ONTOLOGY-MEDIATED QUERY ANSWERING

Part 3: Extending the Applicability of OMQA Approach

Meghyn Bienvenu (*CNRS & Université de Bordeaux*)

PLAN

Discuss some results along three directions:

Understanding limits / possibilities of query rewriting

OMQA with other query languages

Handling inconsistent KBs

CLOSER LOOK AT QUERY REWRITING

WORST-CASE SIZE OF REWRITINGS

Many of the proposed rewriting algorithms produce unions of conjunctive queries (UCQs = $\vee$ of CQs)

WORST-CASE SIZE OF REWRITINGS

Many of the proposed rewriting algorithms produce unions of conjunctive queries (UCQs = $\vee$ of CQs)

Not hard to see smallest UCQ-rewriting may be exponentially large:

- Query: $A_1^0(x) \wedge \ldots \wedge A_n^0(x)$
- Ontology: $A_1^1 \sqsubseteq A_1^0 \quad A_2^1 \sqsubseteq A_2^0 \quad \ldots \quad A_n^1 \sqsubseteq A_n^0$
- Rewriting: $\bigvee_{(i_1,\ldots,i_n)\in\{0,1\}} A_1^{i_1}(x) \wedge A_1^{i_1}(x) \wedge \ldots \wedge A_1^{i_1}(x)$

WORST-CASE SIZE OF REWRITINGS

Many of the proposed rewriting algorithms produce unions of conjunctive queries (UCQs = $\vee$ of CQs)

Not hard to see smallest UCQ-rewriting may be exponentially large:

- Query: $A_1^0(x) \wedge \ldots \wedge A_n^0(x)$
- Ontology: $A_1^1 \sqsubseteq A_1^0 \quad A_2^1 \sqsubseteq A_2^0 \quad \ldots \quad A_n^1 \sqsubseteq A_n^0$
- Rewriting: $\bigvee_{(i_1,\ldots,i_n)\in\{0,1\}} A_1^{i_1}(x) \wedge A_1^{i_1}(x) \wedge \ldots \wedge A_1^{i_1}(x)$

But: simple polysize FO-rewriting does exist! $\bigwedge_{i=1}^{n}(A_i^0(x) \vee A_i^1(x))$

WORST-CASE SIZE OF REWRITINGS

Many of the proposed rewriting algorithms produce unions of conjunctive queries (UCQs = $\vee$ of CQs)

Not hard to see smallest UCQ-rewriting may be exponentially large:

- Query: $A_1^0(x) \wedge \ldots \wedge A_n^0(x)$
- Ontology: $A_1^1 \sqsubseteq A_1^0 \quad A_2^1 \sqsubseteq A_2^0 \quad \ldots \quad A_n^1 \sqsubseteq A_n^0$
- Rewriting: $\bigvee_{(i_1,\ldots,i_n)\in\{0,1\}} A_1^{i_1}(x) \wedge A_1^{i_1}(x) \wedge \ldots \wedge A_1^{i_1}(x)$

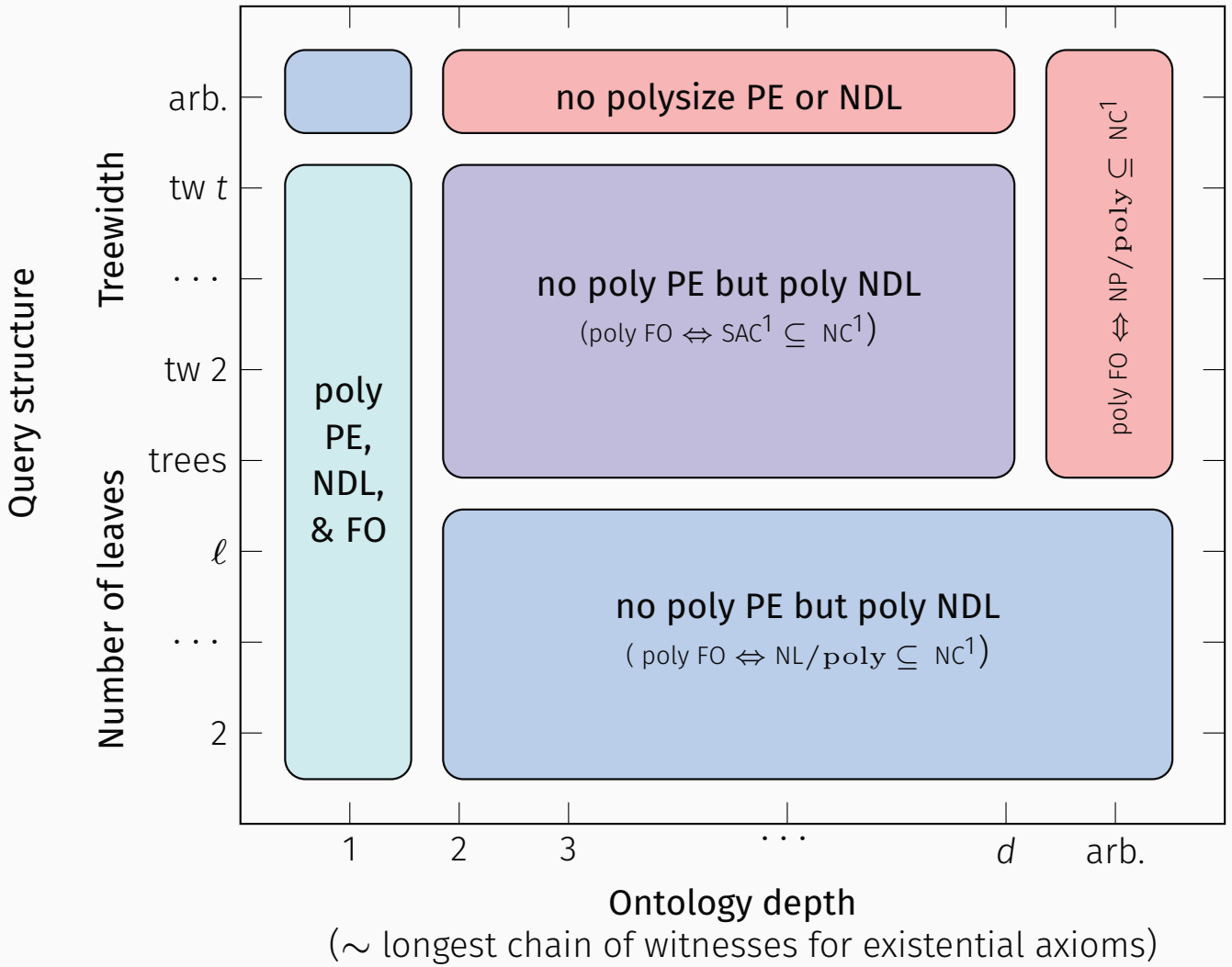
But: simple polysize FO-rewriting does exist! $\bigwedge_{i=1}^{n}(A_i^0(x) \vee A_i^1(x))$

What happens if we adopt other representations?

