DEPARTMENT OF COMPUTER SCIENCE

UNIVERSITY OF OXFORD

Beyond DL-Lite: Big Data meets Heavy Ontologies

Ian Horrocks
Information Systems Group
Department of Computer Science
University of Oxford

Applications: HCLS

- **SNOMED-CT** (Clinical Terms) ontology
  - provides common vocabulary for recording clinical data
  - used in healthcare systems of more than 15 countries, including Australia, Canada, Denmark, Spain, Sweden and the UK
  - “classified and checked for equivalencies” using ontology reasoners
- **OBO foundry** includes more than 100 biological and biomedical ontologies
  - “continuous integration server running Elk and/or HermiT 24/7 checking that multiple independently developed ontologies are mutually consistent”

Focus is mainly on **schema reasoning**

OBDA: Motivational Example

- **Statoil** use data to inform production and exploration management
  - Large and complex data sets are difficult and time consuming to use

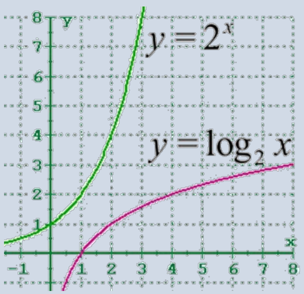
- Ontology Based Data Access (OBDA) can **improve access** to relevant data
  - Intuitive queries over ontology
  - Answers reflect data and knowledge

Focus is mainly on **query answering**

OBDA: Theory $\rightsquigarrow$ Practice

OBDA: Theory $\rightsquigarrow$ Practice

- Most ontologies use **OWL** ontology language
- OWL based on **description logic SROIQ**
  - ✔ Clear semantics
  - ✔ Well understood computational properties (e.g., decidability, complexity)
  - ✔ Simple goal directed reasoning algorithms
  - ✗ N2ExpTime-comlete combined complexity
  - ✗ NP-hard data complexity (-v- logspace for databases)



How can we provide (robustly) scalable query answering?

OWL Profiles

OWL 2 defines language subsets, aka **profiles** that can be "more simply and/or efficiently implemented"

- **OWL 2 QL**
  - Based on DL-Lite
  - AC^0 data complexity (same as DBs)
- **OWL 2 EL**
  - Based on **EL^{++}**
  - PTime-complete for combined and data complexity
- **OWL 2 RL**
  - Based on "**Description Logic Programs**" ($\approx$ DL $\cap$ LP)
  - PTime-complete for combined and data complexity

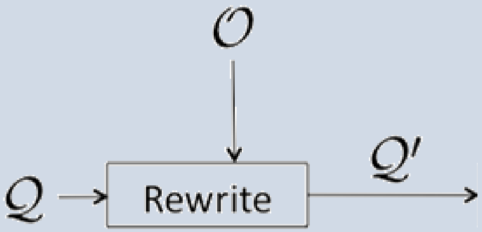
OWL 2 QL and Query Rewriting

Given QL ontology **O** query **Q and mappings M:**

OWL 2 QL and Query Rewriting

Given QL ontology **O** query **Q and mappings M:**

- **Rewrite** **Q** $\rightarrow$ **Q^0** s.t. answering **Q^0** without **O** equivalent to answering **Q** w.r.t. **O** *for any dataset*



OWL 2 QL and Query Rewriting

Given QL ontology **O** query **Q and mappings M:**

- **Rewrite Q** $\rightarrow$ **Q**$^{\mathbf{0}}$ s.t. answering **Q**$^{\mathbf{0}}$ without **O** equivalent to answering **Q** w.r.t. **O** *for any dataset*

$$\mathsf{OilPipeline} \sqsubseteq \mathsf{Pipeline} \sqcap \exists \mathsf{fromFacility}.\mathsf{OilFacility}$$

$$\begin{array}{c} \mathcal{O} \\ \downarrow \\ \mathcal{Q} \longrightarrow \boxed{\mathsf{Rewrite}} \xrightarrow{\mathcal{Q}'} \end{array}$$

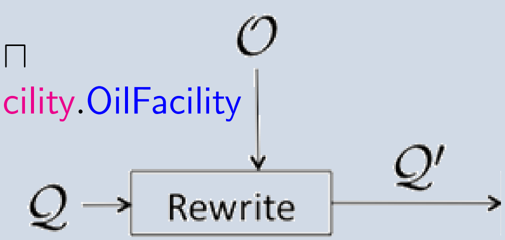
$$Q(x) \leftarrow \mathsf{Pipeline}(x) \wedge \mathsf{fromFacility}(x, y) \wedge \mathsf{OilFacility}(y)$$

