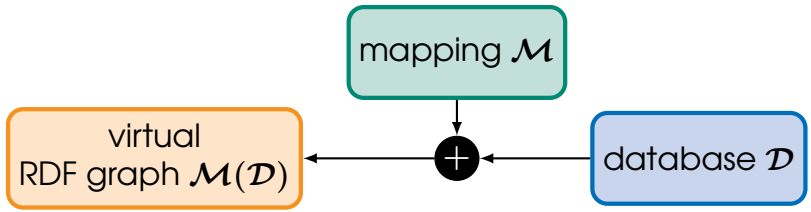
an **OBDA specification** $\mathcal{P} = (\mathcal{O}, \mathcal{M}, \mathcal{S})$

- $\mathcal{O}$ an ontology (class and property inclusions / tgds)
- $\mathcal{M}$ a mapping (assertions of the form $\varphi(\boldsymbol{x}) \rightsquigarrow \psi(\boldsymbol{x})$)
- $\mathcal{S}$ a data source schema with integrity constraints

a **source database** $\mathcal{D}$ (relations conforming to $\mathcal{S}$)

an **OBDA instance** $(\mathcal{P}, \mathcal{D})$

$\mathcal{M}(\mathcal{D})$ is a set of atoms in the signature of $\mathcal{O}$

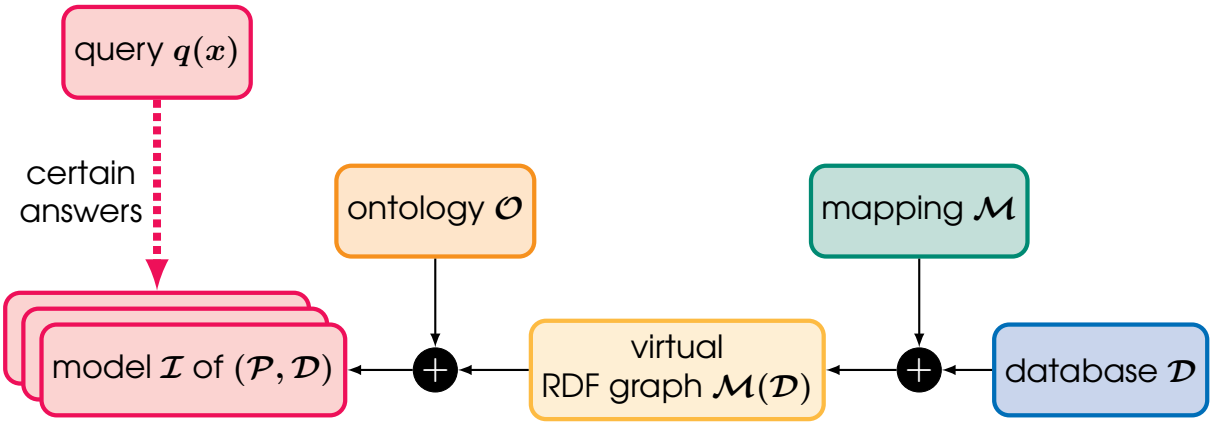


$\mathcal{I}$ is a model of $(\mathcal{P}, \mathcal{D})$ if it satisfies all axioms in $\mathcal{O}$ and contains all $\mathcal{M}(\mathcal{D})$

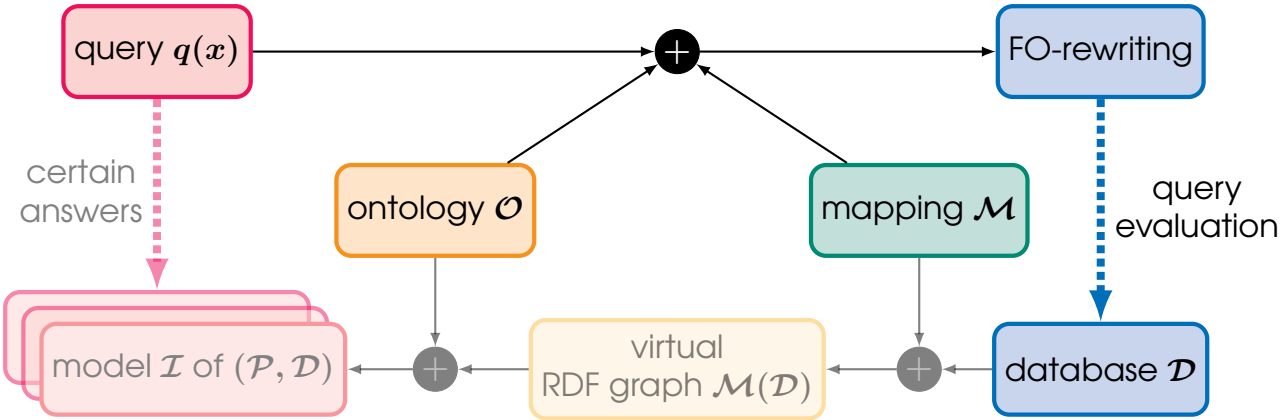
a tuple $\boldsymbol{a}$ of constants in $\mathcal{D}$ is a **certain answer** to a query $q(\boldsymbol{x})$ over $(\mathcal{P}, \mathcal{D})$ if

$\mathcal{I} \models q(\boldsymbol{a})$ for every model $\mathcal{I}$ of $(\mathcal{P}, \mathcal{D})$

OBDA Query Answering



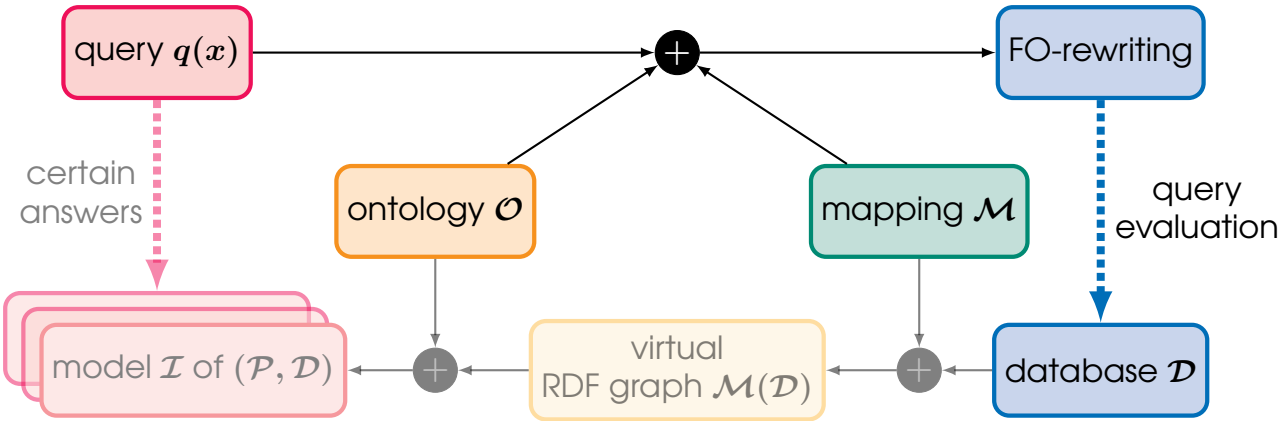
OBDA Query Answering



$q'(x)$ is an **FO-rewriting** of $q(x)$ with respect to $\mathcal{P}$ if, for every $\mathcal{D}$,

