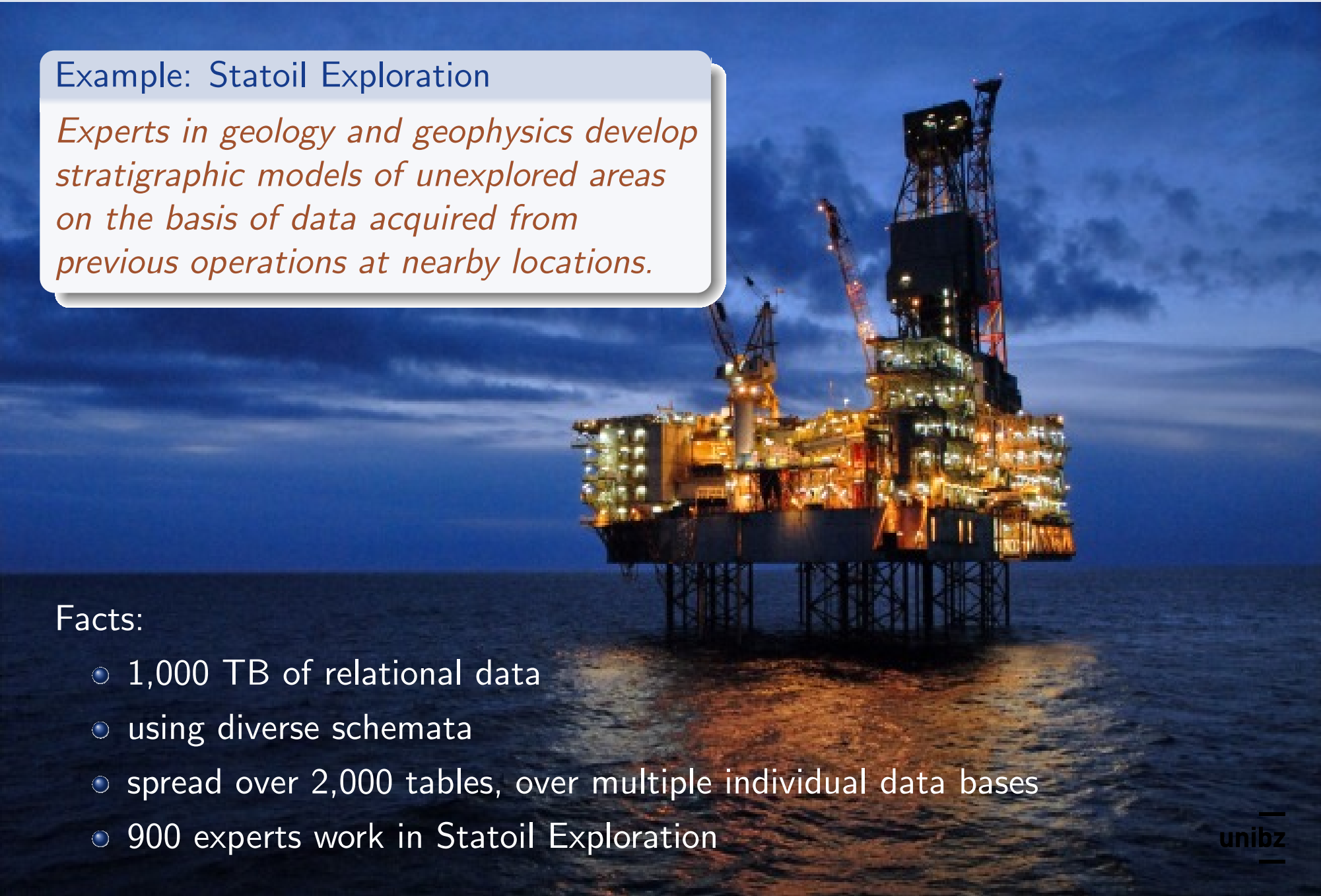
Efficient Ontology-Based Data Integration with Canonical IRIs

Guohui Xiao[1], Dag Hovland[2], Dimitris Bilidas[3],
Martin Rezk[4], Martin Giese[2], *Diego Calvanese*[1]

[1] Free University of Bozen-Bolzano, Italy
[2] University of Oslo, Norway
[3] National and Kapodistrian University of Athens, Greece
[4] Rakuten, Tokyo, Japan

15th Extended Semantic Web Conference (ESWC 2018)
Heraklion, Greece, 3–7 June 2018

Challenge: Accessing heterogeneous data



Example: Statoil Exploration

Experts in geology and geophysics develop stratigraphic models of unexplored areas on the basis of data acquired from previous operations at nearby locations.

