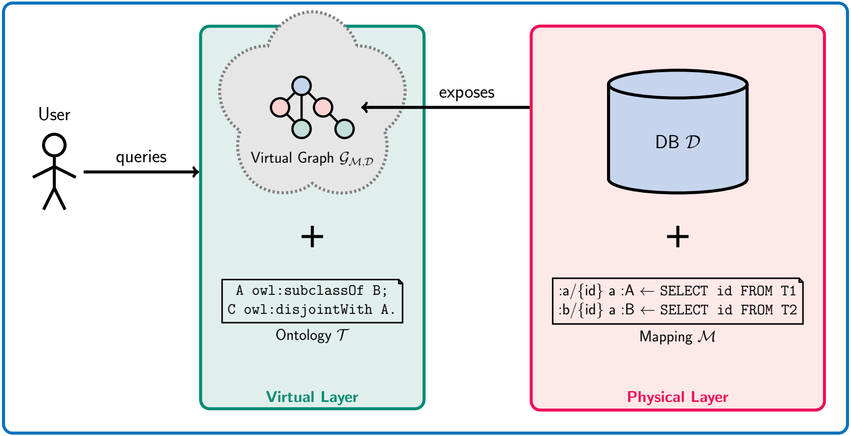
Cost-driven OBDA

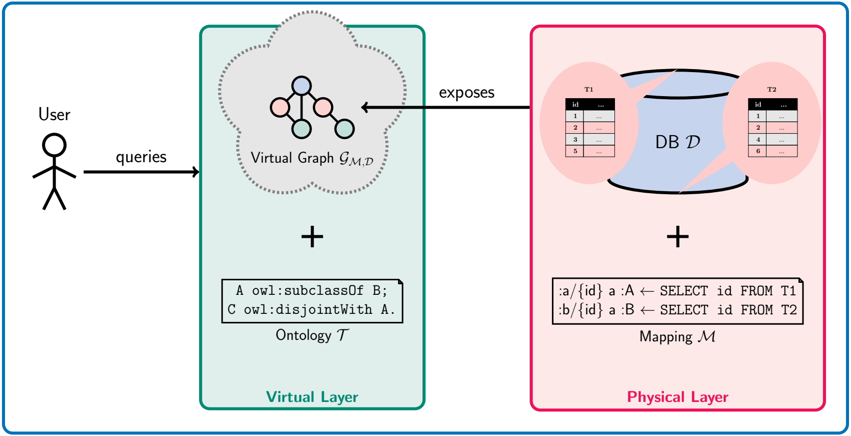
ISWC 2017
Vienna, Austria

- **Speaker:** **Davide Lanti** `dlanti@inf.unibz.it`
- **Joint Work With:** **Guohui Xiao, Diego Calvanese**
- **Institution:** **Free University of Bozen-Bolzano**

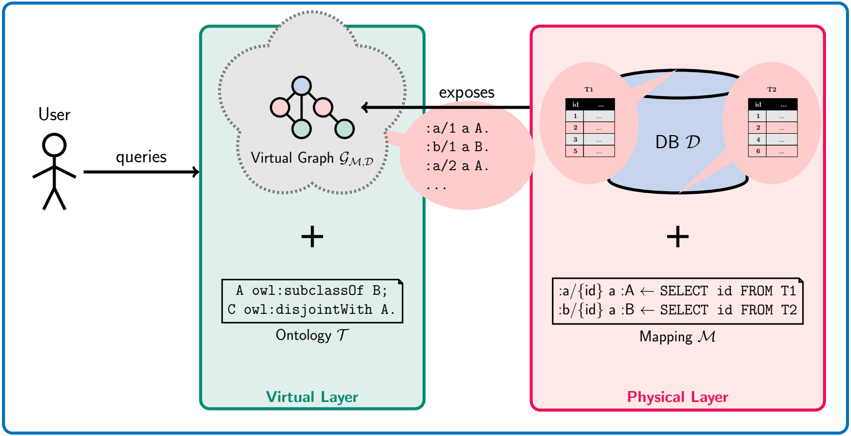
An OBDA Scenario (Semantic Web View)



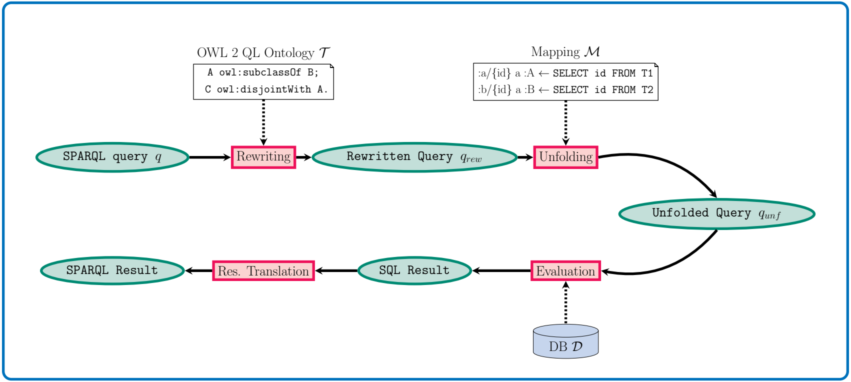
An OBDA Scenario (Semantic Web View)