certain answers to $q(x)$ over $(\mathcal{P}, \mathcal{D})$ = answers to $q'(x)$ over $\mathcal{D}$

OBDA Query Answering

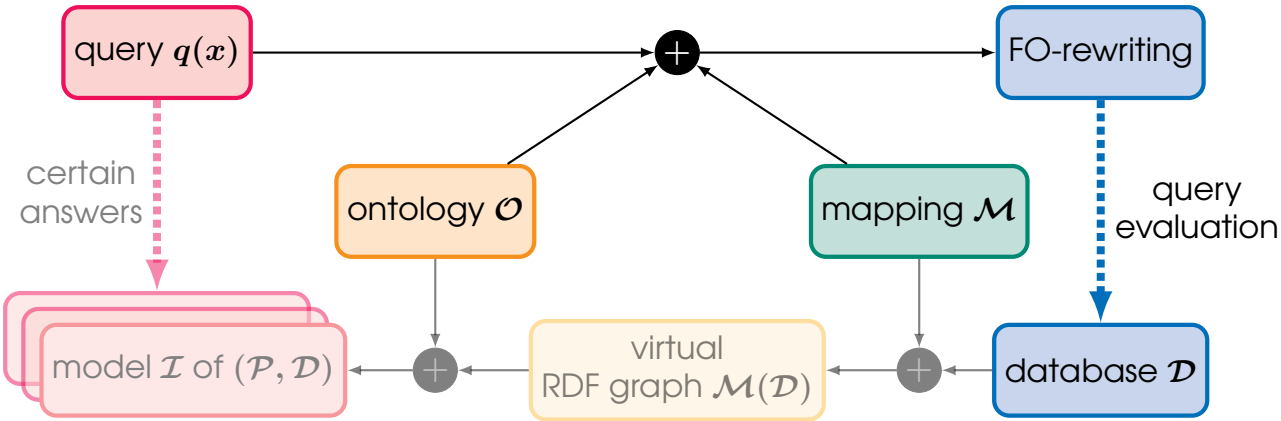


$q'(x)$ is an **FO-rewriting** of $q(x)$ with respect to $\mathcal{P}$ if, for every $\mathcal{D}$,

certain answers to $q(x)$ over $(\mathcal{P}, \mathcal{D})$ = answers to $q'(x)$ over $\mathcal{D}$

evaluating a **fixed** FO-query over databases $\mathcal{D}$ is in AC^0 (**data complexity**)

OBDA Query Answering



$q'(x)$ is an **FO-rewriting** of $q(x)$ with respect to $\mathcal{P}$ if, for every $\mathcal{D}$,

certain answers to $q(x)$ over $(\mathcal{P}, \mathcal{D})$ = answers to $q'(x)$ over $\mathcal{D}$

evaluating a **fixed** FO-query over databases $\mathcal{D}$ is in AC^0 (**data complexity**)

so, the **certain answers problem** should also be in AC^0 for data complexity

NB: AC^0 = circuits of constant depth with AND/OR gates of unbounded fan-in

Limits of OWL 2 QL

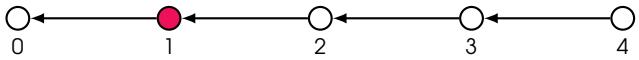
the **certain answers problem** should also be in AC^0 for data complexity

Limits of OWL 2 QL

the **certain answers problem** should also be in AC^0 for data complexity

1 ontology $\{\exists R.A \sqsubseteq A\}$ $\Rightarrow$ **NL-hard** certain answers problem for $A(x)$
(in data complexity)

R is a directed graph,
and A 'labels' all vertices inverse-reachable in R from other A-labelled vertices

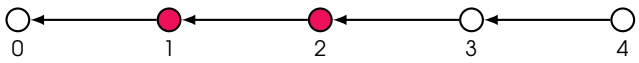


Limits of OWL 2 QL

the **certain answers problem** should also be in AC^0 for data complexity

1 ontology $\{\exists R.A \sqsubseteq A\}$ $\Rightarrow$ **NL-hard** certain answers problem for $A(x)$
(in data complexity)

R is a directed graph,
and A 'labels' all vertices inverse-reachable in R from other A-labelled vertices



Limits of OWL 2 QL

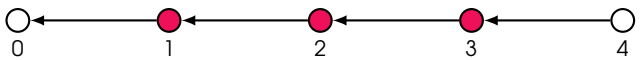
the **certain answers problem** should also be in AC^0 for data complexity

1 ontology $\{\exists \boldsymbol{R}.\boldsymbol{A} \sqsubseteq \boldsymbol{A}\}$ $\Rightarrow$ **NL-hard** certain answers problem for $\boldsymbol{A}(\boldsymbol{x})$

(in data complexity)

$\boldsymbol{R}$ is a directed graph,

and $\boldsymbol{A}$ 'labels' all vertices inverse-reachable in $\boldsymbol{R}$ from other $\boldsymbol{A}$-labelled vertices

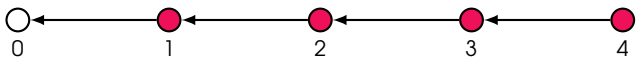


Limits of OWL 2 QL

the **certain answers problem** should also be in AC^0 for data complexity

1 ontology $\{\exists R.A \sqsubseteq A\}$ $\Rightarrow$ **NL-hard** certain answers problem for $A(x)$
(in data complexity)

R is a directed graph,
and A 'labels' all vertices inverse-reachable in R from other A-labelled vertices

