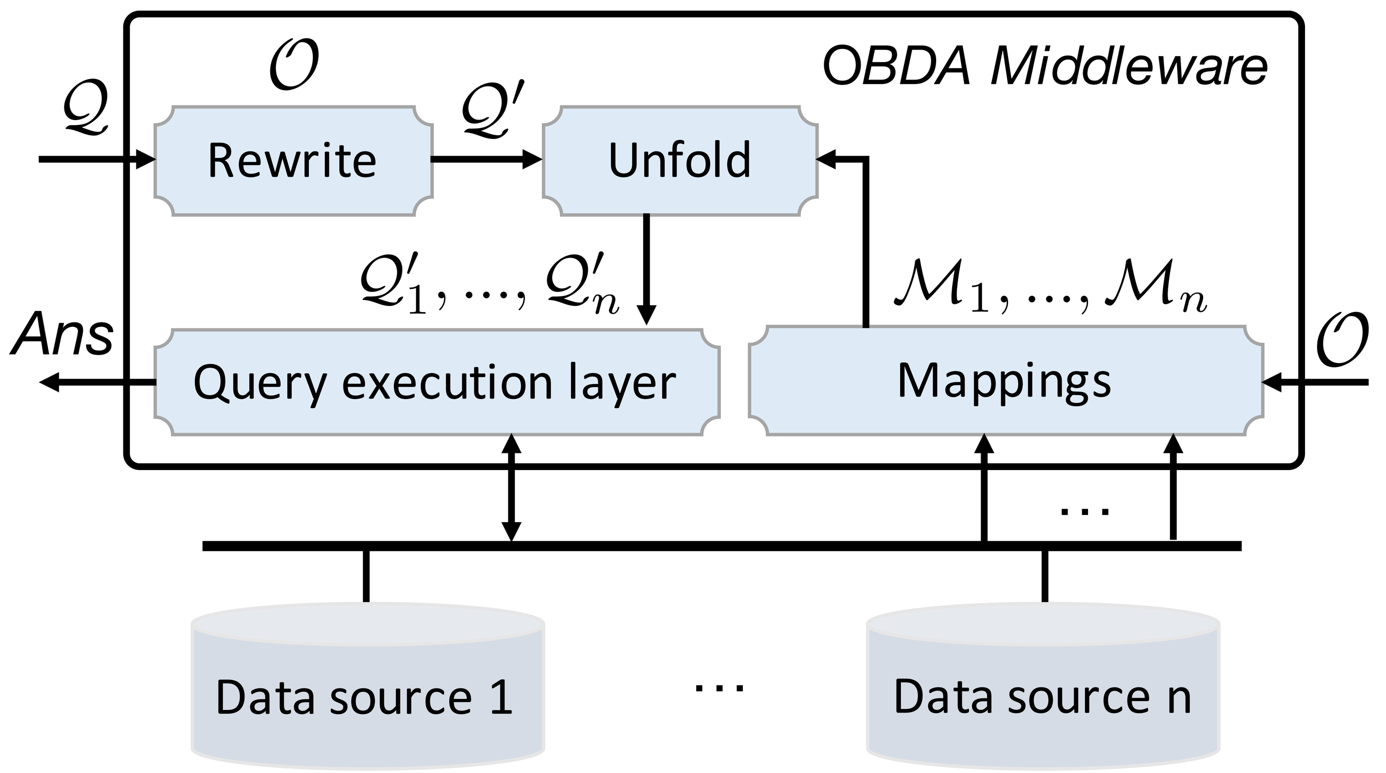
University of Oxford, Computer Science Department

Ontology-mediated Analytic Query Answering

Bernardo Cuenca Grau Ian Horrocks Mark Kaminski
Egor Kostylev Boris Motik Charalampos Nikolaou

Analytics-aware ontology-based data access

OBDA – Classical setting



Limitations and challenges

- *Rewritability* of queries is not always possible
- Limited query language (e.g., no support for analytical operations like $\min, \max, \text{count}, \text{sum}, \text{avg}$)
- *Semantic mismatch* between data sources and ontological level

Data sources		Ontological level
closed world (CWA)		open world (OWA)
bag semantics	**vs.**	set semantics
null values		existential quantification

$\Rightarrow$ Problematic with respect to supporting analytical operations

- *Optimisation* of ontological queries is *hard*

Approach

Motivating example

$$\mathcal{O}: \quad \begin{array}{l} Sensor \sqsubseteq \exists hasMeasmnt \\ Sensor \sqsubseteq \exists hasMUnit \end{array}$$

$\mathcal{M}$:
- SELECT sid FROM Sensors $\rightsquigarrow Sensor(sid)$
- SELECT sid, unit FROM Sensors $\rightsquigarrow hasMUnit(sid, unit)$
- SELECT sid, val FROM Measmnts $\rightsquigarrow hasMeasmnt(sid, val)$

Data source:

Sensors (sid,tid,unit)		
sensor1	turbine1	celsius
sensor2	turbine1	psi
sensor3	turbine2	rpm
...		

Measmnts (sid,val,t)		
sensor1	-50	t1
sensor2	5000	t1
sensor3	2500	t1
sensor1	-50	t2
...		