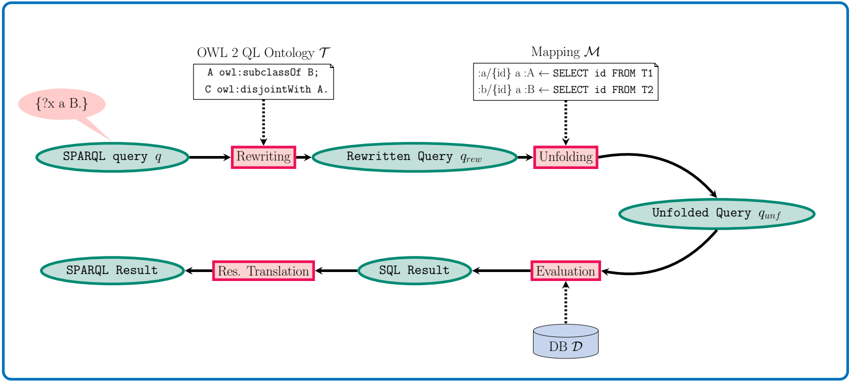
An OBDA Scenario (Semantic Web View)



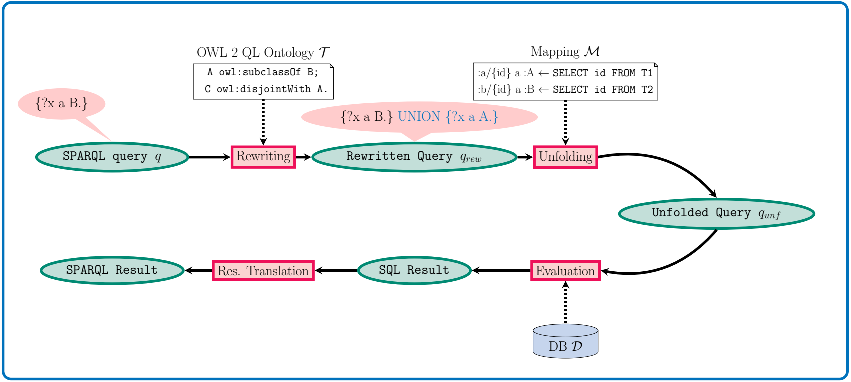
Query Evaluation Process I



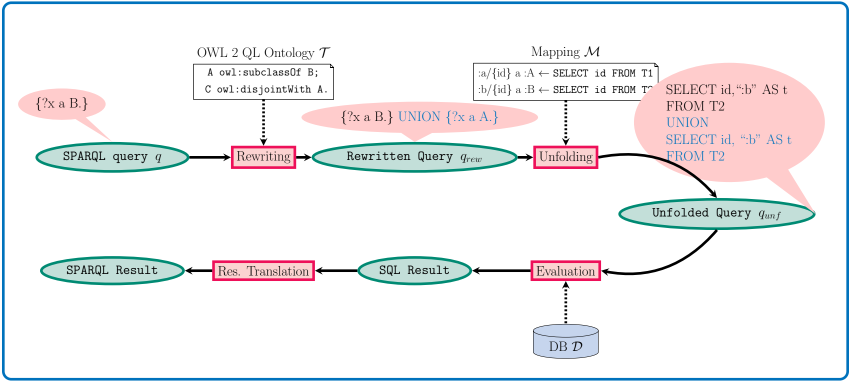
Query Evaluation Process II



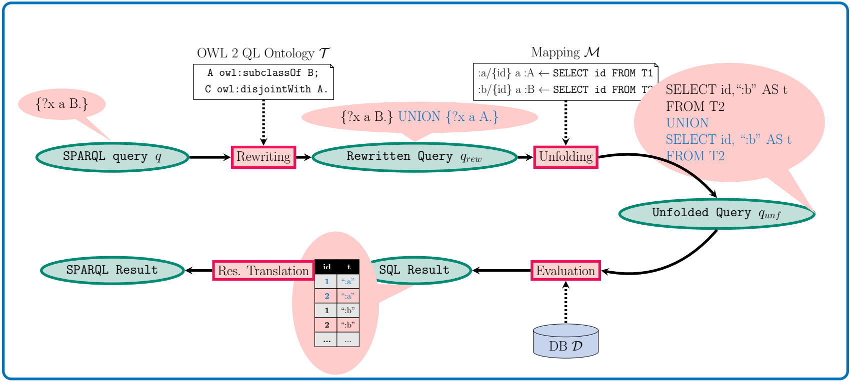
Query Evaluation Process III



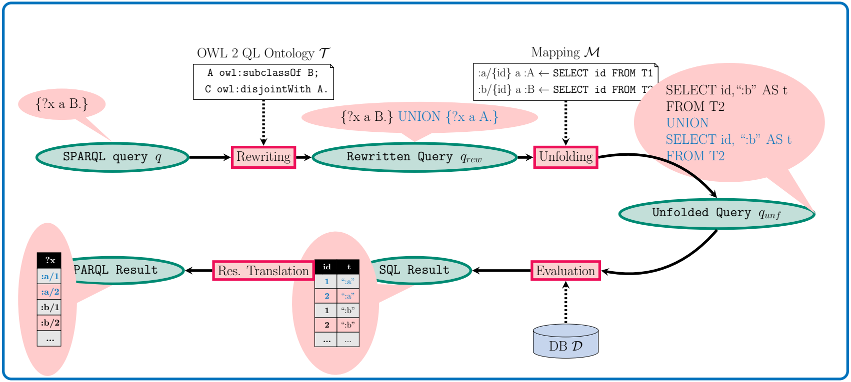
Query Evaluation Process IV



Query Evaluation Process V



Query Evaluation Process VI



The Performance Problem

- **Problem:**
  - **Performance is an issue in real-world applications of OBDA**
- **Current Solutions:**
  - **Data-independent Optimizations**
    - **Structural Optimizations**
    - **Semantic Query Optimizations**