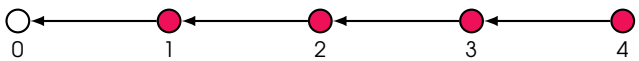


Limits of OWL 2 QL

the **certain answers problem** should also be in AC^0 for data complexity

1 ontology $\{\exists R.A \sqsubseteq A\}$ $\Rightarrow$ **NL-hard** certain answers problem for $A(x)$ (in data complexity)

R is a directed graph,
and A 'labels' all vertices inverse-reachable in R from other A-labelled vertices



2 ontology $\{\exists R_1.A \sqcap \exists R_2.A \sqsubseteq A\}$ $\Rightarrow$ **PTIME-hard** certain answers problem for $A(x)$ (in data complexity)

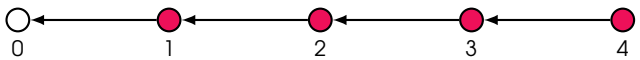
Circuit Value Problem or Path System Accessibility

Limits of OWL 2 QL

the **certain answers problem** should also be in AC^0 for data complexity

1 ontology $\{\exists R.A \sqsubseteq A\}$ $\Rightarrow$ **NL-hard** certain answers problem for $A(x)$ (in data complexity)

R is a directed graph,
and A 'labels' all vertices inverse-reachable in R from other A-labelled vertices



2 ontology $\{\exists R_1.A \sqcap \exists R_2.A \sqsubseteq A\}$ $\Rightarrow$ **PTIME-hard** certain answers problem for $A(x)$ (in data complexity)

Circuit Value Problem or Path System Accessibility

3 ontology $\{A \sqsubseteq A_1 \sqcup A_2\}$ $\Rightarrow$ **coNP-hard** certain answers problem for q_{2+2} (in data complexity)

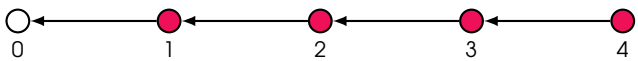
Boolean Satisfiability (2+2CNF)

Limits of OWL 2 QL

the **certain answers problem** should also be in AC^0 for data complexity

1 ontology $\{\exists R.A \sqsubseteq A\}$ $\Rightarrow$ **NL-hard** certain answers problem for $A(x)$ (in data complexity)

R is a directed graph,
and A 'labels' all vertices inverse-reachable in R from other A-labelled vertices



2 ontology $\{\exists R_1.A \sqcap \exists R_2.A \sqsubseteq A\}$ $\Rightarrow$ **PTIME-hard** certain answers problem for $A(x)$ (in data complexity)

Circuit Value Problem or Path System Accessibility

3 ontology $\{A \sqsubseteq A_1 \sqcup A_2\}$ $\Rightarrow$ **coNP-hard** certain answers problem for q_{2+2} (in data complexity)

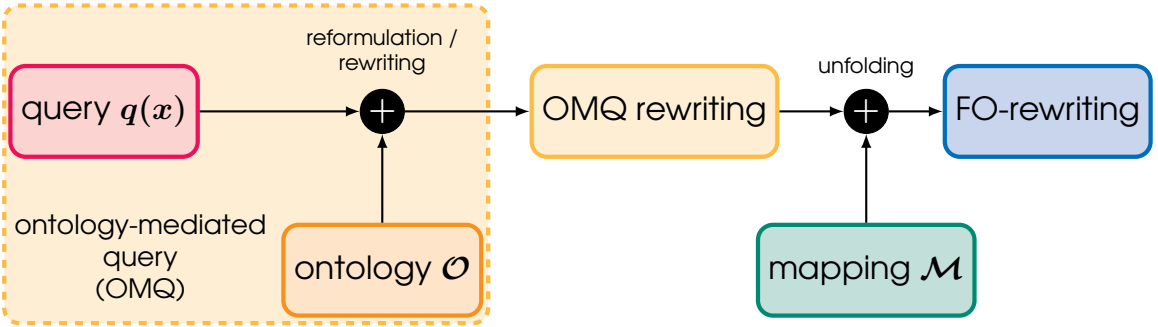
Boolean Satisfiability (2+2CNF)

4 sameAs(a, b) $\Rightarrow$ **L-hard** certain answers problem for $A(x)$ (in data complexity)

NB: $AC^0 \subsetneq L \subseteq NL \subseteq PTIME \subseteq coNP$

OBDA Query Answering in Practice

split construction of FO-rewritings into two separate steps:



Rewriting OMQs with Horn Ontologies

let $\mathcal{O}$ contain

$\mathsf{RA} \sqsubseteq \exists \mathsf{worksOn}.\mathsf{Project}$

$\mathsf{Project} \sqsubseteq \exists \mathsf{isManagedBy}.\mathsf{Professor}$

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

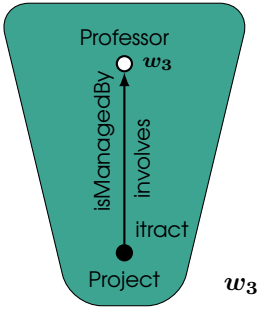
$\mathsf{isManagedBy} \sqsubseteq \mathsf{involves}$

Rewriting OMQs with Horn Ontologies

let $\mathcal{O}$ contain

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

if $\mathcal{D}$ contains Project(itract), then **any model** of $(\mathcal{O}, \mathcal{D})$ will have a fragment similar to



w_3 is a **labelled null**

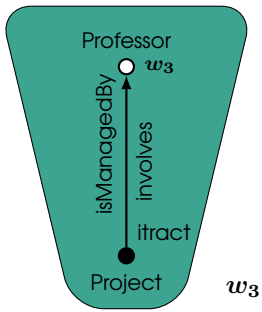
Rewriting OMQs with Horn Ontologies

let $\mathcal{O}$ contain

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

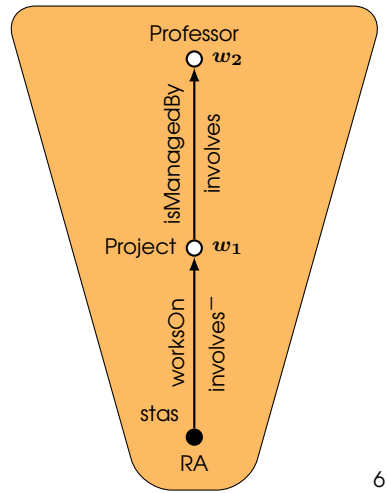
$\mathsf{Project} \sqsubseteq \exists \mathsf{isManagedBy}.\mathsf{Professor}$ $\quad$ $\mathsf{isManagedBy} \sqsubseteq \mathsf{involves}$

if $\mathcal{D}$ contains Project(itract), then **any model** of $(\mathcal{O}, \mathcal{D})$ will have a fragment similar to



w_3 is a **labelled null**

if $\mathcal{D}$ contains RA(stas), then any model of $(\mathcal{O}, \mathcal{D})$ will have a fragment similar to