OWL 2 QL and Query Rewriting

Given QL ontology **O** query **Q and mappings M:**

- **Rewrite Q** $\rightarrow$ **Q^0** s.t. answering **Q^0** without **O** equivalent to answering **Q** w.r.t. **O** *for any dataset*

$$\mathsf{OilPipeline} \sqsubseteq \mathsf{Pipeline} \sqcap \exists \mathsf{fromFacility}.\mathsf{OilFacility}$$



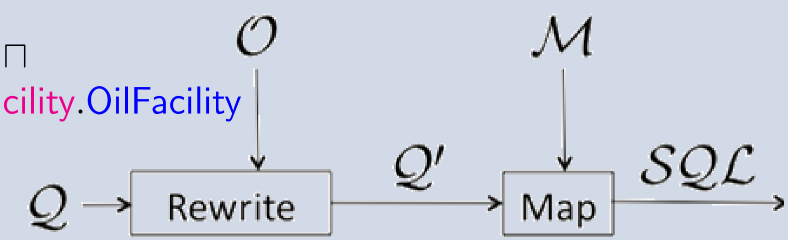
$$Q(x) \leftarrow \mathsf{Pipeline}(x) \wedge \mathsf{fromFacility}(x, y) \wedge \mathsf{OilFacility}(y) \vee \mathsf{OilPipeline}(x)$$

OWL 2 QL and Query Rewriting

Given QL ontology **O** query **Q and mappings M:**

- **Rewrite Q** → **Q⁰** s.t. answering **Q⁰** without **O** equivalent to answering **Q** w.r.t. **O** *for any dataset*
- **Map** ontology queries → DB queries (typically SQL) using mappings **M** to rewrite **Q**' into a DB query

$$\mathsf{OilPipeline} \sqsubseteq \mathsf{Pipeline} \sqcap \exists \mathsf{fromFacility}.\mathsf{OilFacility}$$



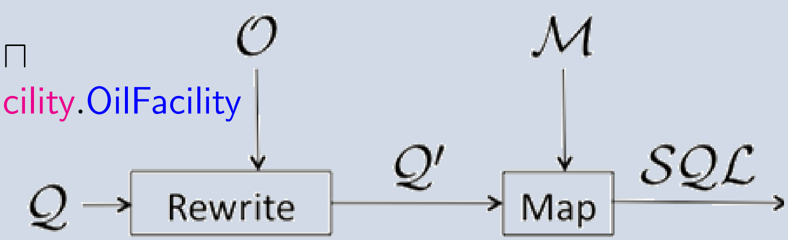
$$Q(x) \leftarrow \mathsf{Pipeline}(x) \wedge \mathsf{fromFacility}(x,y) \wedge \mathsf{OilFacility}(y) \vee \mathsf{OilPipeline}(x)$$

OWL 2 QL and Query Rewriting

Given QL ontology **O** query **Q and mappings M:**

- **Rewrite Q** $\rightarrow$ **Q**^{0} s.t. answering **Q**^{0} without **O** equivalent to answering **Q** w.r.t. **O** *for any dataset*
- **Map** ontology queries $\rightarrow$ DB queries (typically SQL) using mappings **M** to rewrite **Q**’ into a DB query

$\mathsf{OilPipeline} \sqsubseteq \mathsf{Pipeline} \sqcap \exists \mathsf{fromFacility}.\mathsf{OilFacility}$