The Performance Problem

- **Problem:**
  - **Performance is an issue in real-world applications of OBDA**
- **Current Solutions:**
  - **Data-independent Optimizations**
    - **Structural Optimizations**
    - **Semantic Query Optimizations**
- **Contribution of this work:**
  - **Data-dependent Optimizations**
    - **Explore a space of alternatives for unfoldings**
    - **Choose the best alternative, given the instance**

The Performance Problem

- **Problem:**
  - **Performance is an issue in real-world applications of OBDA**
- **Current Solutions:**
  - **Data-independent Optimizations**
    - **Structural Optimizations**
    - **Semantic Query Optimizations**
- **Contribution of this work:**
  - **Data-dependent Optimizations**
    - **Explore a space of alternatives for unfoldings**
    - **Choose the best alternative, given the instance**

Unfolding and Structural Optimizations I

Mapping $\mathcal{M}$

% High-Level Mappings *target $\leftsquigarrow$ source view*

$$
\begin{array}{lcl}
URI_1(\vec{x})\ a\ A & \leftsquigarrow & V_1(\vec{x}) \\
URI_2(\vec{x})\ a\ A & \leftsquigarrow & V_2(\vec{x}) \\
URI_1(\vec{x})\ R\ URI_3(\vec{y}) & \leftsquigarrow & V_3(\vec{x}, \vec{y}) \\
URI_1(\vec{x})\ R\ URI_4(\vec{y}) & \leftsquigarrow & V_4(\vec{x}, \vec{y})
\end{array}
$$

% Low-Level Mappings *source view $\leftsquigarrow$ source query*

$$
\begin{array}{lcl}
V_1(\vec{x}) & \leftsquigarrow & CQ_1(\vec{x}) \\
V_2(\vec{x}) & \leftsquigarrow & CQ_2(\vec{x}) \\
V_3(\vec{x}, \vec{y}) & \leftsquigarrow & CQ_3(\vec{x}, \vec{y}) \\
V_4(\vec{x}, \vec{y}) & \leftsquigarrow & CQ_4(\vec{x}, \vec{y})
\end{array}
$$

Unfolding and Structural Optimizations I

Mapping $\mathcal{M}$

% High-Level Mappings *target $\leftsquigarrow$ source view*

$$
\begin{array}{lcl}
URI_1(\vec{x})\ a\ A & \leftsquigarrow & V_1(\vec{x}) \\
URI_2(\vec{x})\ a\ A & \leftsquigarrow & V_2(\vec{x}) \\
URI_1(\vec{x})\ R\ URI_3(\vec{y}) & \leftsquigarrow & V_3(\vec{x}, \vec{y}) \\
URI_1(\vec{x})\ R\ URI_4(\vec{y}) & \leftsquigarrow & V_4(\vec{x}, \vec{y})
\end{array}
$$

% Low-Level Mappings *source view $\leftsquigarrow$ source query*

$$
\begin{array}{lcl}
V_1(\vec{x}) & \leftsquigarrow & CQ_1(\vec{x}) \\
V_2(\vec{x}) & \leftsquigarrow & CQ_2(\vec{x}) \\
V_3(\vec{x}, \vec{y}) & \leftsquigarrow & CQ_3(\vec{x}, \vec{y}) \\
V_4(\vec{x}, \vec{y}) & \leftsquigarrow & CQ_4(\vec{x}, \vec{y})
\end{array}
$$

Query q

SELECT ?x ?y WHERE {?x a A. ?x R ?y.}

Unfolding and Structural Optimizations II

Query q

SELECT ?x ?y WHERE {?x a A. ?x R ?y.}

SELECT CONCAT

We introduce auxiliary notation:

$$
\begin{aligned}
SC_1(V_1) &= \text{SELECT CONCAT}('URI_1(', \vec{x}, ')') \text{ AS } x \text{ FROM } V_1 \\
&\vdots \\
SC_{1,4}(V_4) &= \text{SELECT CONCAT}('URI_1(', \vec{x}, ')') \text{ AS } x, \\
&\quad \text{CONCAT}('URI_4(', \vec{y}, ')') \text{ AS } y \text{ FROM } V_4
\end{aligned}
$$

Unfolding and Structural Optimizations II

Query q

SELECT ?x ?y WHERE { ?x a A. ?x R ?y.}

SELECT CONCAT

We introduce auxiliary notation:

$$SC_1(V_1) = \textit{SELECT CONCAT}('URI_1(', \vec{x}, ')') \textit{ AS } x \textit{ FROM } V_1$$

$$\vdots$$

$$SC_{1,4}(V_4) = \textit{SELECT CONCAT}('URI_1(', \vec{x}, ')') \textit{ AS } x, \textit{CONCAT}('URI_4(', \vec{y}, ')') \textit{ AS } y \textit{ FROM } V_4$$

Intermediate Unfolding (Similar to [Chebotko et al., 2009])

$$(SC_1(V_1) \cup SC_2(V_2)) \bowtie (SC_{1,3}(V_3) \cup SC_{1,4}(V_4))$$

Unfolding and Structural Optimizations III

Structural Optimization 1: Transform to UCQ

$$
\begin{array}{lcl}
SC_1(V_1) & \bowtie & SC_{1,3}(V_3) \\
SC_1(V_1) & \bowtie & SC_{1,4}(V_4) \\
SC_2(V_2) & \bowtie & SC_{1,3}(V_3) \\
SC_2(V_2) & \bowtie & SC_{1,4}(V_4)
\end{array}
$$