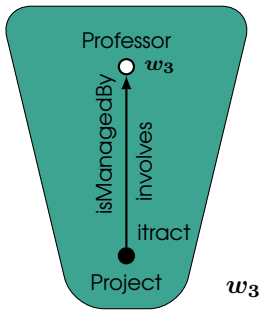


Rewriting OMQs with Horn Ontologies

let $\mathcal{O}$ contain

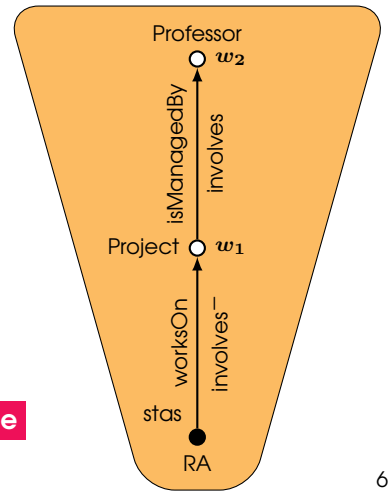
RA $\sqsubseteq \exists$worksOn.Project	worksOn$^- \sqsubseteq$ involves
Project $\sqsubseteq \exists$isManagedBy.Professor	isManagedBy $\sqsubseteq$ involves

if $\mathcal{D}$ contains Project(itract), then **any model** of $(\mathcal{O}, \mathcal{D})$ will have a fragment similar to



w_3 is a **labelled null**

if $\mathcal{D}$ contains RA(stas), then any model of $(\mathcal{O}, \mathcal{D})$ will have a fragment similar to



if $\mathcal{O}$ is Horn (does not contain any disjunctions), then

certain answers to $q(x)$ over $(\mathcal{O}, \mathcal{D})$ =
answers to $q(x)$ over $\mathfrak{C}_{\mathcal{O}}(\mathcal{D})$
the **canonical model / chase**

Tree-Witness Query Rewriting: Example

$$\boldsymbol{q}(\boldsymbol{x}) \;=\; \exists \boldsymbol{y}, \boldsymbol{z}\,\big(\mathsf{worksOn}(\boldsymbol{x}, \boldsymbol{y}) \;\wedge\; \mathsf{involves}(\boldsymbol{y}, \boldsymbol{z}) \;\wedge\; \mathsf{Professor}(\boldsymbol{z})\big)$$

Tree-Witness Query Rewriting: Example

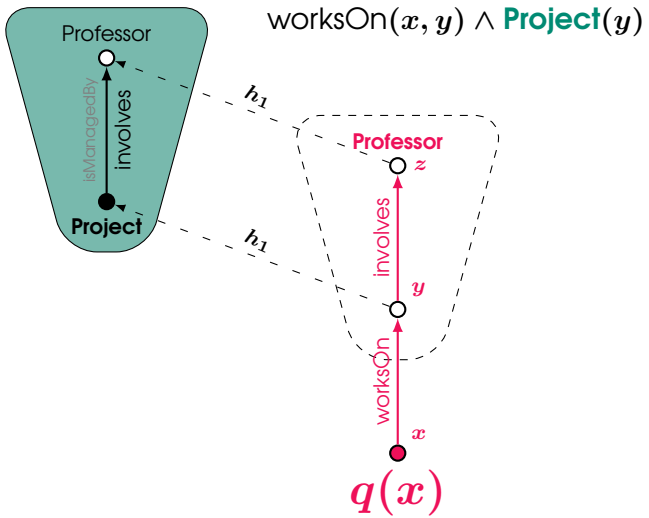
$$q(x) \;=\; \exists y, z \,\big(\mathsf{worksOn}(x, y) \;\wedge\; \mathsf{involves}(y, z) \;\wedge\; \mathsf{Professor}(z)\big)$$

a **tree witness** $\approx$ a query fragment embeddable into a tree of labelled nulls
such that only its 'boundary' (join) variables may be answer variables

Tree-Witness Query Rewriting: Example

$q(x) \;=\; \exists y, z\,(\mathsf{worksOn}(x, y) \;\wedge\; \mathsf{involves}(y, z) \;\wedge\; \mathsf{Professor}(z))$

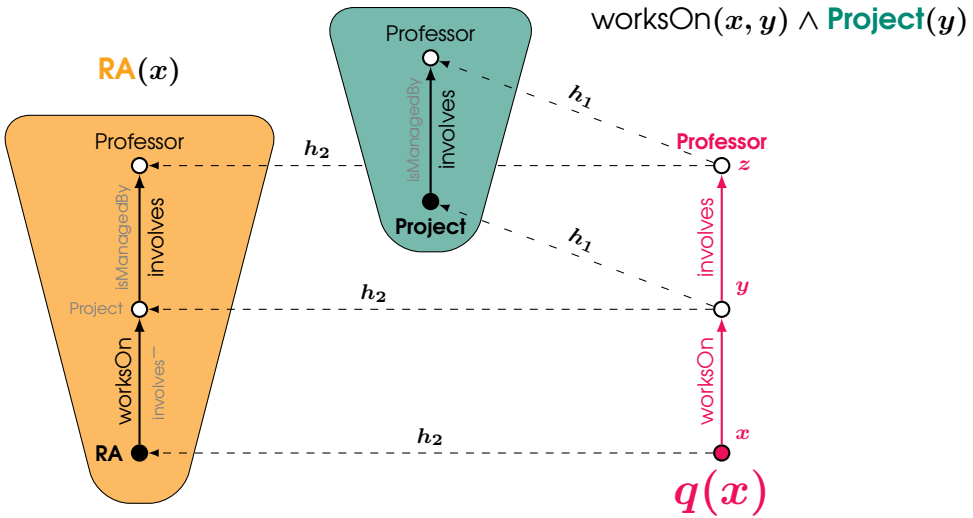
a **tree witness** $\approx$ a query fragment embeddable into a tree of labelled nulls such that only its 'boundary' (join) variables may be answer variables