What is the answer to the following questions?

Q_1: *What is the number of measurements?*
Q_2: *What is the maximum measurement value for each sensor?*

Query evaluation

- ACQ query language: CQ + aggregate functions in head [1, 2]
  $\mathcal{Q}_1: \ q(\text{count}(y)) \leftarrow \ \exists x \ hasMeasmnt(x, y)$
  $\mathcal{Q}_2: q(x, \max(y)) \leftarrow \ hasMeasmnt(x, y)$
- **Rewritability:** Are there $\mathcal{Q}'_1$, $\mathcal{Q}'_2$ over the data sources to answer $\mathcal{Q}_1$ and $\mathcal{Q}_2$ over $\mathcal{O}$?

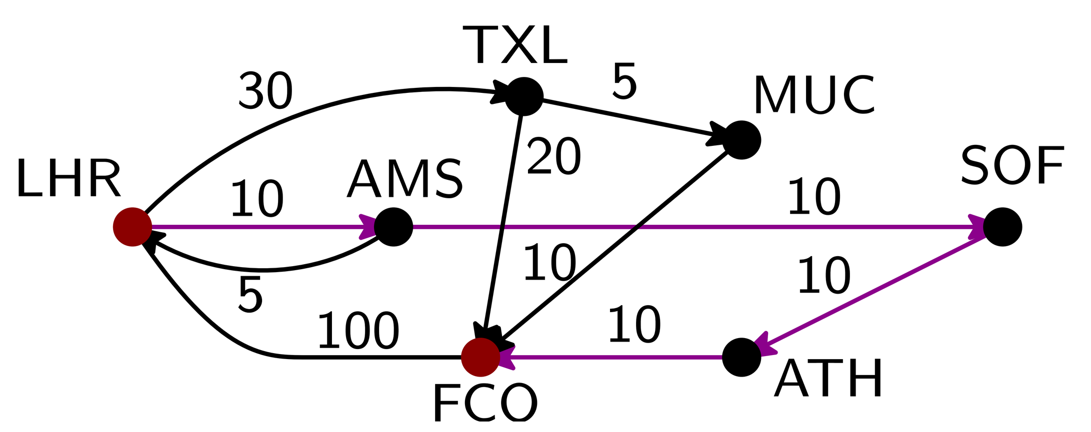
Semantics

- Extend DL-Lite$_{\mathcal{A}}$ with *multi-set* (bag) semantics
  $\Rightarrow$ conservative extension of set semantics
  $\Rightarrow$ preserves cardinality from data sources
- count-ACQs: (a) **minimal** [2] and (b) **skolem-based** semantics
  $\Rightarrow$ identify *safe queries* for which minimality is tractable
- non-count-ACQs: aggregate over known values [1]

Expressiveness: pushing the envelope further

Can we answer the following questions using ACQs?

- *What is the lowest cost for flying from London to Rome?*
- *How many bolts does a car manufacturer need to order for a specific model?*



Challenges

- Apart from *aggregation* (e.g., sum, $\min$, and count), we need also *recursion* to formulate/answer the above questions
- Assume a *rule-based language* extended with *aggregates*
- Interaction between recursion and aggregation is very powerful and non-trivial to manage

Previous work

- No semantics for the general case
- Proposals in the literature unsatisfactory:
  * High complexity, no value invention [3]
  * Undecidability of fact entailment [4, 5, 6]
  * Limited expressivity (e.g., functionality) [3, 6, 7]
  * Unnatural syntactic restrictions (hard to write programs) [4]

Our goals

- Define intuitive semantics leading to a *unique model*
- Offer a natural and *user-friendly syntax*
- *Generalise* existing approaches
- *Low complexity* of query evaluation; *sufficient* expressive power

Example program for cheapest flights

$\texttt{flight}(\texttt{X}, \texttt{Z}, \texttt{C}) \leftarrow \texttt{flight}(\texttt{X}, \texttt{Y}, \texttt{C1}), \ \texttt{flight}(\texttt{Y}, \texttt{Z}, \texttt{C2}), \ \texttt{C} = \texttt{C1} + \texttt{C2}$
$\texttt{cheapest\_flight}(\texttt{X}, \texttt{Y}, \texttt{C}) \leftarrow \texttt{C} = \min(\texttt{C}, \ \texttt{flight}(\texttt{X}, \texttt{Y}, \texttt{C}))$

References

[1] D. Calvanese et al. Aggregate queries over ontologies. In ONISW '08.
[2] E. V. Kostylev and J. L. Reutter. Complexity of answering counting aggregate queries over DL-Lite. JWS '15.
[3] W. Faber et al. Semantics and complexity of recursive aggregates in answer set programming. AI '11.
[4] K. A. Ross and Y. Sagiv. Monotonic aggregation in deductive database. JCSS '97.
[5] A. Van Gelder. Foundations of aggregation in deductive databases. In DOOD'93.
[6] S. Greco. Dynamic programming in datalog with aggregates. TKDE '99.
[7] I. S. Mumick et al. The magic of duplicates and aggregates. In VLDB '90.