Facts:

- 1,000 TB of relational data
- using diverse schemata
- spread over 2,000 tables, over multiple individual data bases
- 900 experts work in Statoil Exploration

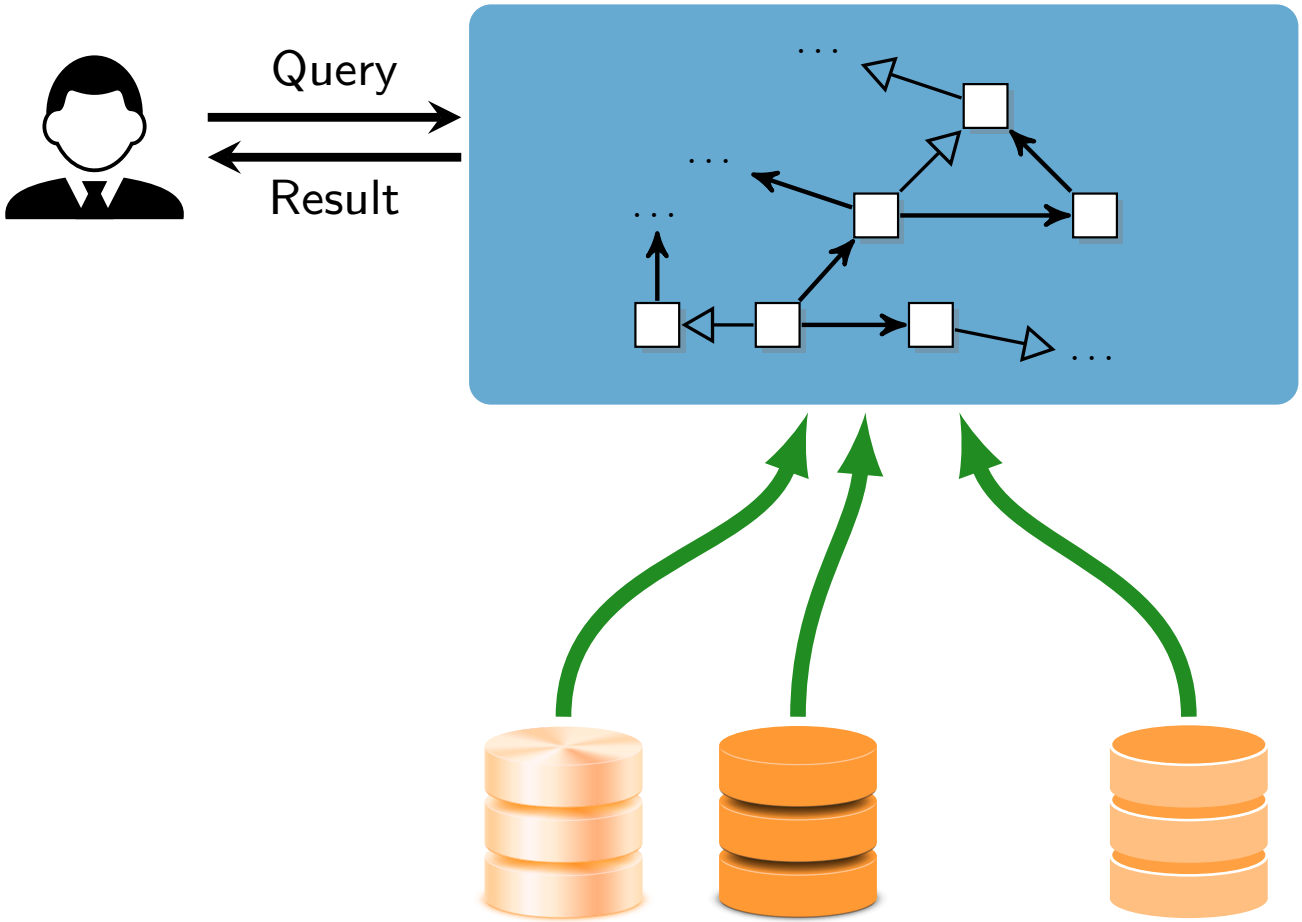
unibz

Problem: Searching for data and establishing its quality

How much time is spent searching for the right data?

Example: in oil&gas, engineers spend 30–70% of their time on this (Crompton, 2008)

Solution: Ontology-based data integration (OBDI)



Ontology
provides uniform common vocabulary and conceptual view of data

Mappings
how to populate the ontology from the data

Data Sources
external and heterogeneous

Logical transparency in accessing data sources:

- does not know where and how the data are stored.
- can only see a conceptual view of the data.

OBDI – Example

Consider two databases nat and corp with one table each (keys in blue):

nat.wellbore		
name	wbField	opPurpose
2-1	BLANE	WILDCAT
3-1		WILDCAT
3-10	OSELVAR	APPRAISAL
4-2	EKOFISK	WILDCAT

corp.drillingops		
name	driStDt	reason
NO-2-1	20-03-1989	WILDCAT
NO-3-1	06-07-1968	WILDCAT
NO-3-A	22-07-2011	PRODUCTION
NO-4-2	18-09-1969	