Unfolding and Structural Optimizations III

Structural Optimization 1: Transform to UCQ

$$\begin{array}{lll} \boldsymbol{SC_1}(\boldsymbol{V_1}) & \bowtie & \boldsymbol{SC_{1,3}}(\boldsymbol{V_3}) \\ \boldsymbol{SC_1}(\boldsymbol{V_1}) & \bowtie & \boldsymbol{SC_{1,4}}(\boldsymbol{V_4}) \\ \boldsymbol{SC_2}(\boldsymbol{V_2}) & \bowtie & \boldsymbol{SC_{1,3}}(\boldsymbol{V_3}) \\ \boldsymbol{SC_2}(\boldsymbol{V_2}) & \bowtie & \boldsymbol{SC_{1,4}}(\boldsymbol{V_4}) \end{array}$$

$\boldsymbol{URI_2} \neq \boldsymbol{URI_1}$ ***!! (Incompatible URIs)***

Unfolding and Structural Optimizations III

Structural Optimization 1: Transform to UCQ

$$\begin{array}{lcl} SC_1(V_1) & \bowtie & SC_{1,3}(V_3) \\ SC_1(V_1) & \bowtie & SC_{1,4}(V_4) \\ SC_2(V_2) & \bowtie & SC_{1,3}(V_3) \\ SC_2(V_2) & \bowtie & SC_{1,4}(V_4) \end{array} \Bigg\} \; URI_2 \neq URI_1 \text{ !! (Incompatible URIs)}$$

Structural Optimization 2: Remove joins between incompatible URIs

$$\begin{array}{lcl} SC_1(V_1) & \bowtie & SC_{1,3}(V_3) \\ SC_1(V_1) & \bowtie & SC_{1,4}(V_4) \end{array}$$

Unfolding and Structural Optimizations III

Structural Optimization 1: Transform to UCQ

$$\begin{array}{lll} SC_1(V_1) & \bowtie & SC_{1,3}(V_3) \\ SC_1(V_1) & \bowtie & SC_{1,4}(V_4) \\ SC_2(V_2) & \bowtie & SC_{1,3}(V_3) \\ SC_2(V_2) & \bowtie & SC_{1,4}(V_4) \end{array}$$

$URI_2 \neq URI_1$ *!! (Incompatible URIs)*

Structural Optimization 2: Remove joins between incompatible URIs

$$\begin{array}{lll} SC_1(V_1) & \bowtie & SC_{1,3}(V_3) \\ SC_1(V_1) & \bowtie & SC_{1,4}(V_4) \end{array}$$

Structural Optimization 3: Remove Joins over templates

$$SC_{1,3}(V_1 \bowtie V_3) \cup SC_{1,4}(V_1 \bowtie V_4)$$

Unfolding and Structural Optimizations III

Structural Optimization 1: Transform to UCQ

$$\begin{array}{lll} SC_1(V_1) & \bowtie & SC_{1,3}(V_3) \\ SC_1(V_1) & \bowtie & SC_{1,4}(V_4) \\ SC_2(V_2) & \bowtie & SC_{1,3}(V_3) \\ SC_2(V_2) & \bowtie & SC_{1,4}(V_4) \end{array}$$

$URI_2 \neq URI_1$!! (Incompatible URIs)

Structural Optimization 2: Remove joins between incompatible URIs

$$\begin{array}{lll} SC_1(V_1) & \bowtie & SC_{1,3}(V_3) \\ SC_1(V_1) & \bowtie & SC_{1,4}(V_4) \end{array}$$

Structural Optimization 3: Remove Joins over templates

$$SC_{1,3}(V_1 \bowtie V_3) \cup SC_{1,4}(V_1 \bowtie V_4)$$

We call the unfolding after structural optimizations UCQ-unfolding

Trade-off Redundancy/Indexing

Mapping $\mathcal{M}$

% High-Level Mappings *target* $\leftsquigarrow$ *source*

$\{URI_1(\vec{x})\ a\ A \leftsquigarrow V_i(\vec{x}) \mid 1 \leqslant i \leqslant 6\}$

$\{URI_1(\vec{x})\ P\ URI_2(\vec{y}) \leftsquigarrow W_i(\vec{x}, \vec{y}) \mid 1 \leqslant i \leqslant 6\}$

$\{URI_1(\vec{x})\ R\ URI_3(\vec{z}) \leftsquigarrow Z_i(\vec{x}, \vec{z}) \mid 1 \leqslant i \leqslant 6\}$

Trade-off Redundancy/Indexing

Mapping $\mathcal{M}$

% High-Level Mappings *target* $\leftsquigarrow$ *source*

$\{URI_1(\vec{x})\ a\ A \leftsquigarrow V_i(\vec{x}) \mid 1 \leqslant i \leqslant 6\}$

$\{URI_1(\vec{x})\ P\ URI_2(\vec{y}) \leftsquigarrow W_i(\vec{x}, \vec{y}) \mid 1 \leqslant i \leqslant 6\}$

$\{URI_1(\vec{x})\ R\ URI_3(\vec{z}) \leftsquigarrow Z_i(\vec{x}, \vec{z}) \mid 1 \leqslant i \leqslant 6\}$