- positive existential queries (PE), non-recursive Datalog (NDL), first-order queries (FO)

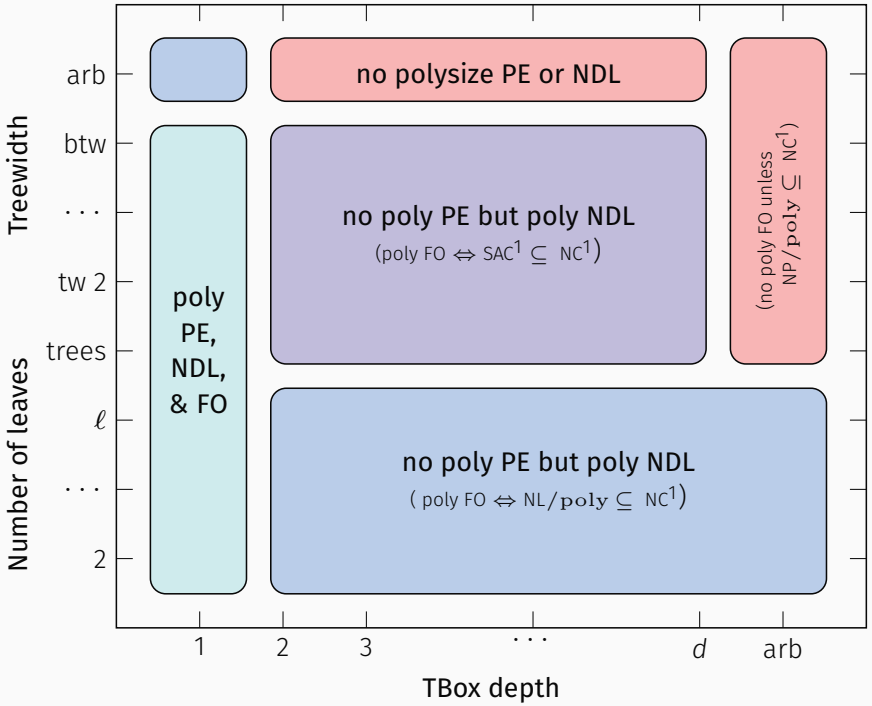
SUCCINCTNESS LANDSCAPE FOR DL-LITE

(for DL-Lite$_\mathcal{R}$ ontologies, so-called 'pure' rewritings)