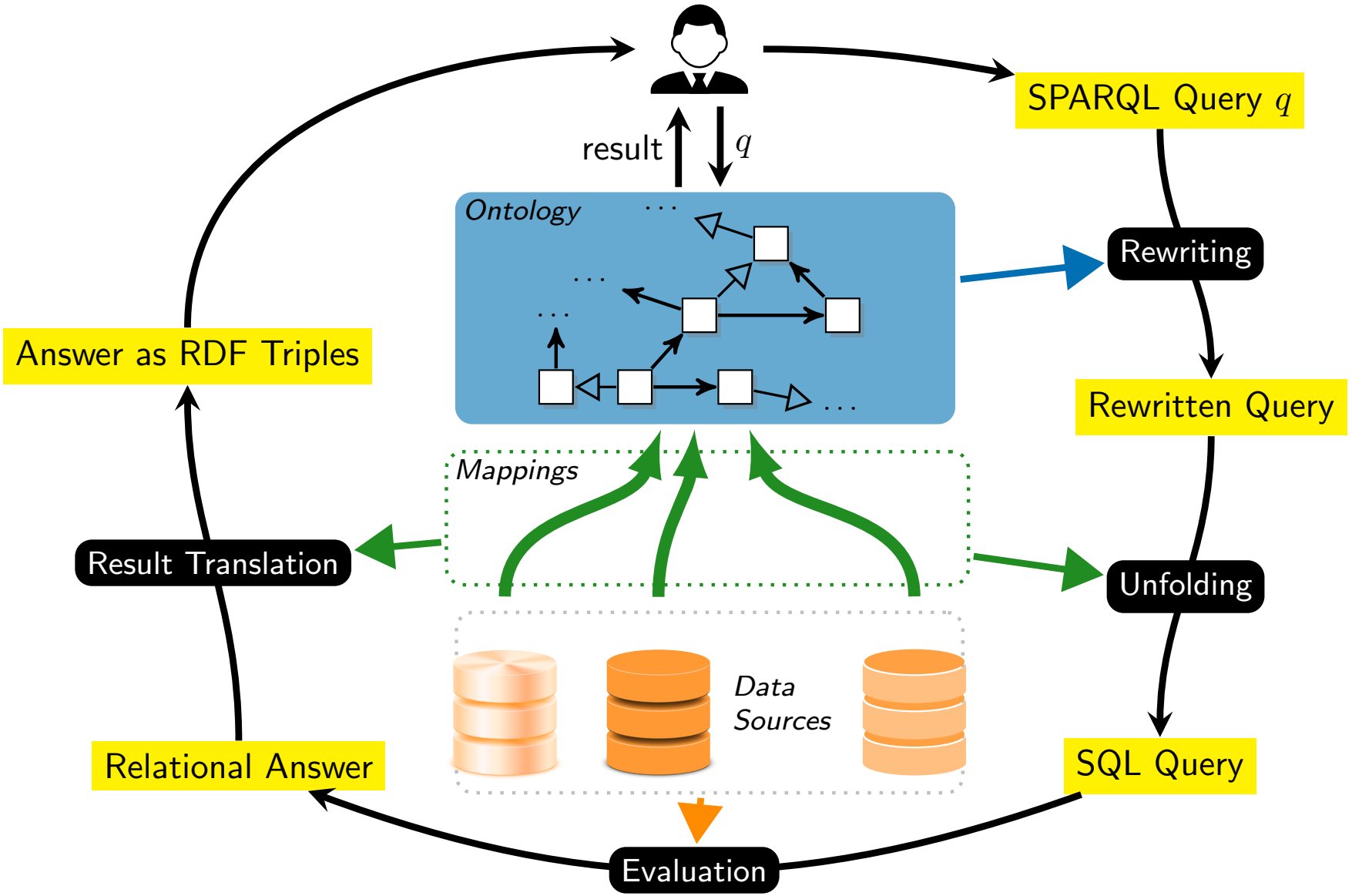
Mapping assertions: Make use of **IRI templates.**

1. `:NatWB/{name} :inField {wbField} ; :purpose {opPurp} .`
   ← `SELECT name, wbField, opPurpose FROM nat.wellbore`
2. `:CorpWB/{name} :drillingStarted {driStDt} ; :purpose {reason} .`
   ← `SELECT name, driStDt, reason FROM corp.drillingops`

Some triples in the ABox defined by the DBs and mapping

```
(:NatWB/2-1,       :inField,          BLANE)
(:NatWB/2-1,       :purpose,          WILDCAT)
(:CorpWB/NO-2-1,   :drillingStarted,  20-03-1989)
(:CorpWB/NO-2-1,   :purpose,          WILDCAT)
```

Query answering by rewriting (conceptual framework)



unibz

Problems in OBDI

The information about one real-world entity can be distributed over several data sources.

Entity resolution

Understand which records actually represent the same real world entity.

We assume that this information is available and/or known to the integration system designer.

Integrated querying

Answer queries that require to integrate data items representing the same entity, but coming from different data sources.

This is the topic of this paper.

unibz

Integrated querying – Example

nat.wellbore		
name	wbField	opPurpose
2-1	BLANE	WILDCAT
3-1		WILDCAT
3-10	OSELVAR	APPRAISAL
4-2	EKOFISK	WILDCAT

corp.drillingops		
name	driStDt	reason
NO-2-1	20-03-1989	WILDCAT
NO-3-1	06-07-1968	WILDCAT
NO-3-A	22-07-2011	PRODUCTION
NO-4-2	18-09-1969	

Some triples in the ABox defined by the DBs and mapping

```
(:NatWB/2-1,      :inField,          BLANE)
(:NatWB/2-1,      :purpose,          WILDCAT)
(:CorpWB/NO-2-1,  :drillingStarted,  20-03-1989)
(:CorpWB/NO-2-1,  :purpose,          WILDCAT)
```

Intuitively, 2-1 in nat and NO-2-1 in corp represent the same wellbore.
Hence the SPARQL query

$$((?w, \texttt{:inField}, ?f),\ \ (?w, \texttt{:drillingStarted}, ?d))$$

should return an answer, e.g.,

$$\{?w \mapsto \texttt{:NatWB/2-1},\ \ ?f \mapsto \texttt{BLANE},\ \ ?d \mapsto \texttt{20-3-1989}\}.$$

unibz

Integrated querying in OBDI

Can be achieved by *merging* the data.