Tree-Witness Query Rewriting: Example

$q(x) = \exists y, z\,(\mathsf{worksOn}(x, y) \wedge \mathsf{involves}(y, z) \wedge \mathsf{Professor}(z))$

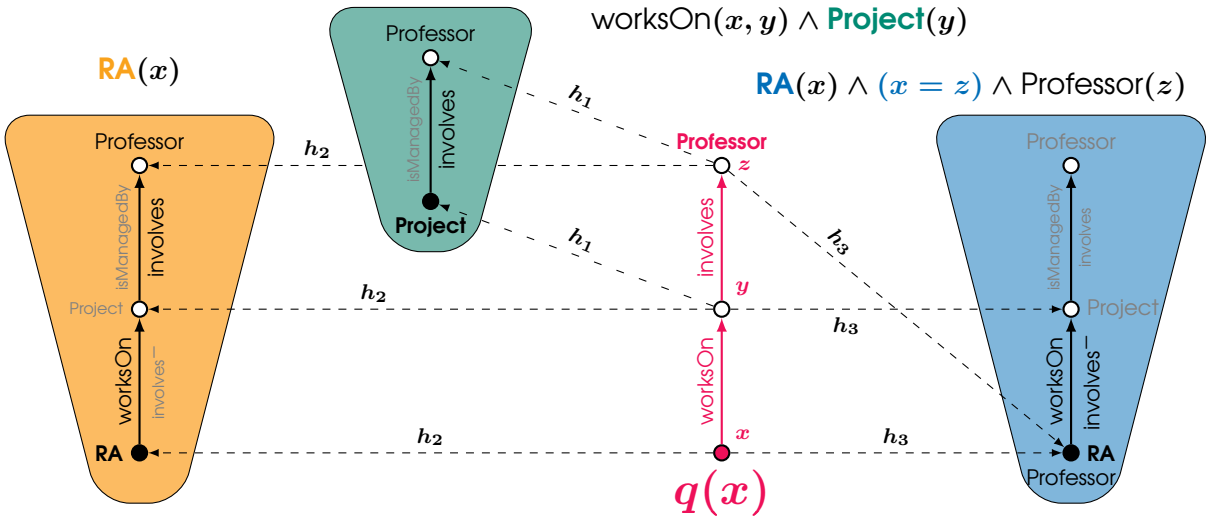
a **tree witness** $\approx$ a query fragment embeddable into a tree of labelled nulls such that only its 'boundary' (join) variables may be answer variables



Tree-Witness Query Rewriting: Example

$q(x) = \exists y, z\,(\text{worksOn}(x, y) \wedge \text{involves}(y, z) \wedge \text{Professor}(z))$

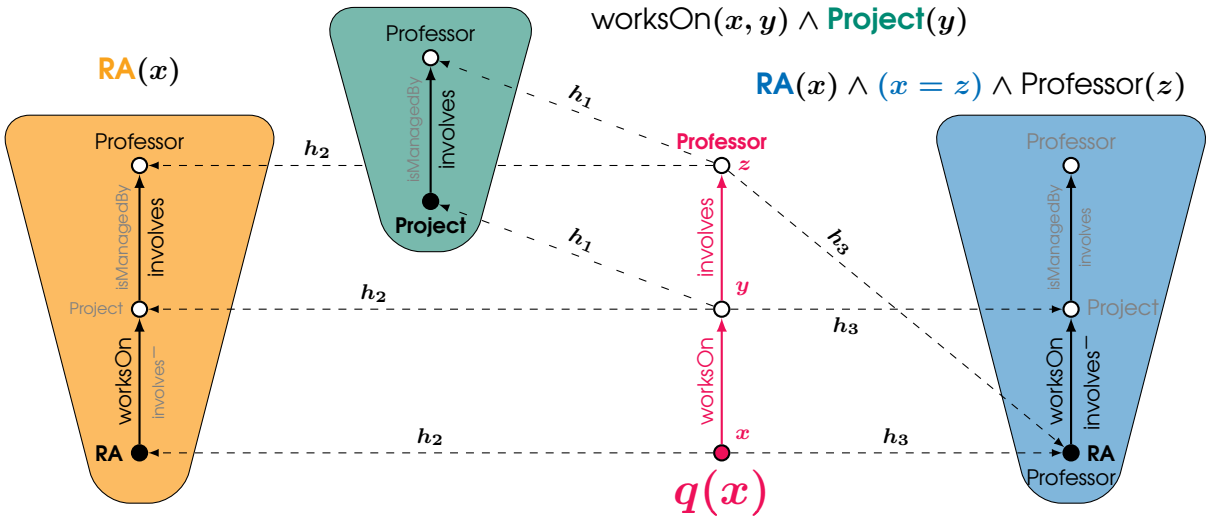
a **tree witness** $\approx$ a query fragment embeddable into a tree of labelled nulls such that only its 'boundary' (join) variables may be answer variables



Tree-Witness Query Rewriting: Example

$q(x) \;=\; \exists y, z\,(\mathsf{worksOn}(x, y) \;\wedge\; \mathsf{involves}(y, z) \;\wedge\; \mathsf{Professor}(z))$

a **tree witness** $\approx$ a query fragment embeddable into a tree of labelled nulls such that only its 'boundary' (join) variables may be answer variables



$q'(x) \;=\; \exists y, z\,\big[\mathsf{RA}(x) \;\vee\; (\mathsf{worksOn}(x, y) \wedge \mathsf{Project}(y)) \;\vee\; (\mathsf{RA}(x) \wedge (x = z) \wedge \mathsf{Professor}(z)) \;\vee$
$(\mathsf{worksOn}(x, y) \wedge [\mathsf{involves}(y, z) \vee \mathsf{worksOn}(z, y) \vee \mathsf{isManagedBy}(y, z)] \wedge \mathsf{Professor}(z))\big]$

Unfolding

idea: replace all the class and property names in the OMQ rewriting
by their SQL definitions in the mapping

$\mathsf{FormationPressure}(\boldsymbol{x}) \land \mathsf{name}(\boldsymbol{x}, \boldsymbol{y}) \land \mathsf{hasDepth}(\boldsymbol{x}, \boldsymbol{z})$

Unfolding

idea: replace all the class and property names in the OMQ rewriting by their SQL definitions in the mapping

