Practical Aspects of Query Rewriting for OWL 2

Héctor Pérez-Urbina, Ian Horrocks, and Boris Motik

Oxford University
Computing Laboratory

OWL: Experiences and Directions Workshop
October 2009

Data Access with OWL 2 QL

Data Access with OWL 2 QL

$Q(x) \leftarrow \text{teaches}(x, y)$

$$\begin{array}{rcl} \text{Teacher} & \sqsubseteq & \exists \text{teaches} \\ \text{Professor} & \sqsubseteq & \text{Teacher} \\ \exists \text{hasTutor}^- & \sqsubseteq & \text{Professor} \end{array}$$

PROFESSOR(*name, office, phone*)
STUDENT(*name, major, tutor*)

Data Access with OWL 2 QL

$Q(x) \leftarrow \text{teaches}(x, y)$

$$\begin{array}{rcl} \text{Teacher} & \sqsubseteq & \exists\text{teaches} \\ \text{Professor} & \sqsubseteq & \text{Teacher} \\ \exists\text{hasTutor}^- & \sqsubseteq & \text{Professor} \end{array}$$

PROFESSOR(*name, office, phone*)
STUDENT(*name, major, tutor*)

Professor $\mapsto$
```
SELECT name
FROM PROFESSOR
```

hasTutor $\mapsto$
```
SELECT name,tutor
FROM STUDENT
```

Data Access with OWL 2 QL

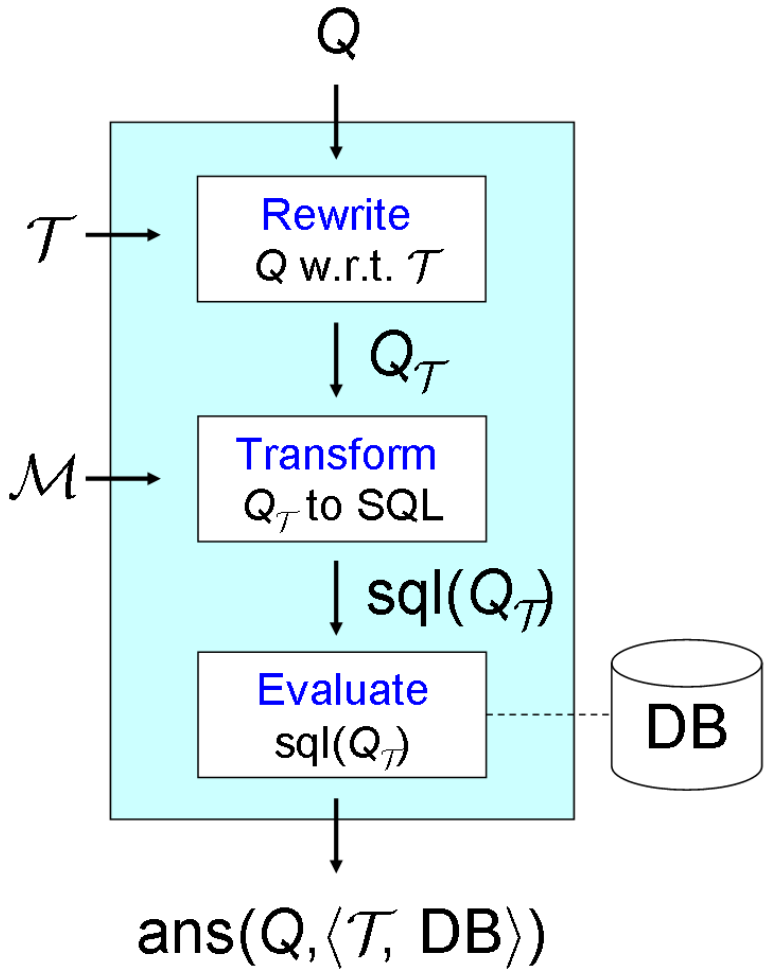
$Q(x) \leftarrow \text{teaches}(x, y)$

$$\begin{array}{rcl} \text{Teacher} & \sqsubseteq & \exists \text{teaches} \\ \text{Professor} & \sqsubseteq & \text{Teacher} \\ \exists \text{hasTutor}^- & \sqsubseteq & \text{Professor} \end{array}$$