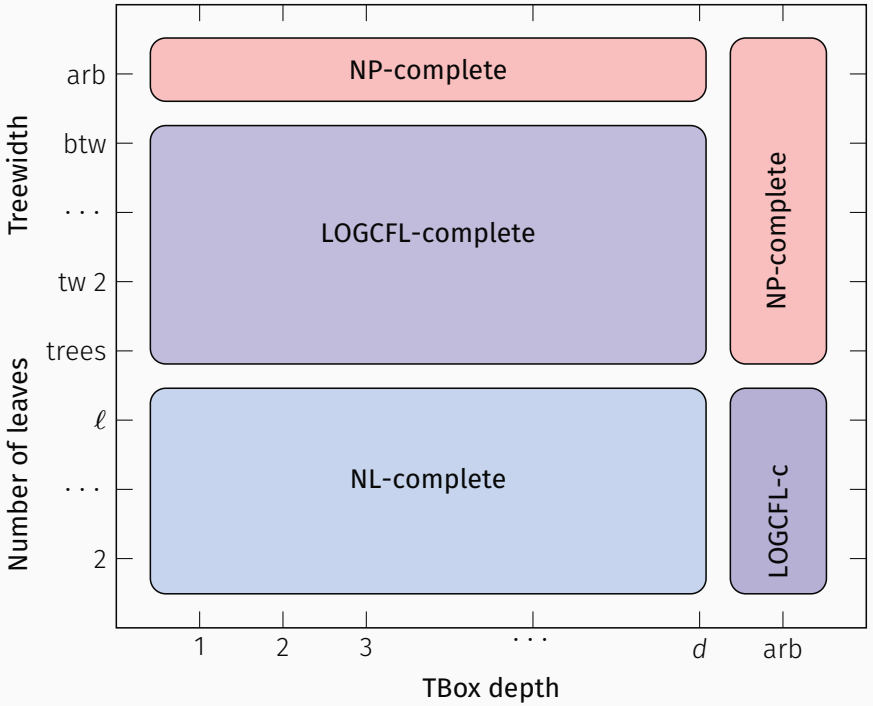


COMPARING SUCCINCTNESS & COMPLEXITY LANDSCAPES

Size of rewritings



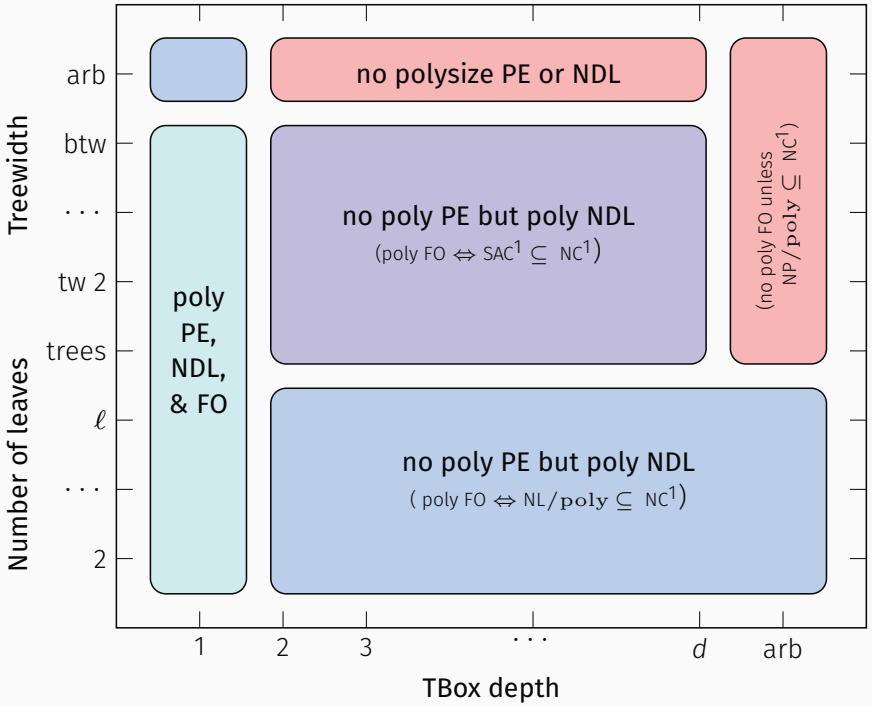
Combined complexity of OMQA



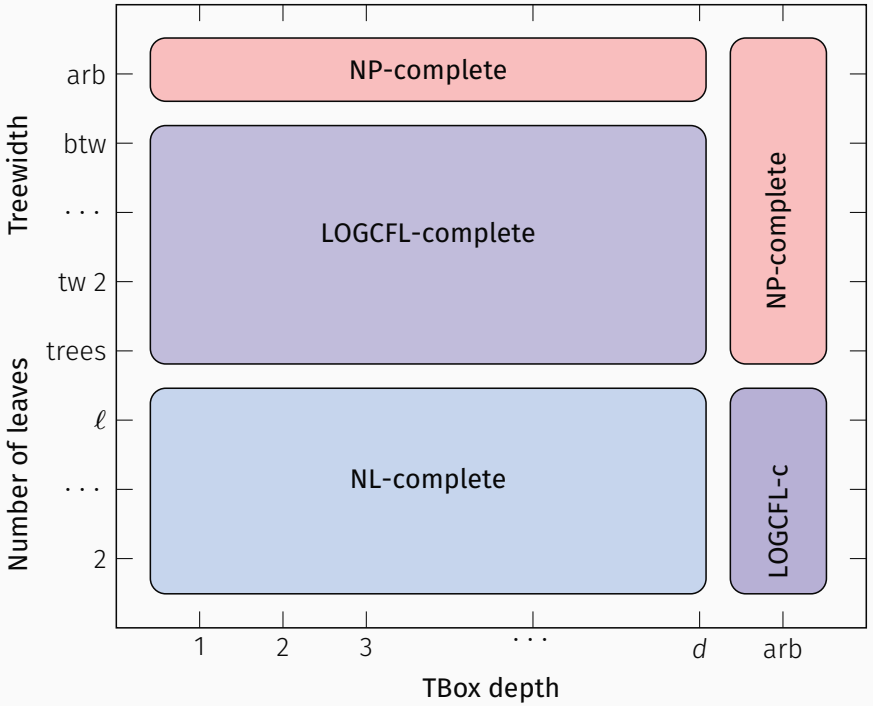
polysize NDL-rewritings ∼ polynomial (LOGCFL / NL) complexity

COMPARING SUCCINCTNESS & COMPLEXITY LANDSCAPES

Size of rewritings



Combined complexity of OMQA



polysize NDL-rewritings ∼ polynomial (LOGCFL / NL) complexity

Can we marry the positive succinctness & complexity results?

OPTIMAL NDL-REWRITINGS

For the three well-behaved classes of OMQs, can define **NDL-rewritings of optimal complexity**:

- rewriting can be constructed by L^C transducer
- evaluating the rewriting can be done in C

with $C \in \{\mathsf{NL}, \mathsf{LOGCFL}\}$ the complexity of the OMQ class

OPTIMAL NDL-REWRITINGS

For the three well-behaved classes of OMQs, can define **NDL-rewritings of optimal complexity**:

- rewriting can be constructed by L^C transducer
- evaluating the rewriting can be done in C

with $C \in \{\mathsf{NL}, \mathsf{LOGCFL}\}$ the complexity of the OMQ class

Upper bound on **time needed to evaluate our NDL-rewritings**:

- depth d / number of leaves ℓ **occur in the exponent**

OPTIMAL NDL-REWRITINGS

For the three well-behaved classes of OMQs, can define
NDL-rewritings of optimal complexity:

- rewriting can be constructed by L^C transducer
- evaluating the rewriting can be done in C

with $C \in \{\mathsf{NL}, \mathsf{LOGCFL}\}$ the complexity of the OMQ class

Upper bound on **time needed to evaluate our NDL-rewritings**:

- depth d / number of leaves ℓ **occur in the exponent**

Is it **possible to do better**?

- formally: **fixed-parameter tractable (FPT)**? $f(d, \ell) \cdot p(|q|, |\mathcal{T}|, |\mathcal{A}|)$

OPTIMAL NDL-REWRITINGS

For the three well-behaved classes of OMQs, can define
NDL-rewritings of optimal complexity:

- rewriting can be constructed by L^C transducer
- evaluating the rewriting can be done in C

with $C \in \{\mathsf{NL}, \mathsf{LOGCFL}\}$ the complexity of the OMQ class

Upper bound on **time needed to evaluate our NDL-rewritings**:

- depth d / number of leaves ℓ **occur in the exponent**

Is it **possible to do better**?

- formally: **fixed-parameter tractable (FPT)**? $f(d, \ell) \cdot p(|q|, |\mathcal{T}|, |\mathcal{A}|)$

Parameterized complexity of answering tree-shaped OMQs $(\mathcal{T}, q)$:

- parameters: depth d of $\mathcal{T}$, number ℓ of leaves in CQs
- **not FPT** if **depth** d taken as parameter W[2]-hard
- **not FPT** if number of **leaves** ℓ taken as parameter W[1]-hard

References: Succinctness & Optimality of Rewritings

Succinctness and combined complexity landscapes:

Bienvenu, Kikot, Kontchakov, Podolskii, and Zakharyaschev: Ontology-Mediated Queries: Combined Complexity and Succinctness of Rewritings via Circuit Complexity. Journal of the ACM (JACM), 2018.

Earlier work on succinctness, also covering 'impure' rewritings:

Gottlob, Kikot, Kontchakov, Podolskii, Schwentick, and Zakharyaschev: The Price of Query Rewriting in Ontology-based Data Access. Artificial Intelligence (AIJ), 2014.

Optimal rewritings and parameterized complexity results:

Bienvenu, Kikot, Kontchakov, Podolskii, Ryzhikov and Zakharyaschev: The Complexity of Ontology-Based Data Access with OWL 2 QL and Bounded Treewidth Queries. PODS 2017.