- **SELECT DISTINCT * WHERE {?x rdf:type A. ?x P ?y. ?x R ?z}**

Trade-off Redundancy/Indexing

Mapping $\mathcal{M}$

% High-Level Mappings *target $\leftsquigarrow$ source*

$\{URI_1(\vec{x})\ a\ A \leftsquigarrow V_i(\vec{x}) \mid 1 \leqslant i \leqslant 6\}$

$\{URI_1(\vec{x})\ P\ URI_2(\vec{y}) \leftsquigarrow W_i(\vec{x}, \vec{y}) \mid 1 \leqslant i \leqslant 6\}$

$\{URI_1(\vec{x})\ R\ URI_3(\vec{z}) \leftsquigarrow Z_i(\vec{x}, \vec{z}) \mid 1 \leqslant i \leqslant 6\}$

- SELECT DISTINCT * WHERE {?x rdf:type A. ?x P ?y. ?x R ?z}
  - **UCQ-unfolding:** $\bigcup_{i,j,k \in \{1,\ldots,6\}} SC(V_i \bowtie W_j \bowtie Z_k)$

Trade-off Redundancy/Indexing

Mapping $\mathcal{M}$

% High-Level Mappings *target $\leftsquigarrow$ source*

$\{URI_1(\vec{x})\ a\ A \leftsquigarrow V_i(\vec{x}) \mid 1 \leqslant i \leqslant 6\}$

$\{URI_1(\vec{x})\ P\ URI_2(\vec{y}) \leftsquigarrow W_i(\vec{x}, \vec{y}) \mid 1 \leqslant i \leqslant 6\}$

$\{URI_1(\vec{x})\ R\ URI_3(\vec{z}) \leftsquigarrow Z_i(\vec{x}, \vec{z}) \mid 1 \leqslant i \leqslant 6\}$

- SELECT DISTINCT * WHERE {?x rdf:type A. ?x P ?y. ?x R ?z}
  - **UCQ-unfolding:** $\bigcup_{i,j,k \in \{1,\dots,6\}} SC(V_i \bowtie W_j \bowtie Z_k)$
    - $6^3 = 216$ CQs

Trade-off Redundancy/Indexing

Mapping $\mathcal{M}$

% High-Level Mappings *target* $\leftsquigarrow$ *source*

$\{URI_1(\vec{x})\ a\ A \leftsquigarrow V_i(\vec{x}) \mid 1 \leqslant i \leqslant 6\}$

$\{URI_1(\vec{x})\ P\ URI_2(\vec{y}) \leftsquigarrow W_i(\vec{x}, \vec{y}) \mid 1 \leqslant i \leqslant 6\}$

$\{URI_1(\vec{x})\ R\ URI_3(\vec{z}) \leftsquigarrow Z_i(\vec{x}, \vec{z}) \mid 1 \leqslant i \leqslant 6\}$

- SELECT DISTINCT * WHERE {?x rdf:type A. ?x P ?y. ?x R ?z}
  - **UCQ-unfolding:** $\bigcup_{i,j,k \in \{1,\ldots,6\}} SC(V_i \bowtie W_j \bowtie Z_k)$
    - $6^3 = 216$ CQs
    - **Redundancy?**

Trade-off Redundancy/Indexing

Mapping $\mathcal{M}$

% High-Level Mappings *target $\rightsquigarrow$ source*

$\{URI_1(\vec{x})\ a\ A \leftsquigarrow V_i(\vec{x}) \mid 1 \leqslant i \leqslant 6\}$

$\{URI_1(\vec{x})\ P\ URI_2(\vec{y}) \leftsquigarrow W_i(\vec{x}, \vec{y}) \mid 1 \leqslant i \leqslant 6\}$

$\{URI_1(\vec{x})\ R\ URI_3(\vec{z}) \leftsquigarrow Z_i(\vec{x}, \vec{z}) \mid 1 \leqslant i \leqslant 6\}$

- SELECT DISTINCT * WHERE {?x rdf:type A. ?x P ?y. ?x R ?z}
  - **UCQ-unfolding:** $\bigcup_{i,j,k \in \{1,\ldots,6\}} SC(V_i \bowtie W_j \bowtie Z_k)$
    - $6^3 = 216$ CQs
    - **Redundancy?**
- SELECT DISTINCT * WHERE {{?x rdf:type A. ?x P ?y} {?x rdf:type A. ?x R ?z}}

Trade-off Redundancy/Indexing

Mapping $\mathcal{M}$

% High-Level Mappings *target* $\leftsquigarrow$ *source*

$\{URI_1(\vec{x})\ a\ A \leftsquigarrow V_i(\vec{x}) \mid 1 \leqslant i \leqslant 6\}$

$\{URI_1(\vec{x})\ P\ URI_2(\vec{y}) \leftsquigarrow W_i(\vec{x}, \vec{y}) \mid 1 \leqslant i \leqslant 6\}$

$\{URI_1(\vec{x})\ R\ URI_3(\vec{z}) \leftsquigarrow Z_i(\vec{x}, \vec{z}) \mid 1 \leqslant i \leqslant 6\}$

- SELECT DISTINCT * WHERE {?x rdf:type A. ?x P ?y. ?x R ?z}
  - **UCQ-unfolding:** $\bigcup_{i,j,k \in \{1,\ldots,6\}} SC(V_i \bowtie W_j \bowtie Z_k)$
    - $6^3 = 216$ CQs
    - **Redundancy?**
- SELECT DISTINCT * WHERE {{?x rdf:type A. ?x P ?y} {?x rdf:type A. ?x R ?z}}
  - **Join of UCQ-unfoldings:** $(\bigcup_{i,j \in \{1,\ldots,6\}} SC(V_i \bowtie W_j)) \bowtie (\bigcup_{i,k \in \{1,\ldots,6\}} SC(V_i \bowtie Z_k))$