PROFESSOR(*name, office, phone*)
STUDENT(*name, major, tutor*)

Professor $\mapsto$
```
SELECT name
FROM PROFESSOR
```

hasTutor $\mapsto$
```
SELECT name,tutor
FROM STUDENT
```

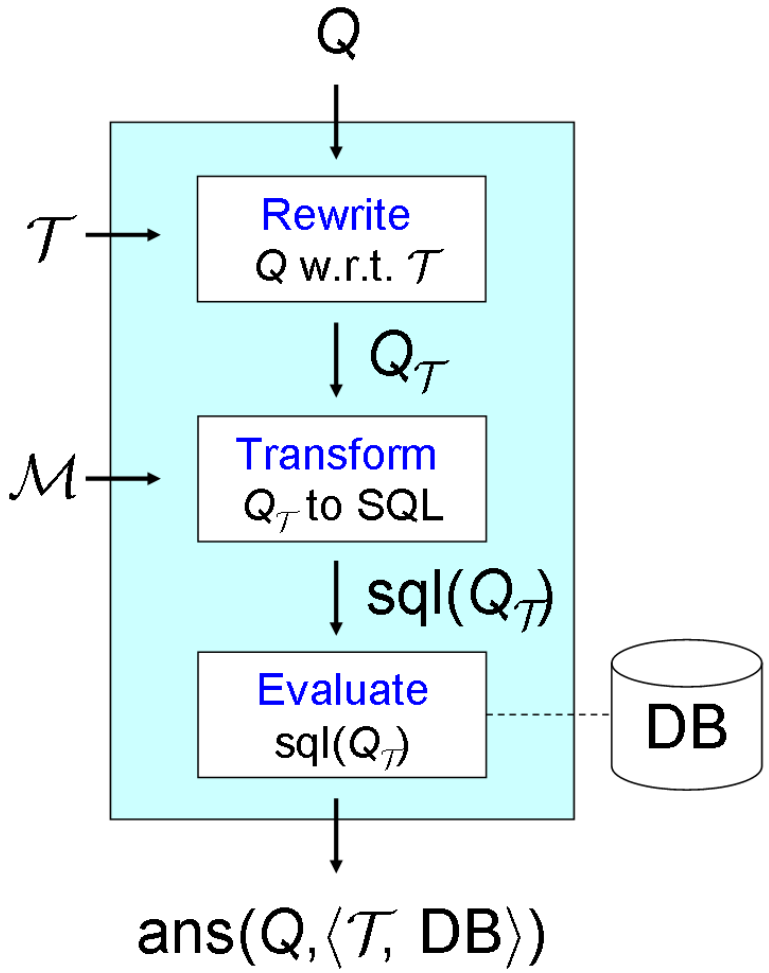


Data Access with OWL 2 QL

$Q(x) \leftarrow \text{teaches}(x, y)$

$$\begin{array}{rcl} \text{Teacher} & \sqsubseteq & \exists\text{teaches} \\ \text{Professor} & \sqsubseteq & \text{Teacher} \\ \exists\text{hasTutor}^- & \sqsubseteq & \text{Professor} \end{array}$$

CGLLR rewriting algorithm by Calvanese et al.