QUERY REWRITING BEYOND DL-LITE

We have seen that:

- for $\mathcal{EL}$ ontologies, FO-rewritings need not exist
- for $\mathcal{ALC}$ ontologies, FO- and Datalog rewritings may not exist

But these are worst-case results

- only say that some OMQ that does not have a rewriting
- possible that rewritings exist for many OMQs encountered in practice

To extend the applicability of query rewriting beyond DL-Lite:

- devise ways of identifying 'good cases'
- construct rewritings when they exist

DECIDING EXISTENCE OF REWRITINGS

Use $(\mathcal{L}, \mathcal{Q})$ to denote set of **ontology-mediated queries** $(\mathcal{T}, q)$ where:

- $\mathcal{T}$ is an $\mathcal{L}$-TBox
- q is a query from $\mathcal{Q}$ $\qquad \mathcal{Q} \in \{\mathsf{IQ}, \mathsf{CQ}\}$

For example: $(\mathcal{EL}, \mathsf{CQ})$, $(\mathcal{ALC}, \mathsf{IQ})$

FO-rewritability in $(\mathcal{L}, \mathcal{Q})$

- Input: OMQ $(\mathcal{T}, q)$ from $(\mathcal{L}, \mathcal{Q})$
- Problem: **decide whether $(\mathcal{T}, q)$ has an FO-rewriting**

Datalog-rewritability decision problem can be **defined analogously**

FO-REWRITABILITY IN EL

$\mathcal{EL} : \sqcap, \exists r.C$

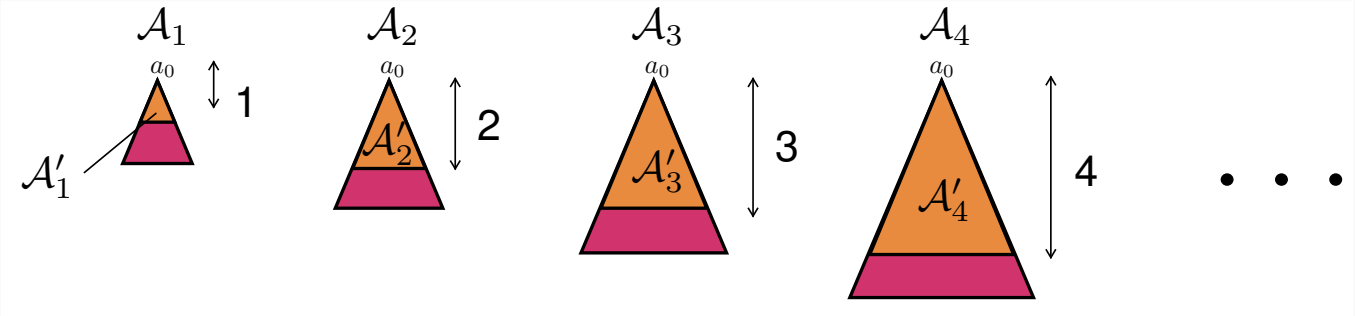
FO-rewritability is **EXPTIME-complete** in $(\mathcal{EL}, \mathsf{IQ})$ and $(\mathcal{EL}, \mathsf{CQ})$

$\mathcal{EL} : \sqcap, \exists r.C$

FO-rewritability is **EXPTIME-complete** in $(\mathcal{EL}, \mathrm{IQ})$ and $(\mathcal{EL}, \mathrm{CQ})$

Characterization of non-existence of FO-rewriting

OMQ $(\mathcal{T}, A(x))$ is **not FO-rewritable** iff there exist tree-shaped ABoxes



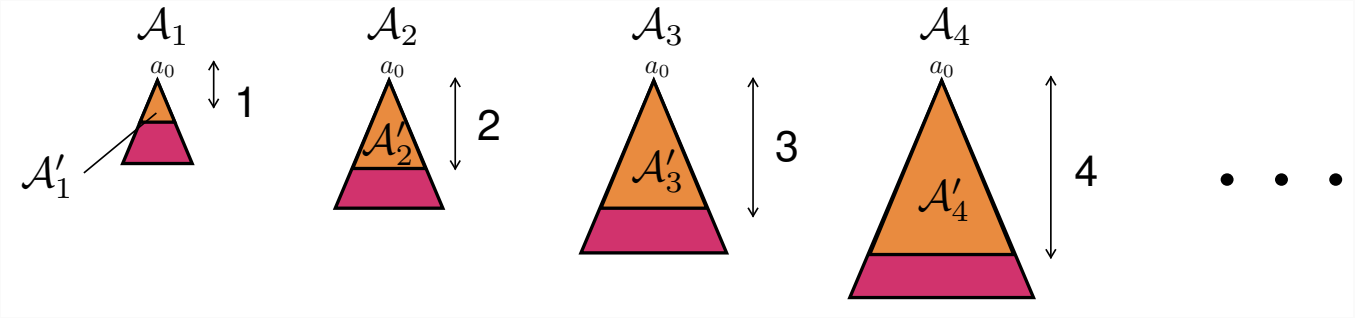
such that for all $i \geq 1$: $\mathcal{T}, \mathcal{A}_i \models A(a_0)$ and $\mathcal{T}, \mathcal{A}'_i \not\models A(a_0)$

$\mathcal{EL} : \sqcap, \exists r.C$

FO-rewritability is **EXPTIME-complete** in $(\mathcal{EL}, \text{IQ})$ and $(\mathcal{EL}, \text{CQ})$

Characterization of non-existence of FO-rewriting

OMQ $(\mathcal{T}, A(x))$ is **not FO-rewritable** iff there exist tree-shaped ABoxes



such that for all $i \geq 1$: $\mathcal{T}, \mathcal{A}_i \models A(a_0)$ and $\mathcal{T}, \mathcal{A}'_i \not\models A(a_0)$

Pumping argument: enough to find ABox of **particular finite size** k_0

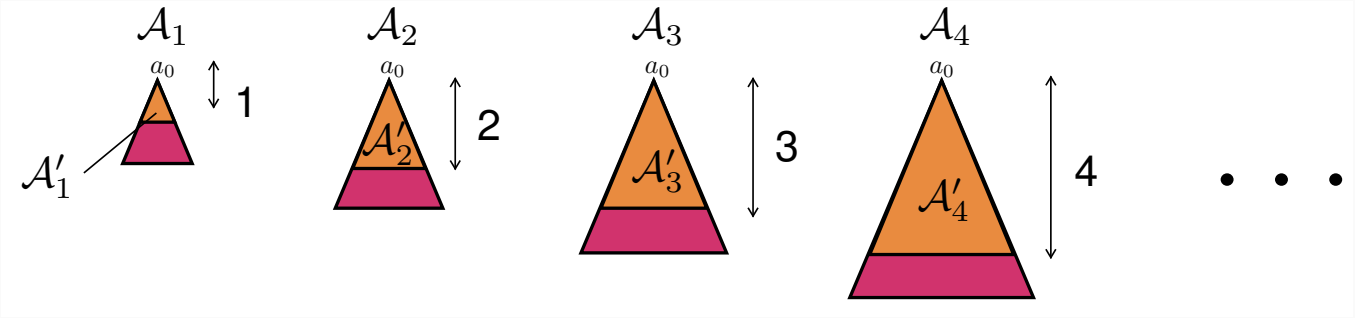
- desired ABox $\mathcal{A}_{k_0}$ exists $\Rightarrow$ can construct full sequence of ABoxes

$\mathcal{EL} : \sqcap, \exists r.C$

FO-rewritability is **EXPTIME-complete** in $(\mathcal{EL}, \text{IQ})$ and $(\mathcal{EL}, \text{CQ})$