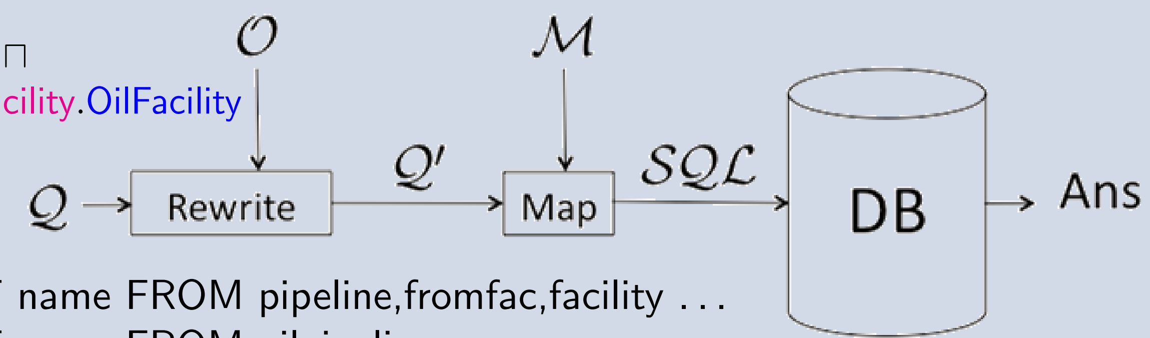
SELECT name FROM pipeline,fromfac,facility . . .
UNION SELECT name FROM oilpipeline

OWL 2 QL and Query Rewriting

Given QL ontology **O** query **Q and mappings M:**

- **Rewrite Q** → **Q^0** s.t. answering **Q^0** without **O** equivalent to answering **Q** w.r.t. **O** *for any dataset*
- **Map** ontology queries → DB queries (typically SQL) using mappings **M** to rewrite **Q**’ into a DB query
- **Evaluate** (SQL) query against DB

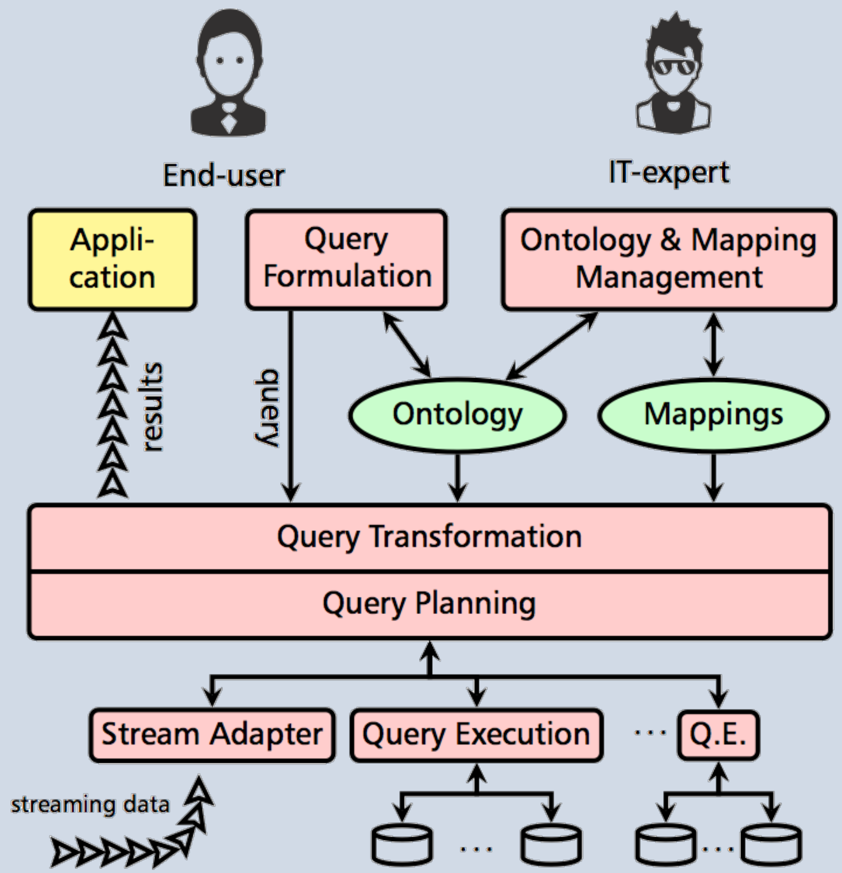
$\mathsf{OilPipeline} \sqsubseteq \mathsf{Pipeline} \sqcap \exists \mathsf{fromFacility}.\mathsf{OilFacility}$