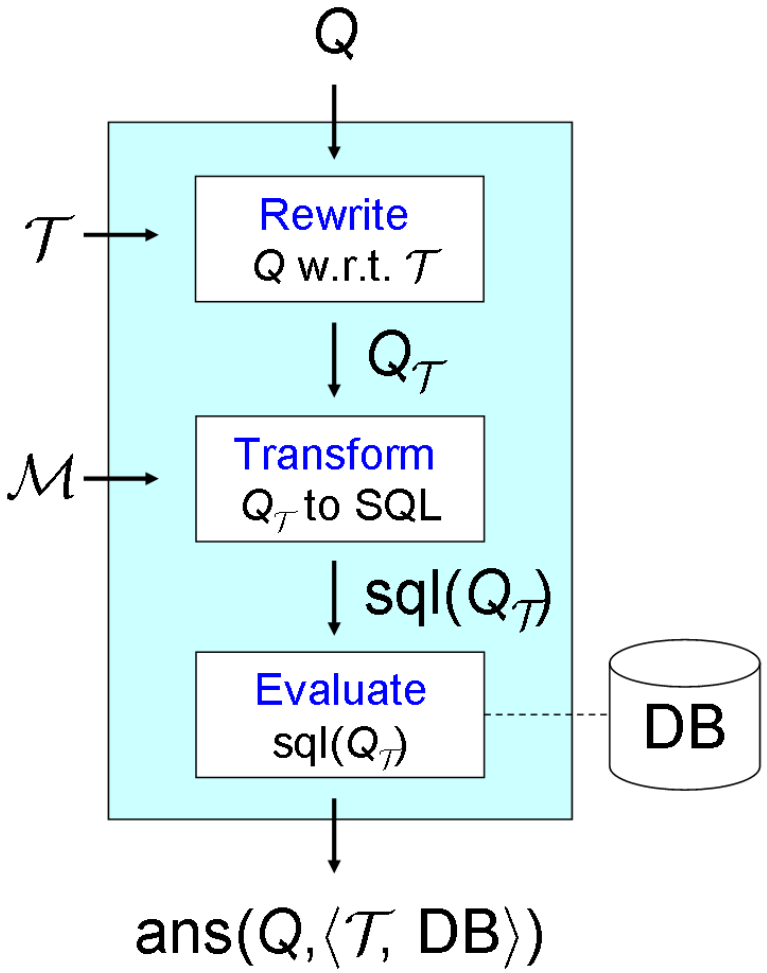


Data Access with OWL 2 QL

$Q(x) \leftarrow \text{teaches}(x, y)$

$$\begin{array}{rcl} \text{Teacher} & \sqsubseteq & \exists \text{teaches} \\ \text{Professor} & \sqsubseteq & \text{Teacher} \\ \exists \text{hasTutor}^- & \sqsubseteq & \text{Professor} \end{array}$$

$$\begin{aligned} Q_{\mathcal{T}} = \{ \quad & Q(x) \leftarrow \text{teaches}(x, y), \\ & Q(x) \leftarrow \text{Teacher}(x), \\ & Q(x) \leftarrow \text{Professor}(x), \\ & Q(x) \leftarrow \text{hasTutor}(y, x) \quad \} \end{aligned}$$