Characterization of non-existence of FO-rewriting

OMQ $(\mathcal{T}, A(x))$ is **not FO-rewritable** iff there exist tree-shaped ABoxes



such that for all $i \geq 1$: $\mathcal{T}, \mathcal{A}_i \models A(a_0)$ and $\mathcal{T}, \mathcal{A}'_i \not\models A(a_0)$

Pumping argument: enough to find ABox of **particular finite size** k_0

- desired ABox $\mathcal{A}_{k_0}$ exists $\Rightarrow$ can construct full sequence of ABoxes

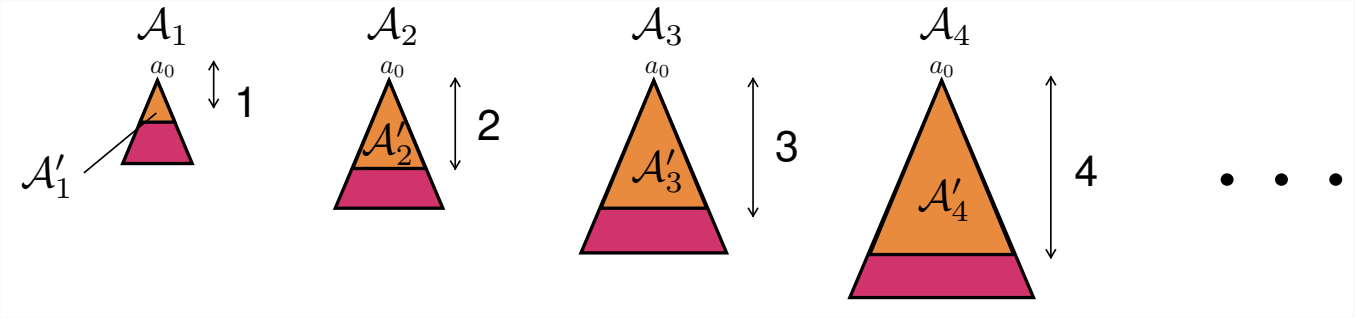
Use **tree automata** to check whether **such a witness ABox exists**

$\mathcal{EL} : \top, \exists r.C$

FO-rewritability is **EXPTIME-complete** in $(\mathcal{EL}, \text{IQ})$ and $(\mathcal{EL}, \text{CQ})$

Characterization of non-existence of FO-rewriting

OMQ $(\mathcal{T}, A(x))$ is **not FO-rewritable** iff there exist tree-shaped ABoxes



such that for all $i \geq 1$: $\mathcal{T}, \mathcal{A}_i \models A(a_0)$ and $\mathcal{T}, \mathcal{A}'_i \not\models A(a_0)$

Pumping argument: enough to find ABox of **particular finite size** k_0

- desired ABox $\mathcal{A}_{k_0}$ exists $\Rightarrow$ can construct full sequence of ABoxes

Use **tree automata** to check whether **such a witness ABox exists**

Can **generalize this technique to handle CQs** as well

COMPUTING FO-REWRITINGS IN EL

Idea for IQs: use **existing backwards-chaining rewriting procedure**

- if FO-rewriting does exist, terminates
- to ensure termination in general: use characterization result

To make practical: **decomposed algorithm**

- allows for structure sharing
- produces (succinct) NDL-rewriting instead of UCQ-rewriting

Experimental results are **very encouraging**:

- terminates quickly, produced rewritings are typically small
- suggests that in practice **FO-rewritings do exist** for **majority of IQs**

Recently **extended to handle CQs** with promising results

REWRITABILITY FOR (ALC,IQ)

FO-rewritability and Datalog-rewritability of $(\mathcal{ALC}, \mathrm{IQ})$ are both NEXPTIME-complete.

REWRITABILITY FOR (ALC,IQ)

FO-rewritability and Datalog-rewritability of $(\mathcal{ALC}, \mathrm{IQ})$ are both NEXPTIME-complete.

Upper bound: connection to constraint satisfaction problems (CSPs)

- CSP($\mathfrak{B}$): decide if homomorphism from input structure $\mathcal{D}$ into $\mathfrak{B}$
- (Boolean) OMQs in $(\mathcal{ALC}, \mathrm{IQ}) \sim$ (complement of) CSPs
- exponential reduction to problem of deciding whether a CSP is definable in FO / Datalog
- use NP upper bounds for latter problems

FO-rewritability of $(\mathcal{ALC}, \mathrm{UCQ})$: 2NEXPTIME-complete

References: FO-rewritability of Horn DLs

FO-rewritability of instance queries

Bienvenu, Lutz, Wolter: First Order-Rewritability of Atomic Queries in Horn Description Logics. IJCAI 2013.

FO-rewritability of conjunctive queries

Bienvenu, Hansen, Lutz, Wolter: First Order-Rewritability and Containment of Conjunctive Queries in Horn Description Logics. IJCAI 2016.

Practical algorithms and implementations

Hansen, Lutz, Seylan, Wolter: Efficient Query Rewriting in the Description Logic EL and Beyond. IJCAI 2015.

Hansen & Lutz: Computing FO-Rewritings in $\mathcal{EL}$ in Practice: from Atomic to Conjunctive Queries. ISWC 2017.

References: Rewritability in expressive DLs

CSP connection & FO- and Datalog-rewritability of IQs

Bienvenu, ten Cate, Lutz, and Wolter: Ontology-based Data Access: A Study through Disjunctive Datalog, CSP, and MMSNP. TODS, 2014.

FO-rewritability of UCQs

Feier, Kuusisto, and Lutz: Rewritability in Monadic Disjunctive Datalog, MMSNP, and Expressive Description Logics. ICDT 2017.

Practical, incomplete approach for Datalog rewritings

Kaminski, Nenov, and Cuenca Grau: Datalog Rewritability of Disjunctive Datalog Programs and its Applications to Ontology Reasoning. AAAI 2014.