$\mathsf{FormationPressure}(x) \land \mathsf{name}(x, y) \land \mathsf{hasDepth}(x, z)$

can be unfolded into

```
SELECT "FP-"||P1.PRESSURE_S" AS x, "P2.IDENTIFIER" AS y,
       "PressureMeasuredDepth-" || P3.PRESSURE_S AS z
FROM PRESSURE P1, PRESSURE P2, PRESSURE P3
WHERE ("FP-" || P1.PRESSURE_S) = ("FP-" || P2.PRESSURE_S)
  AND ("FP-" || P1.PRESSURE_S) = ("FP-" || P3.PRESSURE_S)
```

Unfolding

idea: replace all the class and property names in the OMQ rewriting
by their SQL definitions in the mapping

$\mathsf{FormationPressure}(x) \land \mathsf{name}(x, y) \land \mathsf{hasDepth}(x, z)$

can be unfolded into

```
SELECT "FP-"||P1.PRESSURE_S" AS x, "P2.IDENTIFIER" AS y,
       "PressureMeasuredDepth-" || P3.PRESSURE_S AS z
FROM PRESSURE P1, PRESSURE P2, PRESSURE P3
WHERE ("FP-" || P1.PRESSURE_S) = ("FP-" || P2.PRESSURE_S)
  AND ("FP-" || P1.PRESSURE_S) = ("FP-" || P3.PRESSURE_S)
```

issues in such unfolded SQLs:

- joins over string concatenations (indexes cannot be used by DB engine)
- redundant self-joins (since `PRESSURE.PRESSURE_S` is the **primary key**)

Unfolding

idea: replace all the class and property names in the OMQ rewriting
by their SQL definitions in the mapping

$\mathsf{FormationPressure}(x) \land \mathsf{name}(x, y) \land \mathsf{hasDepth}(x, z)$

can be unfolded into

```
SELECT "FP-"||P1.PRESSURE_S" AS x, "P2.IDENTIFIER" AS y,
       "PressureMeasuredDepth-" || P3.PRESSURE_S AS z
FROM PRESSURE P1, PRESSURE P2, PRESSURE P3
WHERE ("FP-" || P1.PRESSURE_S) = ("FP-" || P2.PRESSURE_S)
  AND ("FP-" || P1.PRESSURE_S) = ("FP-" || P3.PRESSURE_S)
```

issues in such unfolded SQLs: **both are typical for OBDA!**

- joins over string concatenations (indexes cannot be used by DB engine)
- redundant self-joins (since `PRESSURE.PRESSURE_S` is the **primary key**)

use Semantic Query Optimisation

Unfolding

idea: replace all the class and property names in the OMQ rewriting
by their SQL definitions in the mapping

$\mathsf{FormationPressure}(x) \land \mathsf{name}(x, y) \land \mathsf{hasDepth}(x, z)$

can be unfolded into

```
SELECT "FP-"||P1.PRESSURE_S" AS x, "P2.IDENTIFIER" AS y,
       "PressureMeasuredDepth-" || P3.PRESSURE_S AS z
FROM PRESSURE P1, PRESSURE P2, PRESSURE P3
WHERE ("FP-" || P1.PRESSURE_S) = ("FP-" || P2.PRESSURE_S)
  AND ("FP-" || P1.PRESSURE_S) = ("FP-" || P3.PRESSURE_S)
```

issues in such unfolded SQLs: **both are typical for OBDA!**

- joins over string concatenations (indexes cannot be used by DB engine)
- redundant self-joins (since `PRESSURE.PRESSURE_S` is the **primary key**)

use Semantic Query Optimisation

optimised unfolding:

```
SELECT "FP-"||P.PRESSURE_S" AS x, "P.IDENTIFIER" AS y,
       "PressureMeasuredDepth-" || P.PRESSURE_S AS z
FROM PRESSURE P
```

Unfolding with R2RML

IRI templates in R2RML are 'functions,' and so, can encode GLAV mappings

$\varphi(\boldsymbol{x}) \rightsquigarrow \psi(\boldsymbol{x})$, where both $\varphi(\boldsymbol{x})$ and $\psi(\boldsymbol{x})$ are CQs

(with existentially quantified variables)

(for details, see De Giacomo et al. [2018])

NB: the result heavily relies on the lack of UNA and functional properties

Unfolding with R2RML

IRI templates in R2RML are 'functions,' and so, can encode GLAV mappings

$\varphi(\boldsymbol{x}) \rightsquigarrow \psi(\boldsymbol{x})$, where both $\varphi(\boldsymbol{x})$ and $\psi(\boldsymbol{x})$ are CQs

(with existentially quantified variables)

(for details, see De Giacomo et al. [2018])

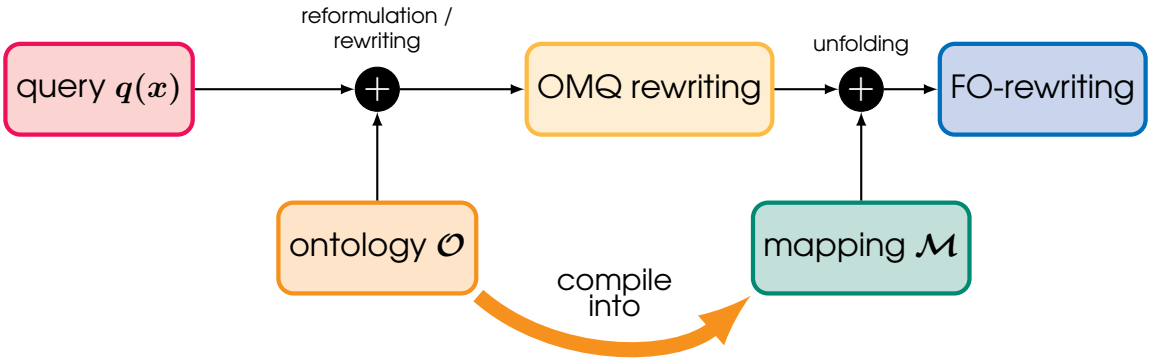
NB: the result heavily relies on the lack of UNA and functional properties

Theorem: every rewriting of an OMQ with an OWL 2 QL ontology can be unfolded with R2RML (equivalently, GLAV) mapping