Physically merge the data (as done in ETL).

- Requires full control over the data sources.
- Requires to move the data $\rightsquigarrow$ issues with freshness, privacy, legal aspects.

$\rightsquigarrow$ *Not possible in many real world scenarios!*

Virtually merge the data using `owl:sameAs` (or simply `sameAs`) and mappings [C. et al. 2015, ISWC]

- `sameAs` is the standard way of dealing with identity resolution in OWL.
- Semantics of `sameAs` may cause an exponential number of query results:
  - detrimental for performance
  - redundancy makes query answers difficult to understand

$\rightsquigarrow$ *Not feasible or desirable in practice!*

unibz

Canonical IRI assertions

Canonical IRIs

- Each entity may have several IRIs, but only **a single canonical representation**.
- This breaks the symmetry between the different representations, and avoids the exponential blowup.

We assume that the ABox $\mathcal{A}$ is augmented with a set $\mathcal{A}^c$ of **canonical IRI assertions** using the property `canIriOf`.

Example canonical IRI assertions

```
(:WB/2, canIriOf, :NatWB/2-1)
(:WB/2, canIriOf, :CorpWB/NO-2-1)
```

Assumption on $\mathcal{A}^c$: Each IRI has at most one canonical IRI.

Formally: `canIriOf` is *inverse functional* in $\mathcal{A}^c$:

$$\{\ \texttt{canIriOf}(c_1, o),\ \texttt{canIriOf}(c_2, o)\ \} \subseteq \mathcal{A}^c \quad \text{implies} \quad c_1 = c_2.$$

Canonical IRI and canonical graph

Consider a set $\mathcal{A}^c$ of canonical IRI assertions.

Canonical IRI of an individual with respect to $\mathcal{A}^c$

$$can_{\mathcal{A}^c}(o) = \begin{cases} c, & \text{if } \texttt{canIriOf}(c, o) \in \mathcal{A}^c \\ o, & \text{otherwise} \end{cases}$$

Example of canonical IRIs

For $\mathcal{A}^c = \{$ `(:WB/2, canIriOf, :NatWB/2-1)`
`(:WB/2, canIriOf, :CorpWB/NO-2-1)` $\}$
we have that $can_{\mathcal{A}^c}(\texttt{:NatWB/2-1}) = \texttt{:WB/2}$
$can_{\mathcal{A}^c}(\texttt{:CorpWB/NO-2-1}) = \texttt{:WB/2}$
$can_{\mathcal{A}^c}(\texttt{:NatWB/4-2}) = \texttt{:NatWB/4-2}$

Canonical graph $cg(\mathcal{A}, \mathcal{A}^c)$ for an ABox $\mathcal{A}$ and $\mathcal{A}^c$

$$\begin{aligned} cg(\mathcal{A}, \mathcal{A}^c) \;=\; & \{\, C(can_{\mathcal{A}^c}(o)) \mid C(o) \in \mathcal{A} \,\} \cup \\ & \{\, P(can_{\mathcal{A}^c}(o_1), can_{\mathcal{A}^c}(o_2)) \mid P(o_1, o_2) \in \mathcal{A} \,\} \end{aligned}$$

Canonical IRI entailment regime $E{+}can$

SPARQL entailment regimes

- Allow for querying RDF graphs with richer reasoning capabilities.
- An **entailment regime** E specifies how to obtain from an RDF graph G an **entailed graph** $eg^E(G)$. (*Note:* typically G is an ontology $(\mathcal{T}, \mathcal{A})$.)
- Let $[\![Q]\!]_G$ denote the answer to a SPARQL query Q over G.
  The **answer** $[\![Q]\!]^E_G$ **to** Q **under** E is defined as $[\![Q]\!]_{eg^E(G)}$.
- Note that entailment regimes only modify the evaluation of BGPs, but not that of other SPARQL operators.

Canonical IRI entailment regime

Let E be an ER in which `canIriOf` is not treated specially.
The **canonical IRI entailment regime** $E{+}can$ is defined as:

$$eg^{E+can}(\mathcal{T}, \mathcal{A} \cup \mathcal{A}^c) \;=\; eg^E(\mathcal{T}, cg(\mathcal{A}, \mathcal{A}^c))$$

Note: can is defined in a layered way, and comes "before" the ER E.

unibz

Relationship between canIriOf and sameAs

Consider:

- a set $\mathcal{A}^c$ of canIriOf assertions,
- a set $\mathcal{A}^s$ of sameAs assertions,
- the reflexive, symmetric, and transitive closure $(\mathcal{A}^s)^*$ of $\mathcal{A}^s$.