BEYOND CONJUNCTIVE QUERIES

QUERIES WITH NEGATION

Conjunctive query with safe negation ($CQ^{\neg s}$):

- like a CQ, but can also have negated atoms
- safety condition: every variable occurs in some positive atom
- example: find menus whose main course is not spicy

$$\exists y\, \mathsf{Menu}(x) \wedge \mathsf{hasMain}(x, y) \wedge \neg \mathsf{Spicy}(y)$$

QUERIES WITH NEGATION

Conjunctive query with safe negation ($\mathrm{CQ}^{\neg s}$):

- like a CQ, but **can also have negated atoms**
- **safety condition**: **every variable occurs in some positive atom**
- example: find menus whose main course is not spicy

$$\exists y\, \mathrm{Menu}(x) \wedge \mathrm{hasMain}(x, y) \wedge \neg \mathrm{Spicy}(y)$$

Conjunctive query with inequalities ($\mathrm{CQ}^{\neq}$)

- like a CQ, but **can also have atoms** $t_1 \neq t_2$ (t_1, t_2 vars or individuals)
- example: find restaurant offering two menus having different dessert courses

$$\exists y_1 y_2 z_1 z_2 \quad \mathrm{offers}(x, y_1) \wedge \mathrm{Menu}(y_1) \wedge \mathrm{hasDessert}(y_1, z_1) \wedge \mathrm{offers}(x, y_2) \wedge \mathrm{Menu}(y_2) \wedge \mathrm{hasDessert}(y_2, z_2) \wedge z_1 \neq z_2$$

Conjunctive query with safe negation ($\mathrm{CQ}^{\neg s}$):

- like a CQ, but can also have negated atoms
- safety condition: every variable occurs in some positive atom
- example: find menus whose main course is not spicy

$$\exists y\, \mathrm{Menu}(x) \wedge \mathrm{hasMain}(x, y) \wedge \neg \mathrm{Spicy}(y)$$

Conjunctive query with inequalities ($\mathrm{CQ}^{\neq}$)

- like a CQ, but can also have atoms $t_1 \neq t_2$ (t_1, t_2 vars or individuals)
- example: find restaurant offering two menus having different dessert courses

$$\exists y_1 y_2 z_1 z_2 \quad \mathrm{offers}(x, y_1) \wedge \mathrm{Menu}(y_1) \wedge \mathrm{hasDessert}(y_1, z_1) \wedge \mathrm{offers}(x, y_2) \wedge \mathrm{Menu}(y_2) \wedge \mathrm{hasDessert}(y_2, z_2) \wedge z_1 \neq z_2$$

Note: can define $\mathrm{UCQ}^{\neg s}$s and $\mathrm{UCQ}^{\neq}$s in the obvious way

UNDECIDABILITY RESULTS FOR QUERIES WITH NEGATION

Adding negation leads to undecidability even in very restricted settings.

Theorem The following problems are undecidable:

- $\text{CQ}^{\neg s}$ answering in DL-Lite_R
- $\text{UCQ}^{\neg s}$ answering in $\mathcal{EL}_\bot$
- $\text{CQ}^{\neq}$ answering in DL-Lite_R
- $\text{CQ}^{\neq}$ answering in $\mathcal{EL}_\bot$

UNDECIDABILITY RESULTS FOR QUERIES WITH NEGATION

Adding negation leads to undecidability even in very restricted settings.

Theorem The following problems are undecidable:

- $\text{CQ}^{\neg^S}$ answering in DL-Lite$_R$
- $\text{UCQ}^{\neg^S}$ answering in $\mathcal{EL}_\bot$
- $\text{CQ}^{\neq}$ answering in DL-Lite$_R$
- $\text{CQ}^{\neq}$ answering in $\mathcal{EL}_\bot$

Possible solution: adopt alternative (epistemic) semantics

UNDECIDABILITY RESULTS FOR QUERIES WITH RECURSION

Significant interest in combining DLs with Datalog rules

Unfortunately, this almost always leads to undecidability:

Theorem Datalog query answering is undecidable in every DL that can express (directly or indirectly) $A \sqsubseteq \exists r.A$

In particular: undecidable in both DL-Lite and $\mathcal{EL}$

UNDECIDABILITY RESULTS FOR QUERIES WITH RECURSION

Significant interest in combining DLs with Datalog rules

Unfortunately, this almost always leads to undecidability:

Theorem Datalog query answering is undecidable in every DL that can express (directly or indirectly) $A \sqsubseteq \exists r.A$

In particular: undecidable in both DL-Lite and $\mathcal{EL}$

Possible solutions:

- use restricted classes of Datalog queries (e.g. path queries)
- DL-safe rules: can only apply rules to (named) individuals

Prominent navigational query languages (graph databases):

Prominent navigational query languages (graph databases):

Regular Path Queries (RPQs): find pairs of objects that are connected by a chain of roles that comply with a given regular language

$$(\mathsf{hasCourse} \cup \mathsf{courseOf}^{-}) \cdot (\mathsf{hasIngred} \cup \mathsf{ingredOf}^{-})^{*} \cdot \mathsf{Spicy?}(x, y)$$

Prominent navigational query languages (graph databases):

Regular Path Queries (RPQs): find pairs of objects that are connected by a chain of roles that comply with a given regular language

$$(\text{hasCourse} \cup \text{courseOf}^{-}) \cdot (\text{hasIngred} \cup \text{ingredOf}^{-})^{*} \cdot \text{Spicy?}(x, y)$$

Conjunctive RPQs: allow to join RPQs conjunctively

- similar to CQs, but each atom is an RPQ
- extend CQs with the navigational power of RPQs

Both languages have 1-way and 2-way variants

COMPLEXITY LANDSCAPE FOR PATH QUERIES

	2RPQ		C2RPQ	
	data	combined	data	combined
Graph DBs & RDFS	NL-c	NL-c	NL-c	NP-c
DL-Lite	NL-c	PTIME-c	NL-c	PSPACE-c
$\mathcal{EL}$... Horn-$\mathcal{SHIQ}$	PTIME-c	PTIME-c	PTIME-c	PSPACE-c
$\mathcal{ALC}$... $\mathcal{SHIQ}$	coNP-h	EXP-c	coNP-h	2EXP-c

Remarks:

- positive results for Horn DLs: low complexity for 2RPQs, remains PTIME data for C2RPQs
- tractable data complexity: shown by adapting CQ answering approach presented in this course

COUNTING QUERIES

Aggregate queries (COUNT, SUM, AVG) very common in databases, but not well explored for OMQA

Counting conjunctive queries: how many ways to map the CQ?

Cardinality queries: how many A? how many r?

Aggregate queries (COUNT, SUM, AVG) very common in databases, but not well explored for OMQA

Counting conjunctive queries: how many ways to map the CQ?

Cardinality queries: how many A? how many r?

Challenging to handle:

- coNP-c data complexity even for cardinality queries in DL-Lite
- 2EXPTIME-c in combined complexity (counting CQs + DL-Lite$_R$/ $\mathcal{EL}$)
- cannot rely upon canonical model to get optimal value

Some tractable / lower complexity cases have been identified

Queries with Inequalities or Negation

Gutiérrez-Basulto, Ibáñez-García, Kontchakov, Kostylev: Queries with Negation and Inequalities over Lightweight Ontologies. Journal of Web Semantics, 2015.

Regular Path Queries

Bienvenu, Ortiz, Simkus: Regular Path Queries in Lightweight Description Logics: Complexity and Algorithms. JAIR, 2015.

Counting Queries

Bienvenu, Manière, Thomazo: Answering Counting Queries over DL-Lite Ontologies. IJCAI 2020.

HANDLING INCONSISTENT KBS

HANDLING DATA INCONSISTENCIES

In realistic settings, can expect some errors in the data

- ABox likely to be inconsistent with the TBox (ontology)

Standard semantics: everything is implied - not informative!