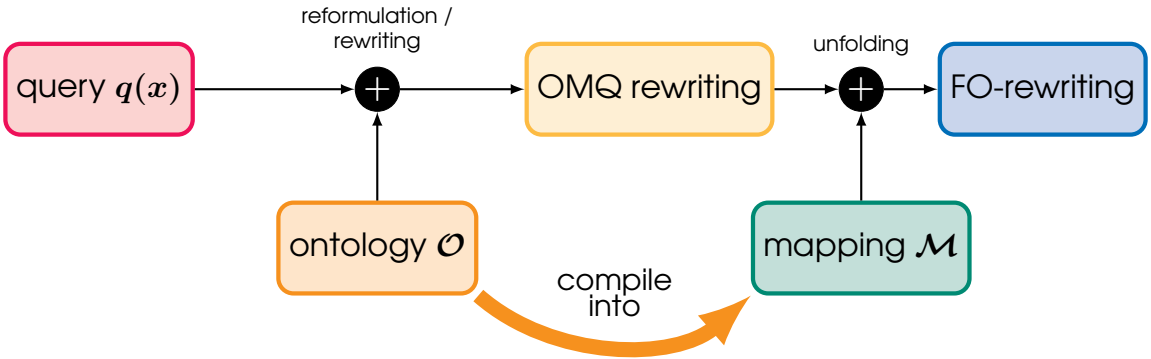
idea: careful unification of query fragments with mappings

(see Calvanese et al. [2012])

Mapping Saturation (aka T-mappings)



Mapping Saturation (aka T-mappings)

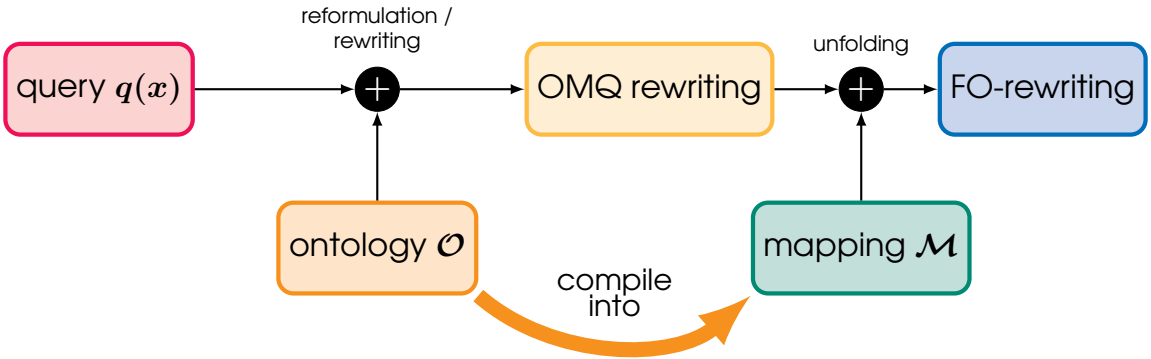


$\exists \text{hasFormationPressure}^- \sqsubseteq \text{FormationPressure}$

```
SELECT WELLBORE.IDENTIFIER, PRESSURE.PRESSURE_S
FROM WELLBORE, PRESSURE
WHERE WELLBORE.REF_EXISTENCE_KIND = 'actual'
      WELLBORE.WELLBORE_S = PRESSURE.FACILITY_S
    ⇝ hasFormationPressure(iri("Wellbore-", IDENTIFIER), iri("FP-", PRESSURE_S))
```

```
SELECT PRESSURE_S FROM PRESSURE
    ⇝ FormationPressure(iri("FP-", PRESSURE_S))
```

Mapping Saturation (aka T-mappings)



$\exists \text{hasFormationPressure}^- \sqsubseteq \text{FormationPressure}$

```
SELECT WELLBORE.IDENTIFIER, PRESSURE.PRESSURE_S
FROM WELLBORE, PRESSURE
WHERE WELLBORE.REF_EXISTENCE_KIND = 'actual'
      WELLBORE.WELLBORE_S = PRESSURE.FACILITY_S
```

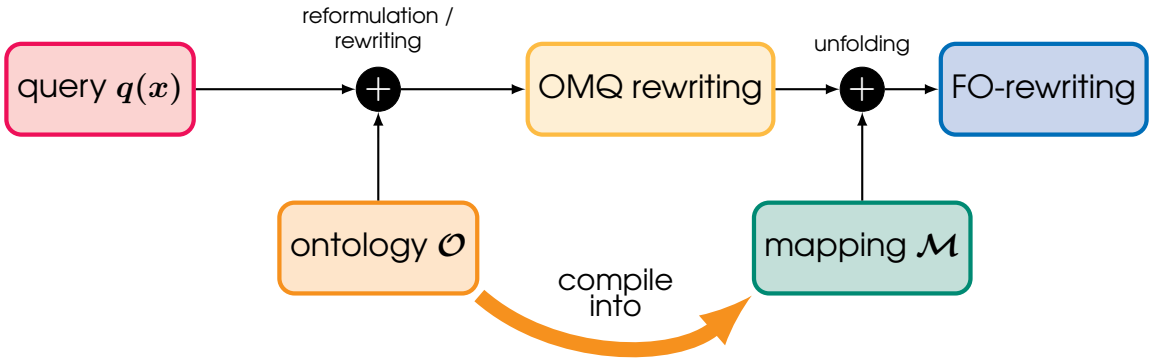
$\rightsquigarrow$ hasFormationPressure(iri("Wellbore-", IDENTIFIER), iri("FP-", PRESSURE_S))

$\rightsquigarrow$ FormationPressure(iri("FP-", PRESSURE_S))

```
SELECT PRESSURE_S FROM PRESSURE
```

$\rightsquigarrow$ FormationPressure(iri("FP-", PRESSURE_S))

Mapping Saturation (aka T-mappings)



~~$\exists \text{hasFormationPressure}^- \sqsubseteq \text{FormationPressure}$~~ **redundant**

```
SELECT WELLBORE.IDENTIFIER, PRESSURE.PRESSURE_S
FROM WELLBORE, PRESSURE
WHERE WELLBORE.REF_EXISTENCE_KIND = 'actual'
      WELLBORE.WELLBORE_S = PRESSURE.FACILITY_S
    ⇝ hasFormationPressure(iri("Wellbore-", IDENTIFIER), iri("FP-", PRESSURE_S))
```

```
    ⇝ FormationPressure(iri("FP-", PRESSURE_S))
```

```
SELECT PRESSURE_S FROM PRESSURE
    ⇝ FormationPressure(iri("FP-", PRESSURE_S))
```

applies to subclass / sub-property axioms domain & range axioms

Optimising Saturated Mappings (1)

```
SELECT PRESSURE.PRESSURE_S
FROM WELLBORE, PRESSURE
WHERE WELLBORE.REF_EXISTENCE_KIND = 'actual'
      WELLBORE.WELLBORE_S = PRESSURE.FACILITY_S
UNION
SELECT PRESSURE_S FROM PRESSURE
```