Trade-off Redundancy/Indexing

Mapping $\mathcal{M}$

% High-Level Mappings   target $\leftsquigarrow$ source

$\{URI_1(\vec{x})\ a\ A \leftsquigarrow V_i(\vec{x}) \mid 1 \leqslant i \leqslant 6\}$

$\{URI_1(\vec{x})\ P\ URI_2(\vec{y}) \leftsquigarrow W_i(\vec{x}, \vec{y}) \mid 1 \leqslant i \leqslant 6\}$

$\{URI_1(\vec{x})\ R\ URI_3(\vec{z}) \leftsquigarrow Z_i(\vec{x}, \vec{z}) \mid 1 \leqslant i \leqslant 6\}$

- SELECT DISTINCT * WHERE {?x rdf:type A. ?x P ?y. ?x R ?z}
  - **UCQ-unfolding:** $\bigcup_{i,j,k \in \{1,\ldots,6\}} SC(V_i \bowtie W_j \bowtie Z_k)$
    - $6^3 = 216$ CQs
    - **Redundancy?**
- SELECT DISTINCT * WHERE {{?x rdf:type A. ?x P ?y} {?x rdf:type A. ?x R ?z}}
  - **Join of UCQ-unfoldings:** $(\bigcup_{i,j \in \{1,\ldots,6\}} SC(V_i \bowtie W_j)) \bowtie (\bigcup_{i,k \in \{1,\ldots,6\}} SC(V_i \bowtie Z_k))$
    - $6^2 + 6^2 = 64$ CQs

Trade-off Redundancy/Indexing

Mapping $\mathcal{M}$

% High-Level Mappings *target* $\leftsquigarrow$ *source*

$\{URI_1(\vec{x})\ a\ A \leftsquigarrow V_i(\vec{x}) \mid 1 \leqslant i \leqslant 6\}$

$\{URI_1(\vec{x})\ P\ URI_2(\vec{y}) \leftsquigarrow W_i(\vec{x}, \vec{y}) \mid 1 \leqslant i \leqslant 6\}$

$\{URI_1(\vec{x})\ R\ URI_3(\vec{z}) \leftsquigarrow Z_i(\vec{x}, \vec{z}) \mid 1 \leqslant i \leqslant 6\}$

- SELECT DISTINCT * WHERE {?x rdf:type A. ?x P ?y. ?x R ?z}
  - **UCQ-unfolding:** $\bigcup_{i,j,k \in \{1,\dots,6\}} SC(V_i \bowtie W_j \bowtie Z_k)$
    - $6^3 = 216$ CQs
    - **Redundancy?**
- SELECT DISTINCT * WHERE {{?x rdf:type A. ?x P ?y} {?x rdf:type A. ?x R ?z}}
  - **Join of UCQ-unfoldings:** $(\bigcup_{i,j \in \{1,\dots,6\}} SC(V_i \bowtie W_j)) \bowtie (\bigcup_{i,k \in \{1,\dots,6\}} SC(V_i \bowtie Z_k))$
    - $6^2 + 6^2 = 64$ CQs
    - **Indexes?**

Key Ideas

- The UCQ form for the unfolding is a "one-fits-all" solution
  - However, other forms could be more efficient over certain data instances
- Ideally, one would want to choose the best form for the given OBDA setting[1]
- This is typical in databases, where cost models are used to select the best among different candidate query execution plans
- [Bursztyn et al., 2015] adapted such cost-based solutions to the OBQA[2] context
- We here study how to adapt such techniques to the OBDA scenario

[1] OBDA setting: ontology, mappings, database schema and database instance.
[2] Ontology-based Query Answering. No mappings.

Aspects to Consider

- **Adapting such techniques requires to consider several aspects:**
  - **Alternative unfoldings and structural optimizations**
  - **Additional information available in an OBDA input**
  - **More complex, structured, search space**
  - **Interaction of the mappings and data with the ontology**

Aspects to Consider

- **Adapting such techniques requires to consider several aspects:**
  - **Alternative unfoldings and structural optimizations**
  - **Additional information available in an OBDA input**
  - **More complex, structured, search space**
  - **Interaction of the mappings and data with the ontology**

Collecting Statistics (for Properties)

- Consider mappings $\mathcal{M}$ and a data instance $\mathcal{D}$
- Let $URI_1(\vec{x})$ P $URI_2(\vec{y}) \leftsquigarrow V(\vec{x}, \vec{y})$ be a mapping in $\mathcal{M}$
- We collect:
  - $|V^{\mathcal{D}}|$ **(S1)**
  - $|\pi_{\vec{x}}(V^{\mathcal{D}})|$, $|\pi_{\vec{y}}(V^{\mathcal{D}})|$ **(S2)**
- If $URI_1(\vec{x})$ R $URI_3(\vec{w}) \leftsquigarrow W(\vec{x}, \vec{w})$ belongs to $\mathcal{M}$, then we also collect:
  - $|\pi_{\vec{x}}(V^{\mathcal{D}}) \cap \pi_{\vec{x}}(W^{\mathcal{D}})|$ **(S3)**