- when $\mathcal{K}$ unsatisfiable, cert$(q, \mathcal{K})$ contains all possible tuples

HANDLING DATA INCONSISTENCIES

In realistic settings, can expect some errors in the data

- ABox likely to be inconsistent with the TBox (ontology)

Standard semantics: everything is implied - not informative!

- when $\mathcal{K}$ unsatisfiable, cert$(q, \mathcal{K})$ contains all possible tuples

Two approaches to inconsistency handling:

- resolve the inconsistencies
  - preferable, but not always applicable!
- live with the inconsistencies - adopt alternative semantics
  - meaningful answers to queries despite inconsistencies

Note: focus on case where errors in ABox (assume TBox reliable)

EXAMPLE: REASONABLE INFERENCES

TBox $\mathcal{T}_{\text{univ}}$:

$\text{Prof} \sqsubseteq \text{Fac}$	$\text{Prof} \sqsubseteq \exists\text{teaches}$	$\text{Prof} \sqsubseteq \neg\text{Lect}$	$\text{Fac} \sqsubseteq \neg\text{Course}$
$\text{Lect} \sqsubseteq \text{Fac}$	$\text{Lect} \sqsubseteq \exists\text{teaches}$	$\text{Prof} \sqsubseteq \neg\text{Fellow}$	
$\text{Fellow} \sqsubseteq \text{Fac}$	$\exists\text{teaches}^- \sqsubseteq \text{Course}$	$\text{Lect} \sqsubseteq \neg\text{Fellow}$	

Consider following ABoxes:

$$\mathcal{A}_1 = \{\text{Prof}(\text{anna}), \text{Lect}(\text{anna}), \text{Fellow}(\text{alex})\}$$
$$\mathcal{A}_2 = \{\text{Prof}(\text{anna}), \text{Fellow}(\text{alex}), \text{Lect}(\text{alex})\}$$

Which assertions would be reasonable to infer from these two KBs?

$\text{Prof}(\text{anna})$	$\text{Lect}(\text{anna})$	$\text{Fac}(\text{anna})$
$\text{Fellow}(\text{alex})$	$\text{Lect}(\text{alex})$	$\text{Fac}(\text{alex})$

REPAIRS

Many proposed semantics are based upon the notion of repair

Repair of knowledge base $(\mathcal{T}, \mathcal{A})$

= inclusion-maximal subset of $\mathcal{A}$ s.t. $(\mathcal{T}, \mathcal{A})$ satisfiable

Intuition: different ways of achieving consistency while retaining as much of the original data as possible

Many proposed semantics are based upon the notion of repair

Repair of knowledge base $(\mathcal{T}, \mathcal{A})$

= inclusion-maximal subset of $\mathcal{A}$ s.t. $(\mathcal{T}, \mathcal{A})$ satisfiable

Intuition: different **ways of achieving consistency** while **retaining as much of the original data as possible**

For example, the following knowledge base $\mathcal{K}^* = \langle \mathcal{T}^*, \mathcal{A}^* \rangle$ with

$$\mathcal{T}^* = \{\mathsf{Prof} \sqsubseteq \mathsf{PhD}, \mathsf{Postdoc} \sqsubseteq \mathsf{PhD}, \mathsf{Prof} \sqsubseteq \neg\mathsf{Postdoc}\}$$

$$\mathcal{A}^* = \{\mathsf{Postdoc}(\mathit{ann}), \mathsf{Prof}(\mathit{ann}), \mathsf{Teach}(\mathit{ann}, c)\}$$

has two repairs:

$$\mathcal{R}_1 = \{\mathsf{Postdoc}(\mathit{ann}), \mathsf{Teach}(\mathit{ann}, c)\}$$

$$\mathcal{R}_2 = \{\mathsf{Prof}(\mathit{ann}), \mathsf{Teach}(\mathit{ann}, c)\}$$

AR SEMANTICS: PLAUSIBLE ANSWERS

AR semantics ('AR' for ABox Repair):
query each repair separately, intersect results

$$\mathcal{T}, \mathcal{A} \models_{\mathsf{AR}} q(\vec{a}) \quad \Leftrightarrow \quad \mathcal{T}, \mathcal{B} \models q(\vec{a}) \text{ for every repair } \mathcal{B} \in \mathit{Rep}(\mathcal{K})$$

Plausible answers: hold no matter which repair is chosen

Generally viewed as natural semantics:

- cautious inference (KR), consistent query answering (DBs)

AR SEMANTICS: PLAUSIBLE ANSWERS

AR semantics ('AR' for ABox Repair):
query each repair separately, intersect results

$$\mathcal{T}, \mathcal{A} \models_{\mathsf{AR}} q(\vec{a}) \quad \Leftrightarrow \quad \mathcal{T}, \mathcal{B} \models q(\vec{a}) \text{ for every repair } \mathcal{B} \in \mathit{Rep}(\mathcal{K})$$

Plausible answers: hold no matter which repair is chosen

Generally viewed as natural semantics:

- cautious inference (KR), consistent query answering (DBs)

On our example KB:

- $\mathcal{K}^* \not\models_{\mathsf{AR}} \mathsf{Postdoc}(\mathit{ann})$ $\quad \langle \mathcal{T}^*, \mathcal{R}_2 \rangle \not\models \mathsf{Postdoc}(\mathit{ann})$
- $\mathcal{K}^* \models_{\mathsf{AR}} \mathsf{PhD}(\mathit{ann})$ $\quad \mathsf{Postdoc}(\mathit{ann}) \in \mathcal{R}_1$ & $\mathsf{Prof}(\mathit{ann}) \in \mathcal{R}_2$
- $\mathcal{K}^* \models_{\mathsf{AR}} \mathsf{Teach}(\mathit{ann}, c)$ $\quad \mathsf{Teach}(\mathit{ann}, c) \in \mathcal{R}_1 \cap \mathcal{R}_2$

AR SEMANTICS: PLAUSIBLE ANSWERS

AR semantics ('AR' for ABox Repair):
query each repair separately, intersect results

$$\mathcal{T}, \mathcal{A} \models_{\mathsf{AR}} q(\vec{a}) \quad \Leftrightarrow \quad \mathcal{T}, \mathcal{B} \models q(\vec{a}) \text{ for every repair } \mathcal{B} \in \mathit{Rep}(\mathcal{K})$$

Plausible answers: hold no matter which repair is chosen

Generally viewed as natural semantics:

- cautious inference (KR), consistent query answering (DBs)

On our example KB:

- $\mathcal{K}^* \not\models_{\mathsf{AR}} \mathsf{Postdoc}(\mathit{ann})$ $\quad \langle \mathcal{T}^*, \mathcal{R}_2 \rangle \not\models \mathsf{Postdoc}(\mathit{ann})$
- $\mathcal{K}^* \models_{\mathsf{AR}} \mathsf{PhD}(\mathit{ann})$ $\quad \mathsf{Postdoc}(\mathit{ann}) \in \mathcal{R}_1$ & $\mathsf{Prof}(\mathit{ann}) \in \mathcal{R}_2$
- $\mathcal{K}^* \models_{\mathsf{AR}} \mathsf{Teach}(\mathit{ann}, c)$ $\quad \mathsf{Teach}(\mathit{ann}, c) \in \mathcal{R}_1 \cap \mathcal{R}_2$

Bad news: querying under AR semantics is **intractable**

- **coNP-hard** in data complexity, even in simple settings (DL-Lite)

BRAVE AND IAR SEMANTICS: POSSIBLE AND (ALMOST) SURE ANSWERS

Brave semantics: answers that hold in at least one repair

$$\mathcal{T}, \mathcal{A} \models_{\text{brave}} q(\vec{a}) \quad \Leftrightarrow \quad \mathcal{T}, \mathcal{R} \models q(\vec{a}) \text{ for some repair } \mathcal{R} \in \mathit{Rep}(\mathcal{K})$$

Answers supported by consistent part of data $\rightsquigarrow$ possible answers

More permissive than AR semantics: $\mathcal{K}^* \models_{\text{brave}} \text{Postdoc}(\mathit{ann})$

BRAVE AND IAR SEMANTICS: POSSIBLE AND (ALMOST) SURE ANSWERS

Brave semantics: answers that hold in at least one repair

$$\mathcal{T}, \mathcal{A} \models_{\text{brave}} q(\vec{a}) \quad \Leftrightarrow \quad \mathcal{T}, \mathcal{R} \models q(\vec{a}) \text{ for some repair } \mathcal{R} \in \mathit{Rep}(\mathcal{K})$$

Answers supported by consistent part of data $\rightsquigarrow$ possible answers

More permissive than AR semantics: $\mathcal{K}^* \models_{\text{brave}} \text{Postdoc}(\mathit{ann})$

IAR semantics: query the intersection of the repairs

$$\mathcal{T}, \mathcal{A} \models_{\text{IAR}} q(\vec{a}) \quad \Leftrightarrow \quad \mathcal{T}, \mathcal{D} \models q(\vec{a}) \text{ where } \mathcal{D} = \bigcap_{\mathcal{R} \in \mathit{Rep}(\mathcal{K})} \mathcal{R}$$

Idea: focus on surest facts, not involved in any contradiction

Stricter than AR semantics: $\mathcal{K}^* \not\models_{\text{IAR}} \text{PhD}(\mathit{ann})$

BRAVE AND IAR SEMANTICS: POSSIBLE AND (ALMOST) SURE ANSWERS

Brave semantics: answers that hold in **at least one repair**

$$\mathcal{T}, \mathcal{A} \models_{\text{brave}} q(\vec{a}) \quad \Leftrightarrow \quad \mathcal{T}, \mathcal{R} \models q(\vec{a}) \text{ for some repair } \mathcal{R} \in \mathit{Rep}(\mathcal{K})$$

Answers **supported by consistent part of data** $\rightsquigarrow$ **possible answers**

More permissive than AR semantics: $\mathcal{K}^* \models_{\text{brave}} \text{Postdoc}(\mathit{ann})$

IAR semantics: query the **intersection of the repairs**

$$\mathcal{T}, \mathcal{A} \models_{\text{IAR}} q(\vec{a}) \quad \Leftrightarrow \quad \mathcal{T}, \mathcal{D} \models q(\vec{a}) \text{ where } \mathcal{D} = \bigcap_{\mathcal{R} \in \mathit{Rep}(\mathcal{K})} \mathcal{R}$$

Idea: focus on surest facts, not involved in any contradiction

Stricter than AR semantics: $\mathcal{K}^* \not\models_{\text{IAR}} \text{PhD}(\mathit{ann})$

Good news: brave and IAR semantics **tractable for DL-Lite**

- still possible to use **query rewriting**

REFERENCES: INCONSISTENCY-TOLERANT OMQA

Introductory chapter

Bienvenu, Bourgaux: Inconsistency-Tolerant Querying of Description Logic Knowledge Bases. Lecture Notes of the 12th International Reasoning Web Summer School, 2016.

Recent survey article

Bienvenu: A Short Survey on Inconsistency Handling in Ontology-Mediated Query Answering. Künstliche Intelligenz, 2020.

SAT-based approach to AR semantics

Bienvenu, Bourgaux, Goasdoué: Computing and Explaining Query Answers over Inconsistent DL-Lite Knowledge Bases. JAIR, 2019.

OMQA RESEARCH

RESEARCH TOPICS IN OMQA

Ontology-mediated query answering:

- new paradigm for intelligent information systems
- offers many advantages, but also computational challenges
- active area with lots left to explore!

RESEARCH TOPICS IN OMQA

Ontology-mediated query answering:

- new paradigm for intelligent information systems
- offers many advantages, but also computational challenges
- active area with lots left to explore!

Efficient OMQA algorithms:

- optimized rewriting algorithms: compact rewritings, exploit mapping structure, cost-based rewriting selection
- tackling more expressive DLs: identify easier cases (existence of rewritings), use upper + lower approximations

RESEARCH TOPICS IN OMQA

Ontology-mediated query answering:

- new paradigm for intelligent information systems
- offers many advantages, but also computational challenges
- active area with lots left to explore!

Efficient OMQA algorithms:

- optimized rewriting algorithms: compact rewritings, exploit mapping structure, cost-based rewriting selection
- tackling more expressive DLs: identify easier cases (existence of rewritings), use upper + lower approximations

Support for building and maintaining OMQA systems

- ontology + mapping bootstrapping, module extraction, debugging, ontology evolution and versioning
- learning ontologies, queries, mappings
- inspired new reasoning tasks: query inseparability, query emptiness, justification finding, logical difference, ...

RESEARCH TOPICS IN OMQA

Improving the usability of OMQA systems

- interfaces and support for query formulation
- explaining query (non-)answers

RESEARCH TOPICS IN OMQA

Improving the usability of OMQA systems

- interfaces and support for query formulation
- explaining query (non-)answers

Broadening the scope:

- new data formats: graph data, key-value stores, temporal data
- further query languages: regular path queries, streaming queries

RESEARCH TOPICS IN OMQA

Improving the usability of OMQA systems

- interfaces and support for query formulation
- explaining query (non-)answers

Broadening the scope:

- new data formats: graph data, key-value stores, temporal data
- further query languages: regular path queries, streaming queries

Beyond classical OMQA:

- inconsistency-tolerant query answering
- probabilistic query answering
- privacy-aware query answering