Data Access with OWL 2 QL

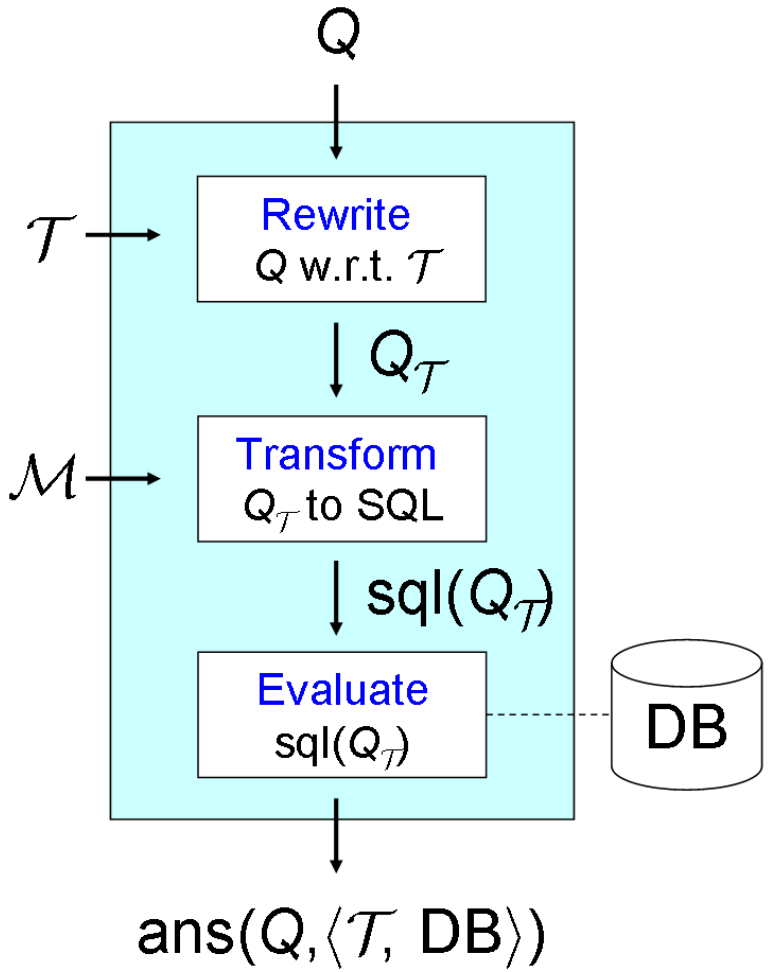
$Q(x) \leftarrow \text{teaches}(x, y)$

$$\begin{array}{rcl} \text{Teacher} & \sqsubseteq & \exists \text{teaches} \\ \text{Professor} & \sqsubseteq & \text{Teacher} \\ \exists \text{hasTutor}^- & \sqsubseteq & \text{Professor} \end{array}$$

$$\begin{aligned} Q_{\mathcal{T}} = \{ \quad & Q(x) \leftarrow \text{teaches}(x, y), \\ & Q(x) \leftarrow \text{Teacher}(x), \\ & Q(x) \leftarrow \text{Professor}(x), \\ & Q(x) \leftarrow \text{hasTutor}(y, x) \quad \} \end{aligned}$$

$\text{sql}(Q_{\mathcal{T}}) =$

```
SELECT name
FROM PROFESSOR
UNION
SELECT Tutor
FROM Student
```



SO, ARE WE DONE HERE?

So, are we done here?

- $Q_{\mathcal{T}}$ is worst-case exponential w.r.t. Q and $\mathcal{T}$
  - Costly to compute
  - Costly (or impossible) to evaluate

So, are we done here?

- $Q_{\mathcal{T}}$ is worst-case exponential w.r.t. Q and $\mathcal{T}$
  - Costly to compute
  - Costly (or impossible) to evaluate
- Applications may require constructs that go beyond QL
  - Student $\sqcap \exists$hasSupervisor $\sqsubseteq$ GraduateStudent
  - $\exists$studies.Course $\sqsubseteq$ Student
  - OxfordStudent $\sqsubseteq \exists$studiesAt.$\{$OxfordUniversity$\}$

So, are we done here?

- $Q_{\mathcal{T}}$ is worst-case exponential w.r.t. Q and $\mathcal{T}$
  - Costly to compute
  - Costly (or impossible) to evaluate
- Applications may require constructs that go beyond QL
  - Student $\sqcap \exists$hasSupervisor $\sqsubseteq$ GraduateStudent
  - $\exists$studies.Course $\sqsubseteq$ Student
  - OxfordStudent $\sqsubseteq \exists$studiesAt.$\{$OxfordUniversity$\}$

RQR (Resolution-based Query Rewriting)

- Handles $\mathcal{ELHIO}^{\neg}$ (most of OWL 2 EL)

So, are we done here?