We say that $\mathcal{A}^c$ is **compliant with** $\mathcal{A}^s$ if the following hold:

- If $\texttt{canIriOf}(o_1, o_2) \in \mathcal{A}^c$, then $\texttt{sameAs}(o_1, o_2) \in (\mathcal{A}^s)^*$.
- If $\texttt{sameAs}(o_1, o_2) \in (\mathcal{A}^s)^*$, then $can_{\mathcal{A}^c}(o_1) = can_{\mathcal{A}^c}(o_2)$.

Example of $\mathcal{A}^c$ compliant with $\mathcal{A}^s$

$$
\begin{aligned}
\mathcal{A}^s \;=\; \{\; & (\texttt{:NatWB/2},\ \texttt{sameAs},\ \texttt{:CorpWB/2}),\\
& (\texttt{:NatWB/3},\ \texttt{sameAs},\ \texttt{:CorpWB/3})\ \}\\
\mathcal{A}^c \;=\; \{\; & (\texttt{:WB/2},\ \texttt{canIriOf},\ \texttt{:NatWB/2-1}),\\
& (\texttt{:WB/2},\ \texttt{canIriOf},\ \texttt{:CorpWB/NO-2-1}),\\
& (\texttt{:WB/3},\ \texttt{canIriOf},\ \texttt{:NatWB/3-1}),\\
& (\texttt{:WB/3},\ \texttt{canIriOf},\ \texttt{:CorpWB/NO-3-1})\ \}
\end{aligned}
$$

Notice that we have introduced new IRIs WB/2, WB/3 as the canonical ones.

Equivalence between canonical IRI and sameAs ERs

Consider:

- $\mathcal{A}^c$ is compliant with $\mathcal{A}^s$,
- an ER E that does not imply new `sameAs` or `canIriOf` assertions, and
- a SPARQL query Q not containing `sameAs` or `canIriOf`.

Theorem

$$[\![Q]\!]^{E+sameAs}_{\mathcal{T},\mathcal{A}\cup\mathcal{A}^s} \quad \text{corresponds to} \quad [\![cg(Q,\mathcal{A}^c)]\!]^{E+can}_{\mathcal{T},\mathcal{A}\cup\mathcal{A}^c}$$

More formally:

- If $\{?x_1 \mapsto o_1, \ldots, ?x_n \mapsto o_n\} \in [\![Q]\!]^{E+sameAs}_{\mathcal{T},\mathcal{A}\cup\mathcal{A}^s}$, then
$$\{?x_1 \mapsto can_{\mathcal{A}^c}(o_1), \ldots, ?x_n \mapsto can_{\mathcal{A}^c}(o_n)\} \in [\![cg(Q,\mathcal{A}^c)]\!]^{E+can}_{\mathcal{T},\mathcal{A}\cup\mathcal{A}^c}.$$
- If $\{?x_1 \mapsto o_1, \ldots, ?x_n \mapsto o_n\} \in [\![Q]\!]^{E+can}_{\mathcal{T},\mathcal{A}\cup\mathcal{A}^c}$, and $\texttt{sameAs}(o_j, o'_j) \in (\mathcal{A}^s)^*$, for $1 \leq j \leq n$, then
$$\{?x_1 \mapsto o'_1, \ldots, x_n \mapsto o'_n\} \in [\![Q]\!]^{E+sameAs}_{\mathcal{T},\mathcal{A}\cup\mathcal{A}^s}.$$

unibz

Handling canonical IRI semantics by query rewriting

Our aim is to answer SPARQL queries under the canonical IRI ER efficiently.
We rely on a common approach to deal with ERs: **query rewriting**

Objective

Rewrite each SPARQL query Q (under the $E{+}can$ ER) to a SPARQL query R (under a standard ER E) that is equivalent to Q, i.e., such that:

$$[\![Q]\!]^{E+can}_{\mathcal{T},\mathcal{A}\cup\mathcal{A}^c} = [\![R]\!]^{E}_{\mathcal{T},\mathcal{A}\cup\mathcal{A}^c}$$

Observation: BGP queries under canonical IRI semantics are non-monotonic.

Example: consider: $G = \{(a, \texttt{rdf:type}, C)\}$, and $Q = (?x, \texttt{rdf:type}, C)$.
Then: $\{?x \mapsto a\} \in [\![Q]\!]^{can}_{G}$.

But for $\Delta G = \{(c, \texttt{canIriOf}, a)\}$, we have $\{?x \mapsto a\} \notin [\![Q]\!]^{can}_{G\cup\Delta G}$.

Hence, we cannot equivalently rewrite a BGP under the canonical IRI semantics to a union of BGPs under the standard SPARQL semantics and expect to obtain the same answers.

Rewriting using NOT EXISTS

We first introduce the subquery $qc[?xc, ?x]$ returning $can_{\mathcal{A}^c}(?x)$ in $?xc$.
To do so, we simply encode the definition of $can_{\mathcal{A}^c}$:

$$
\begin{aligned}
qc[?xc, ?x] = \texttt{SELECT}\ & ?xc, ?x\ \texttt{WHERE} \\
& ((?xc, \texttt{canIriOf}, ?x) \\
& \quad \texttt{UNION} \\
& (\texttt{SELECT}\ ?x\ \texttt{AS}\ ?xc, ?x\ \texttt{WHERE}\ ((?x, ?p_1, ?o)\ \texttt{UNION}\ (?s, ?p_2, ?x)) \\
& \ \texttt{FILTER NOT EXISTS}\ (?y, \texttt{canIriOf}, ?x)))
\end{aligned}
$$