$\rightsquigarrow$ FormationPressure(`iri("FP-", PRESSURE_S)`)

Optimising Saturated Mappings (1)

```
SELECT PRESSURE.PRESSURE_S
FROM WELLBORE, PRESSURE
WHERE WELLBORE.REF_EXISTENCE_KIND = 'actual'
      WELLBORE.WELLBORE_S = PRESSURE.FACILITY_S
UNION
SELECT PRESSURE_S FROM PRESSURE
```

$\rightsquigarrow$ FormationPressure(iri("FP-", PRESSURE_S))

Query Containment

```
SELECT PRESSURE.PRESSURE_S
FROM WELLBORE, PRESSURE
WHERE WELLBORE.REF_EXISTENCE_KIND = 'actual'
      WELLBORE.WELLBORE_S = PRESSURE.FACILITY_S
UNION
```

```
SELECT PRESSURE_S FROM PRESSURE
```

$\rightsquigarrow$ FormationPressure(iri("FP-", PRESSURE_S))

Optimising Saturated Mappings (1)

```
SELECT PRESSURE.PRESSURE_S
FROM WELLBORE, PRESSURE
WHERE WELLBORE.REF_EXISTENCE_KIND = 'actual'
      WELLBORE.WELLBORE_S = PRESSURE.FACILITY_S
UNION
SELECT PRESSURE_S FROM PRESSURE
```

$\rightsquigarrow$ FormationPressure(iri("FP-", PRESSURE_S))

Query Containment

```
SELECT PRESSURE.PRESSURE_S
FROM WELLBORE, PRESSURE
WHERE WELLBORE.REF_EXISTENCE_KIND = 'actual'
      WELLBORE.WELLBORE_S = PRESSURE.FACILITY_S
UNION
```

```
SELECT PRESSURE_S FROM PRESSURE
```

$\rightsquigarrow$ FormationPressure(iri("FP-", PRESSURE_S))

this optimisation need not be performed on EACH query
it can be done only ONCE, offline, when the system starts

if, however, the mapping is not saturated, then **every** query containing expl:FormationPressure will have to go through the **expansion & reduction** due to query containment

Optimising Saturated Mappings (2)

```
SELECT wellbore_exploration_all.wlbNpdidWellbore, field.fldNpdidField
FROM wellbore_exploration_all INNER JOIN field
    ON wellbore_exploration_all.fldNpdidField = field.fldNpdidField
    ⇝ explorationWellboreForField(iri("wellbore/", wlbNpdidWellbore),
                                  iri("field/", fldNpdidField))
```

Optimising Saturated Mappings (2)

```
SELECT wellbore_exploration_all.wlbNpdidWellbore, field.fldNpdidField
FROM wellbore_exploration_all INNER JOIN field
    ON wellbore_exploration_all.fldNpdidField = field.fldNpdidField
    ⇝ explorationWellboreForField(iri("wellbore/", wlbNpdidWellbore),
                                  iri("field/", fldNpdidField))
```

Query Containment with

```
ALTER TABLE wellbore_exploration_all
...
FOREIGN KEY (fldNpdidField)
        REFERENCES field (fldNpdidField)
```

```
SELECT wlbNpdidWellbore, fldNpdidField
FROM wellbore_exploration_all
    ⇝ explorationWellboreForField(iri("wellbore/", wlbNpdidWellbore),
                                  iri("field/", fldNpdidField))
```

Optimising Saturated Mappings (3)

```
SELECT seaName FROM seis_acquisition
WHERE seaSurveyTypeMain = 'Grunnundersøkelser'
UNION
SELECT seaName FROM seis_acquisition
WHERE seaSurveyTypeMain = 'Ordinær seismisk undersøkelse'
UNION
```

...(5 more from subclasses of :Survey)

$\leadsto$ Survey(iri("survey/", seaName))

Optimising Saturated Mappings (3)

```
SELECT seaName FROM seis_acquisition
WHERE seaSurveyTypeMain = 'Grunnundersøkelser'
UNION
SELECT seaName FROM seis_acquisition
WHERE seaSurveyTypeMain = 'Ordinær seismisk undersøkelse'
UNION
...(5 more from subclasses of :Survey)
    ⇝ Survey(iri("survey/", seaName))
```

OR

```
SELECT seaName FROM seis_acquisition
WHERE seaSurveyTypeMain = 'Grunnundersøkelser'
   OR seaSurveyTypeMain = 'Ordinær seismisk undersøkelse'
   OR
   ...(5 more from subclasses of :Survey)
    ⇝ Survey(iri("survey/", seaName))
```

References

1. R. Kontchakov, M. Rezk, M. Rodríguez-Muro, G. Xiao & M. Zakharyaschev. "Answering SPARQL Queries under the OWL 2 QL Entailment Regime with Databases". In: Proc. of the 13th Int. Semantic Web Conf., ISWC 2014, Part I, vol. 8796 of LNCS, pp. 552–567. Springer, 2014.

2. M. Rodríguez-Muro, R. Kontchakov & M. Zakharyaschev. "Ontology-based data access: Ontop of databases". In: Proc. of the 12th Int. Semantic Web Conf., ISWC 2013, vol. 8218 of LNCS, pp. 558–573. Springer, 2013.

3. J. F. Sequeda, M. Arenas, and D. P. Miranker. "OBDA: Query rewriting or materialization? In practice, both!" In: Proc. of the 13th Int. Semantic Web Conf., ISWC 2014, Part I, vol. 8796 of LNCS, pp. 535–551. Springer, 2014.

4. G. De Giacomo, D. Lembo, M. Lenzerini, A. Poggi & R. Rosati. "Using ontologies for semantic data integration." In: A Comprehensive Guide through the Italian Database Research over the Last 25 Years, vol. 31 of Studies in Big Data. Springer, 2018.

5. D. Calvanese, G. De Giacomo, M. Lenzerini & M. Y. Vardi. "Query processing under GLAV mappings for relational and graph databases". In: PVLDB, 6(2):61–72, 2012.

6. A. Artale, D. Calvanese, R. Kontchakov & M. Zakharyaschev. "The DL-Lite family and relations." In: JAIR, 36:1–69, 2009.