Cardinality Estimation

- For a single join:

$$f_{\mathcal{D}}(V \bowtie_{V.\vec{x}=W.\vec{y}} W) = \left\lceil |(V \bowtie_{V.\vec{x}=W.\vec{y}} W)^{\mathcal{D}}| \cdot \frac{|V^{\mathcal{D}}|}{|\pi_{\vec{x}} V^{\mathcal{D}}|} \cdot \frac{|W^{\mathcal{D}}|}{|\pi_{\vec{y}} W|} \right\rceil$$

- **Statistics Used: (S1), (S2), (S3)**
- **Assumptions:**
  - **Uniformity:** $P(C = v_1) = P(C = v_2)$, for all $v_1, v_2 \in C$
  - **Independence:** $P(C_1 = v_1 | C_2 = v_2) = P(C_1 = v_1)$, for all $v_1 \in C_1$ and $v_2 \in C_2$

Cardinality Estimation for Basic CQs

Given a basic CQ E', $f_{\mathcal{D}}(E')$ estimates the number $|E'^{\mathcal{D}}|$ of distinct results in the evaluation of E' over $\mathcal{D}$. We define it as:

$$f_{\mathcal{D}}(E \bowtie_{V_{[p]}.\vec{x}=W_{[q]}.\vec{y}} W_{[q]}) = \begin{cases} \left\lceil \dfrac{k_{\mathcal{D}}(V_{[p]} \bowtie_{V_{[p]}.\vec{x}=W_{[q]}.\vec{y}} W_{[q]}) \cdot |V^{\mathcal{D}}| \cdot |W^{\mathcal{D}}|}{dist_{\mathcal{D}}(V, V_{[p]}.\vec{x}) \cdot dist_{\mathcal{D}}(W, W_{[q]}.\vec{y})} \right\rceil, & \text{if } E = V \\ \left\lceil \dfrac{k_{\mathcal{D}}(E \bowtie_{V_{[p]}.\vec{x}=W_{[q]}.\vec{y}} W_{[q]}) \cdot f_{\mathcal{D}}(E) \cdot |W^{\mathcal{D}}|}{dist_{\mathcal{D}}(E, V_{[p]}.\vec{x}) \cdot dist_{\mathcal{D}}(W, V_{[p]}.\vec{y})} \right\rceil, & \text{otherwise.} \end{cases}$$

Cost Model

Cost for the Unfolding of a UCQ

$$c(q^{ucq}) \quad = \quad \sum_i c(q_i^{cq}) + c_u(q^{ucq})$$

- $c(q_i^{cq})$ *is the cost of evaluating each* q_i^{cq} *in* q^{ucq}
- $c_u(q^{ucq})$ *is the cost of removing duplicate results*

Cost for the Unfolding of a Join of UCQs (JUCQ)

$$c(q^{jucq}) \quad = \quad \sum_i c(q_i^{ucq}) + \sum_{i \neq k} c_{mat}(q_i^{ucq}) + c_{mj}(q^{jucq}) + c_u(q^{jucq})$$

- $c(q_i^{ucq})$ *is the cost of evaluating each UCQ component* q_i^{ucq}
- $\sum_{i \neq k} c_{mat}(q_i^{ucq})$ *is the cost of materializing the intermediate results from* q_i^{ucq}
- $c_{mj}(q^{jucq})$ *is the cost of a merge join over the materialized intermediate results*
- $c_u(q^{jucq})$ *is the cost of removing duplicate results*

Empirical Evaluation

- **Test suites:** System testing (OBDA extension of the Wisconsin Benchmark) and application-driven (NPD Benchmark)
- **Hardware:** HP Proliant 24 cores@3.47GHz; 106 GB of RAM
- **RDBMS:** PostgreSQL 9.6

Wisconsin Experiment

- **Comparison between:**
  - SELECT DISTINCT * WHERE {?x :*mr*P1 ?y. ?x :*jmr*P2 ?z. ?x :*jmr*P3 ?w} (q_{ucq})
  - SELECT DISTINCT ?x ?y ?z ?w WHERE
    {{?x :*mr*P1 ?y. ?x :*jmr*P2 ?z} {?x :*mr*P1 ?y1. ?x :*jmr*P3 ?w}} (q_{jucq})
- **Parameters:**
  - *m*: **number of mappings**, in $\{1, \ldots, 6\}$
  - *r*: **number of redundant mappings**[3], in $\{1, \ldots, m-1\}$
  - *j*: **join-selectivity**[4], in $\{5, 10, 15, 20\}$
- **All intra-BGP joins are over indexed values**

[3] Mappings which do not contribute to new assertions

[4] **%** of the number of retrieved rows for each mapping defining the property (each mapping retrieves **200**k rows)

Analysis of the Trade-Off Redundancy/Indexing

- **Blue color**: q^{ucq} **is better**
- **Red color**: q^{jucq} **is better**
- **Numbers in cells: 1** $- (\textit{time}(q^{jucq}) / \textit{time}(q^{ucq}))$