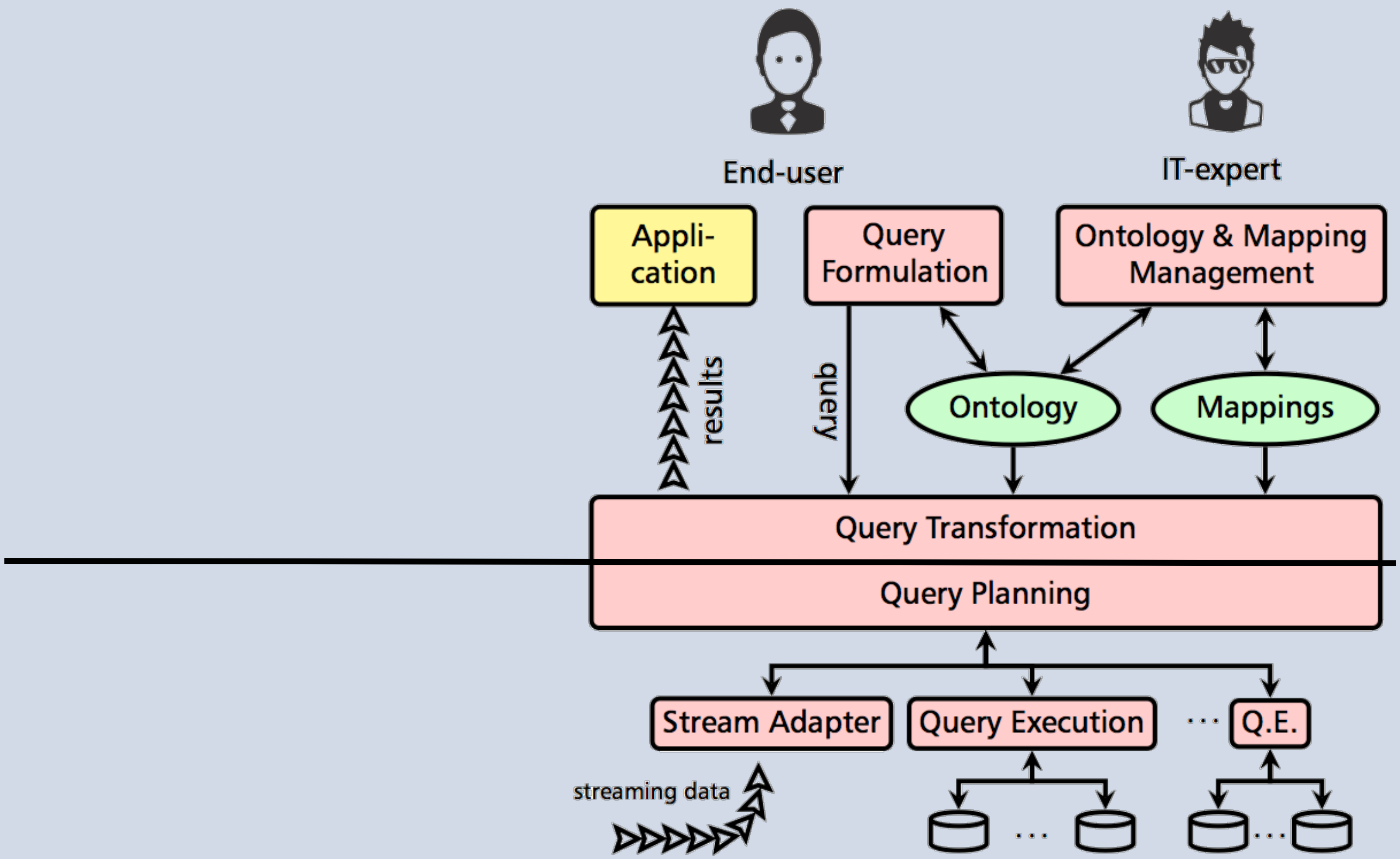
SELECT name FROM pipeline,fromfac,facility . . .
UNION SELECT name FROM oilpipeline

UNIVERSITY OF OXFORD DEPARTMENT OF COMPUTER SCIENCE
Information Systems Group
EPSRC Engineering and Physical Sciences Research Council
Optique
SEVENTH FRAMEWORK PROGRAMME