This query correctly computes the canonical IRI for an individual o, i.e.:

$$\{?xc \mapsto c, ?x \mapsto o\} \in [\![qc[?xc, ?x]]\!]_{\mathcal{A} \cup \mathcal{A}^c} \quad \text{iff} \quad c = can_{\mathcal{A}^c}(o).$$

unibz

Canonical-iri rewriting of a SPARQL query

We use $qc[?xc, ?x]$ to compute the canonical-iri rewriting of a SPARQL query:

The canonical-iri rewriting $\psi(Q)$ of a SPARQL query Q

is obtained from Q by replacing each triple pattern t with the sub-query $\psi(t)$ obtained from t as follows:

1. for each variable $?x$, introduce a fresh variable $?xc$,
2. for each occurrence of a variable $?x$, introduce a fresh variable $?xo$, change $?x$ to $?xo$, and join with $qc[?xc, ?xo]$, and
3. for each variable $?x$, add a projection $?xc$ `AS` $?x$.

Example

Let $Q =$ `SELECT` $?w$, $?f$, $?d$ `WHERE` $((?w, \texttt{:inField}, ?f), (?w, \texttt{:drillingStarted}, ?d))$
Then $\psi(Q) =$ `SELECT` $?w$, $?f$, $?d$ `WHERE` (
(`SELECT` $?wc1$ `AS` $?w$, $?fc$ `AS` $?f$ `WHERE`
$((?wo1, \texttt{:inField}, ?fo),\ qc[?wc1, ?wo1],\ qc[?fc, ?fo])$),
(`SELECT` $?wc2$ `AS` $?w$, $?dc$ `AS` $?d$ `WHERE`
$((?wo2, \texttt{:drillingStarted}, ?do),\ qc[?wc2, ?wo2],\ qc[?dc, ?do])$))

Correctness of canonical-iri rewriting

Theorem

For any E, $\mathcal{T}$, $\mathcal{A}$, $\mathcal{A}^c$, and Q, we have that

$$[\![\psi(Q)]\!]^{E}_{\mathcal{T},\mathcal{A}\cup\mathcal{A}^c} \;=\; [\![Q]\!]^{E+can}_{\mathcal{T},\mathcal{A}\cup\mathcal{A}^c}.$$

I.e., the rewritten query $\psi(Q)$ under the ordinary ER E returns the same answers as the original query Q under the canonical-iri ER $E{+}can$.

The size of the rewritten query is linear in the size of the original one.

However, this approach is impractical.

- The number of variables in the `NOT EXISTS` clauses and in the inner `UNION` clauses is linear in the number of variables in the original query.
- For each such variable, a (naive) query execution engine would need to enumerate over all individuals in the ABox.
- Hence, the query execution time would essentially grow exponentially in the number of variables in the query.

Handling canonical IRI statements in OBDI

- We propose a practical approach for canonical IRI semantics in OBDI.
- We assume that the mapping $\mathcal{M}$ includes **assertions $\mathcal{M}^c$ populating** `canIriOf`.
- The mapping $\mathcal{M}^c$ may be fed from *master tables*, typical of many corporate scenarios (e.g., as in Statoil).
- However, for $\mathcal{M}^c$ we do not rely on master tables, and may use arbitrary SQL queries to ordinary tables.

Example master table and mapping

central.masterTable		
id	natName	corpName
2	2-1	NO-2-1
3	3-1	NO-3-1
4	4-2	NO-4-2
5		NO-3-A
6	3-10	

- `:WB/{id} canIriOf :NatWB/{natName} .`
  ← `SELECT id, natName FROM central.masterTable`
- `:WB/{id} canIriOf :CorpWB/{corpName} .`
  ← `SELECT id, corpName FROM central.masterTable`

Mapping rewriting to deal with canonical IRIs

- We propose a practical method based on compiling the consequences of canonical IRI semantics into mappings $\rightsquigarrow$ **Mapping rewriting**
- Inspired by the mapping saturation algorithm in classical OBDA.
- We need to ensure inverse functionality of `canIriOf`.

Assumption on the mappings

For each IRI template iri, at most one mapping assertion in $\mathcal{M}^c$ of the form:

$$iri_c(\vec{a})\ \texttt{canIriOf}\ iri(\vec{b})\ \leftarrow\ \mathtt{sql}(\vec{a},\vec{b})$$

Note:

- This assumption suffices: if $\mathcal{M}^c$ satisfies it, then for every database D, `canIriOf` is inverse functional in the extracted (virtual) ABox $\mathcal{A}_{\mathcal{M}^c,D}$.
- Is stronger than inverse functionality of `canIriOf`.
- But is reasonable in practice.

unibz

Mapping rewriting algorithm

To rewrite the mapping, we replace individuals and IRI-templates in the mapping by their canonical representation.

Let $\mathcal{M} = \mathcal{M}^t \cup \mathcal{M}^c$ be a set of mapping assertions.