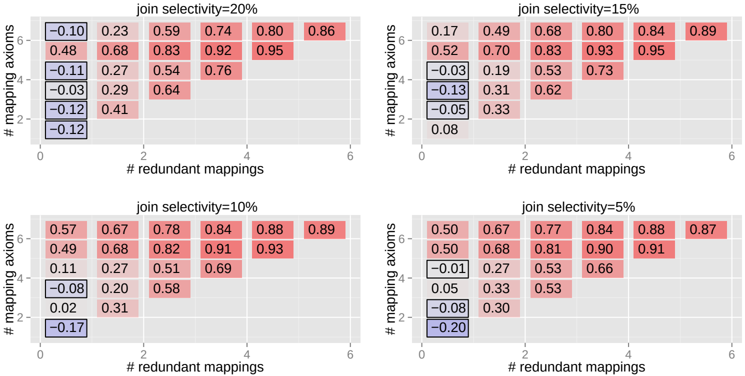


Figure: Performance gain of JUCQ compared with UCQ

Cost

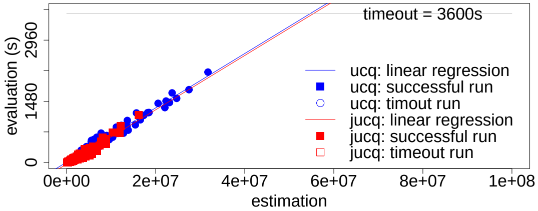


Figure: Our cost model vs. evaluation

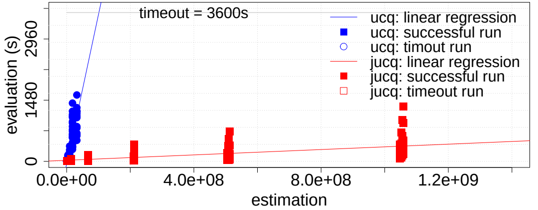


Figure: PostgreSQL cost model vs. evaluation

Evaluation of PostgreSQL Cardinality Estimation (UCQs)

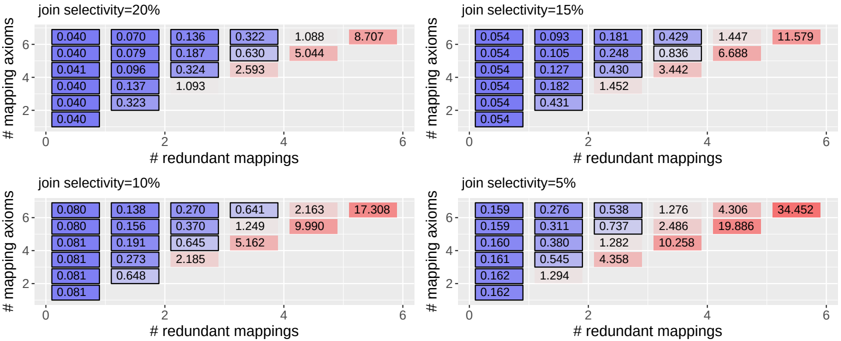


Figure: UCQs: (PostgreSQL estimated cardinality) / (real cardinality)

Evaluation of PostgreSQL Cardinality Estimation (JUCQs)

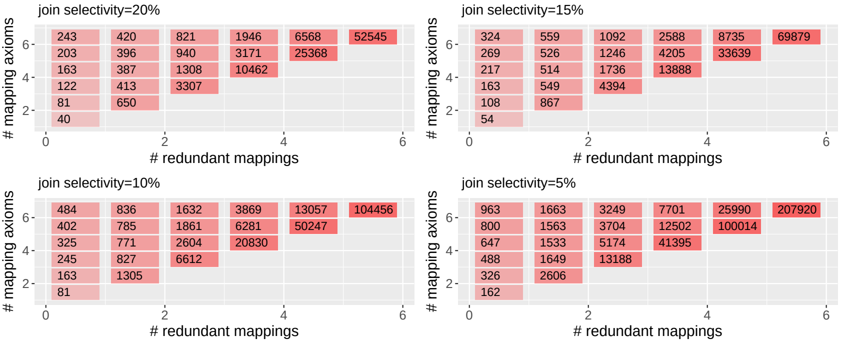


Figure: JUCQs: (PostgreSQL estimated cardinality) / (real cardinality)

NPD Benchmark Experiment

- **The NPD benchmark is an OBDA-specific benchmark aimed at reproducing the use of an OBDA system in an industrial real world scenario**

Table: Evaluation over the NPD benchmark

SPARQL Query		Unfolding for UCQs		Unfolding for JUCQs		
name	# triple patterns	time (s)	# CQs	time (s)	# Frags	# CQs
q_6	7	2.18	48	1.20	2	14
q_{11}	8	3.39	24	0.40	2	12
q_{12}	10	6.67	48	0.47	2	14
q_{31}	10	54.27	3840	1.58	2	327

Conclusion and Future Work

- **Contribution:**
  - We studied how to find the most efficient unfolding of a user query
    - We introduced a space of unfolding alternatives
    - We defined structural optimizations for the new alternatives
    - We devised a cost model to search for the best alternative
  - We performed two empirical evaluations, and observed that:
    - Alternative unfoldings can be orders of magnitude more efficient than traditional UCQ unfoldings
    - The devised cost model is well suited to select the best unfolding
- **Future work:**
  - Integrate cost estimation with semantic optimizations
  - Explore more unfolding alternatives
  - Relax estimation assumptions by integrating our model with existing techniques based on histograms

References I

[Bursztyn et al., 2015] Bursztyn, D., Goasdoué, F., and Manolescu, I. (2015).
Reformulation-based query answering in RDF: alternatives and performance.
Proc. of VLDB, 8(12):1888–1891.

[Chebotko et al., 2009] Chebotko, A., Lu, S., and Fotouhi, F. (2009).
Semantics preserving sparql-to-sql translation.
Data & Knowledge Engineering, 68(10):973 – 1000.