- $Q_{\mathcal{T}}$ is worst-case exponential w.r.t. Q and $\mathcal{T}$
  - Costly to compute
  - Costly (or impossible) to evaluate
- Applications may require constructs that go beyond QL
  - Student $\sqcap \exists$hasSupervisor $\sqsubseteq$ GraduateStudent
  - $\exists$studies.Course $\sqsubseteq$ Student
  - OxfordStudent $\sqsubseteq \exists$studiesAt.{OxfordUniversity}

RQR (Resolution-based Query Rewriting)

- Handles $\mathcal{ELHIO}^{\neg}$ (most of OWL 2 EL)
- $Q_{\mathcal{T}}$ might be a datalog query
- "Pay-as-you-go" behavior: extends and generalizes CGLLR

EVALUATION

Evaluation

Number of inferences ≈ Time

	REQUIEM (RQR)	C (CGLLR)
Overall	70,846	343,813
Average	2,200	12,066

- REQUIEM: 73% smaller, 0% larger, and 27% equal

Evaluation

Number of inferences ≈ Time

	REQUIEM (RQR)	C (CGLLR)
Overall	70,846	343,813
Average	2,200	12,066

- REQUIEM: 73% smaller, 0% larger, and 27% equal

Number of queries

	REQUIEM (RQR)	C (CGLLR)
Overall	10,931	75,301
Average	289	2,682

- REQUIEM: 83% smaller, 0% larger, and 17% equal

Optimizing the Rewritings

Query Subsumption

$$Q_1 = Q(x) \leftarrow \text{teaches}(x, y)$$
$$Q_2 = Q(x) \leftarrow \text{teaches}(x, y) \wedge \text{Student}(y)$$

Optimizing the Rewritings

Query Subsumption

$Q_1 = Q(x) \leftarrow \text{teaches}(x, y)$
$Q_2 = Q(x) \leftarrow \text{teaches}(x, y) \wedge \text{Student}(y)$

Q_1 subsumes Q_2

Optimizing the Rewritings

Query Subsumption

$$Q_1 = Q(x) \leftarrow \text{teaches}(x, y)$$
$$Q_2 = Q(x) \leftarrow \text{teaches}(x, y) \wedge \text{Student}(y)$$

Q_1 subsumes Q_2

- Discard subsumed queries a posteriori
- Significant reduction in the size of the rewritings

Optimizing the Rewritings

Query Subsumption

$Q_1 = Q(x) \leftarrow \text{teaches}(x, y)$
$Q_2 = Q(x) \leftarrow \text{teaches}(x, y) \wedge \text{Student}(y)$

Q_1 subsumes Q_2

- Discard subsumed queries a posteriori
- Significant reduction in the size of the rewritings
- On the fly: forward/backward subsumption
- Straightforwardly applicable to RQR

Going Beyond QL

Considerations

- $Q_{\mathcal{T}}$ is not guaranteed to be a UCQ: use of deductive database systems

Going Beyond QL

Considerations

- $Q_{\mathcal{T}}$ is not guaranteed to be a UCQ: use of deductive database systems
- Additional optimizations
  - Empty EDB predicates pruning
  - Dependency graph pruning

Going Beyond QL

Considerations

- $Q_{\mathcal{T}}$ is not guaranteed to be a UCQ: use of deductive database systems
- Additional optimizations
  - Empty EDB predicates pruning
  - Dependency graph pruning
- Greedy unfolding

Going Beyond QL

Considerations

- $Q_{\mathcal{T}}$ is not guaranteed to be a UCQ: use of deductive database systems
- Additional optimizations
  - Empty EDB predicates pruning
  - Dependency graph pruning
- Greedy unfolding

Evaluation

- Good performance w.r.t. time and size of the rewritings
- Greedy unfolding produces UCQs in many cases

Conclusions and Future Work

Conclusions

- RQR: significantly smaller rewritings in significantly fewer steps than existing algorithms
- Amenable to various straightforward optimizations
- Use of databases for realistic OWL 2 EL ontologies
- Open source implementation: REQUIEM[1]

[1] http://www.comlab.ox.ac.uk/projects/requiem/

Conclusions and Future Work

Conclusions

- RQR: significantly smaller rewritings in significantly fewer steps than existing algorithms
- Amenable to various straightforward optimizations
- Use of databases for realistic OWL 2 EL ontologies
- Open source implementation: REQUIEM[1]

Future Work

- Evaluate REQUIEM with databases

[1] http://www.comlab.ox.ac.uk/projects/requiem/