Canonical-iri rewriting $cm(\mathcal{M}^t, \mathcal{M}^c)$ of $\mathcal{M}$

Is obtained by processing each mapping assertion $ma \in \mathcal{M}^t$ as follows:

1. For each IRI template $iri(\vec{a})$ in ma, if $\mathcal{M}^c$ contains a mapping assertion
$$iri_c(\vec{b_0}) \texttt{ canIriOf } iri(\vec{b_1}) \leftarrow \mathtt{sql}(\vec{b_0}, \vec{b_1})$$
then replace $iri(\vec{a})$ in the target of ma by $iri_c(\vec{b_0})$, and
join the source query of ma with $\mathtt{sql}(\vec{b_0}, \vec{b_1}), \vec{a} = \vec{b_1}$.
2. Process IRIs directly occurring in ma in the same way.

unibz

Mapping rewriting – Example

Mapping $\mathcal{M}^t$

1. `:NatWB/{name} :inField {wbField} ; :purpose {opPurp} .`
   `← SELECT name, wbField, opPurpose FROM nat.wellbore`
2. `:CorpWB/{name} :drillingStarted {driStDt} ; :purpose {reason} .`
   `← SELECT name, driStDt, reason FROM corp.drillingops`

Mapping $\mathcal{M}^c$

1. `:WB/{id} canIriOf :NatWB/{natName} .`
   `← SELECT id, natName FROM central.masterTable`
2. `:WB/{id} canIriOf :CorpWB/{corpName} .`
   `← SELECT id, corpName FROM central.masterTable`

Canonical-iri rewriting $cm(\mathcal{M}^t, \mathcal{M}^c)$ of $\mathcal{M}^t \cup \mathcal{M}^c$

1. `:WB/{id} :inField {wlbField} ; :purpose {opPurp} .`
   `← SELECT wlbFld, opPurp, id`
   `FROM nat.wellbore, central.masterTable WHERE name = natName`
2. `:WB/{id} :drillingStarted {driStDt} ; :purpose {reason} .`
   `← SELECT driStDt, reason, id`
   `FROM corp.drillingops, central.masterTable WHERE name = corpName`

Correctness of mapping rewriting

The mapping rewriting algorithm has the same effect as applying cg to the RDF graph: for a mapping $\mathcal{M} = \mathcal{M}^t \cup \mathcal{M}^c$ and a database instance D, we have:

$$cg(\mathcal{A}_{\mathcal{M}^t,D}, \mathcal{A}_{\mathcal{M}^c,D}) \;=\; \mathcal{A}_{cm(\mathcal{M}^t,\mathcal{M}^c),D}.$$

It follows that the mapping rewriting algorithm is sound and complete.

Theorem

Let E be an entailment regime, $\mathcal{T}$ a TBox, $\mathcal{M} = \mathcal{M}^t \cup \mathcal{M}^c$ an OBDI mapping, D a database instance, and Q a SPARQL query. Then

$$[\![Q]\!]^{E+can}_{\mathcal{T},\mathcal{A}_{\mathcal{M},D}} \;=\; [\![Q]\!]^{E}_{\mathcal{T},\mathcal{A}_{cm(\mathcal{M}^t,\mathcal{M}^c),D}}$$

unibz

Implemented in the *Ontop* system

- OBDA platform developed at the KRDB Research Centre of the Free University of Bozen-Bolzano.
- Supports all major database engines (e.g., Oracle, DB2, MS SQL Server, PostgreSQL, MySQL).
- Open source under Apache 2 License

`http://ontop.inf.unibz.it/`

unibz

Results over Statoil query catalog

We have applied this approach in the Statoil use case:

- 7 data sources: DDR, Compass, Slegge, Recall, CoreDB, GeoChemDB, and OpenWorks
- We have exploited existing *master tables*.
- The mappings for canonical IRIs are simple mappings into these tables.
- Query catalog with 76 challenging SPARQL queries constructed from information needs by geologists and geoscientists.

Results:

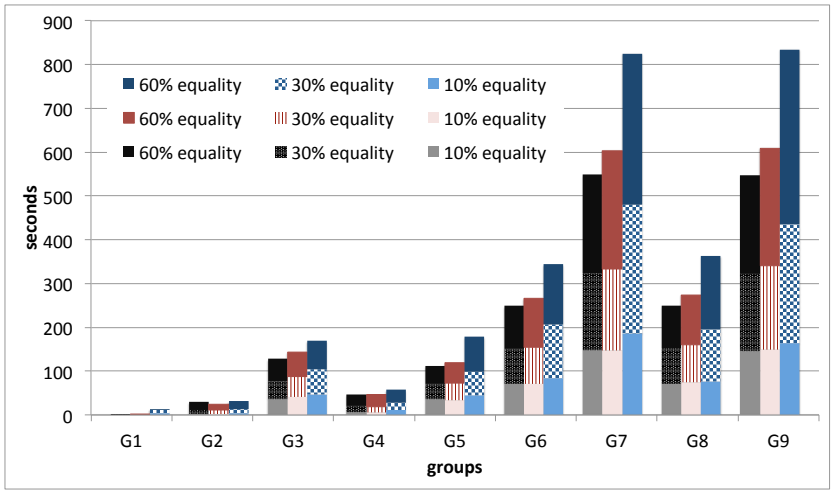
	sameAs	canonical IRI
Total queries	76	76
Timeouts	31	11
Successful	45	65
Success %	59%	85%
Min exec. time	12s	0.50s
Mean exec. time	11m	4.3m
Median exec. time	11m	0.77m

(limit = 100K tuples, timeout = 20 minutes)

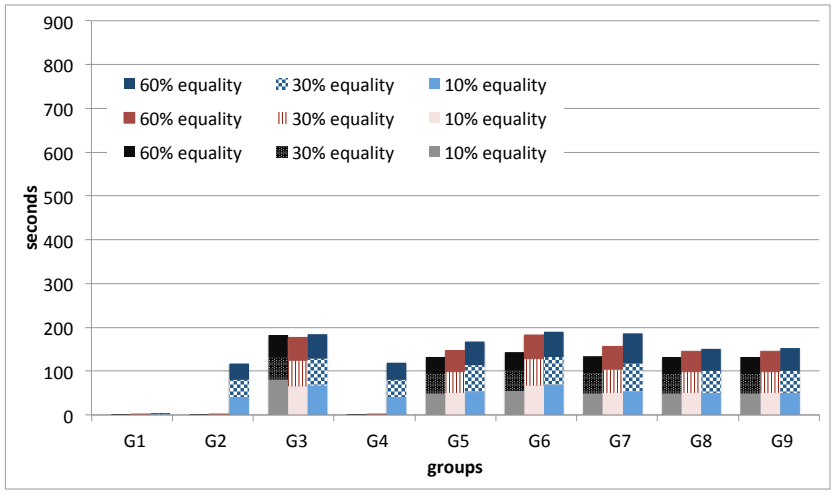
unibz

Results over benchmark data

2 datasets: execution times of most expensive queries

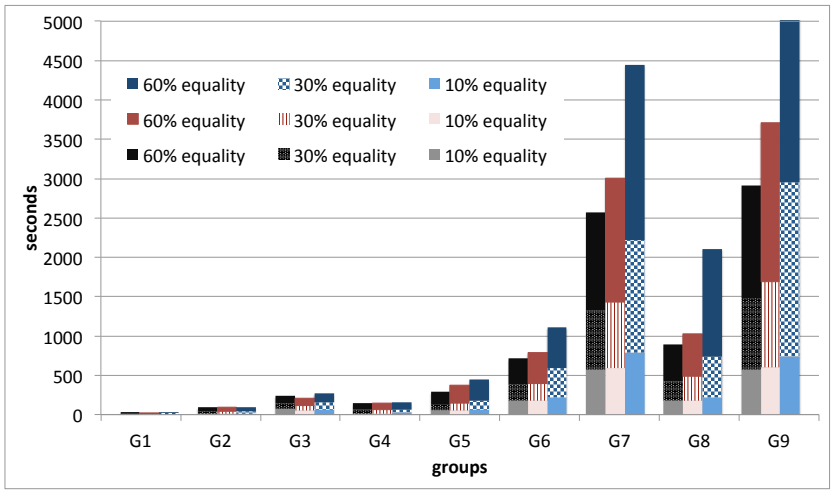


Standard `owl:sameAs`

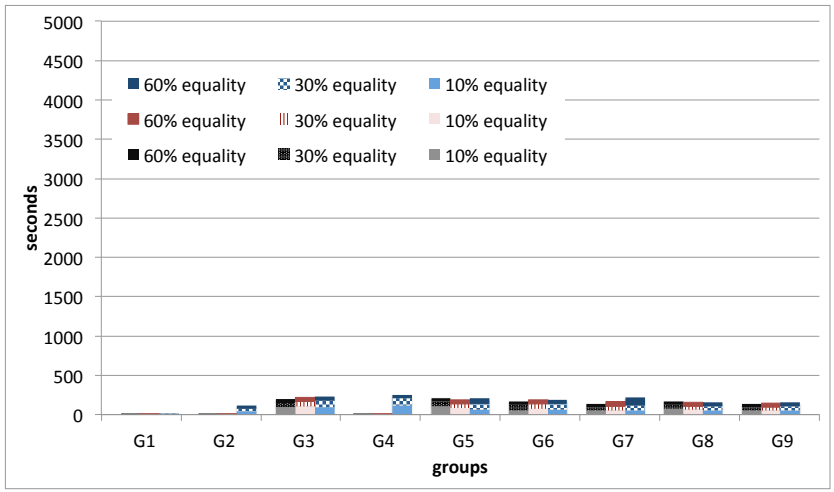


Canonical IRI

3 datasets: execution times of most expensive queries



Standard `owl:sameAs`



Canonical IRI

unibz

Conclusions

- The proposed approach allows for a simple implementation of the canonical IRI semantics:
  - The canonical-iri rewriting step is embedded into the startup phase of an OBDA system.
- However, effort is shifted from rewriting to mapping construction.
- Compared with the `sameAs` approach, more information must be encoded in the mappings:
  - encode transitivity
  - choose a representative for each IRI equivalence class

unibz

Thanks

Thank you for your attention!

unibz

References I

[1] Diego C. et al. “Ontology-based Integration of Cross-linked Datasets”. In: *Proc. of the 14th Int. Semantic Web Conf. (ISWC)*. Vol. 9366. Lecture Notes in Computer Science. Springer, 2015, pp. 199–216. DOI: 10.1007/978-3-319-25007-6_12.

unibz