Optique Platform Architecture



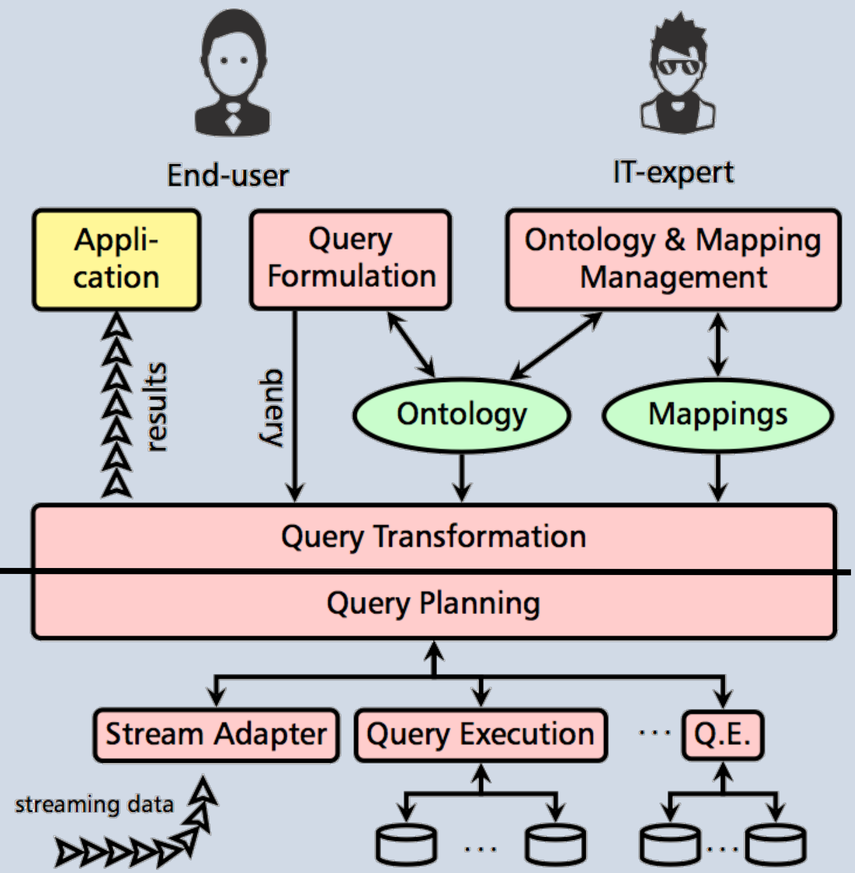
Optique Platform Architecture



Optique Platform Architecture

Query rewriting:

- uses ontology & mappings
- computationally hard
- ontology & mappings small



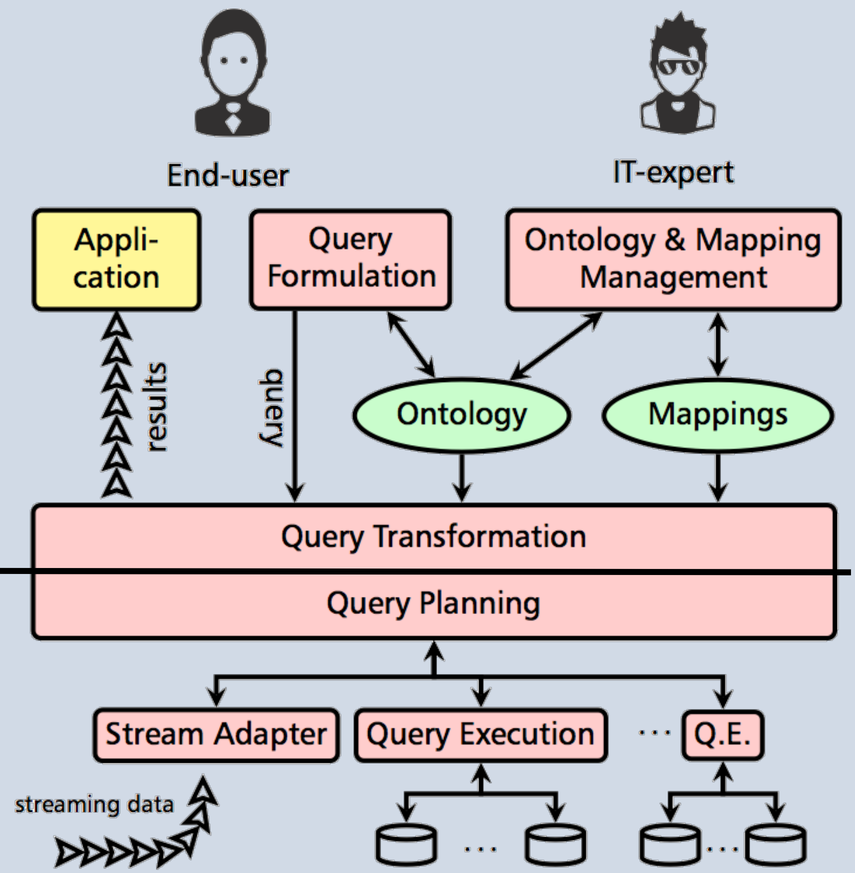
Optique Platform Architecture

Query rewriting:

- uses ontology & mappings
- computationally hard
- ontology & mappings small

Query evaluation:

- ind. of ontology & mappings
- computationally tractable
- data sets very large



Query Rewriting — Issues

1 **Expressivity**

- QL (necessarily) has (very) restricted expressive power

2 **Rewriting**

- May be large (worst case exponential in size of ontology)
- Queries may be hard for existing DBMSs

3 **Mappings**

- May be difficult to develop and maintain
- Relatively little work in this area to date

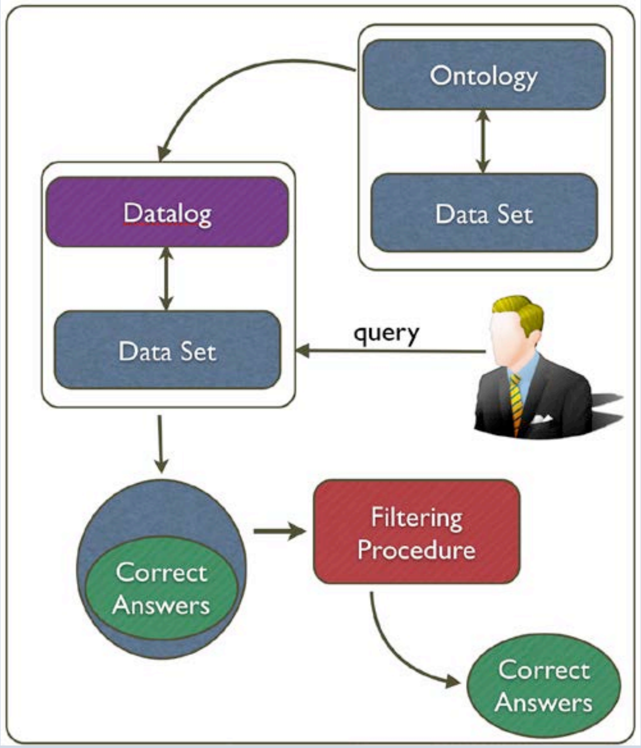
OWL 2 EL and Combined Approach

Given (RDF) data DB, EL ontology O and query Q:

OWL 2 EL and Combined Approach

Given (RDF) data DB, EL ontology O and query Q:

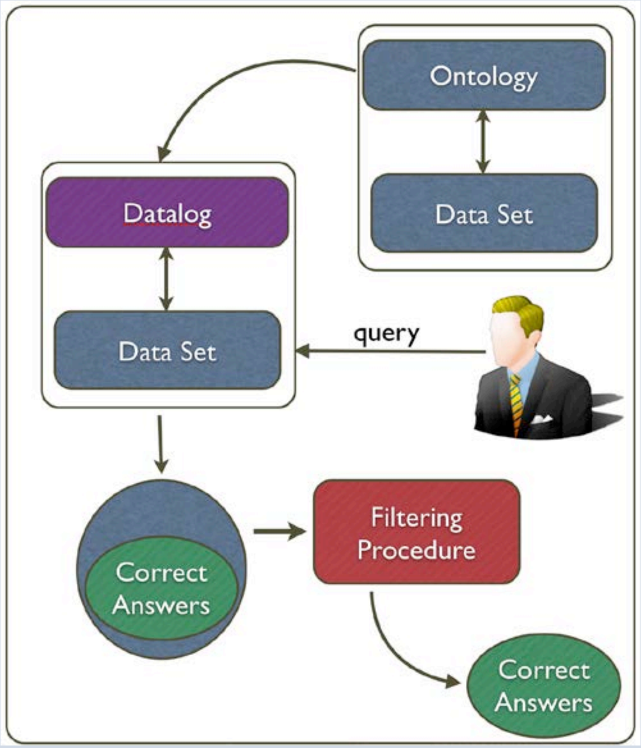
- **Over-approximate** O into Datalog program D



OWL 2 EL and Combined Approach

Given (RDF) data DB, EL ontology O and query Q:

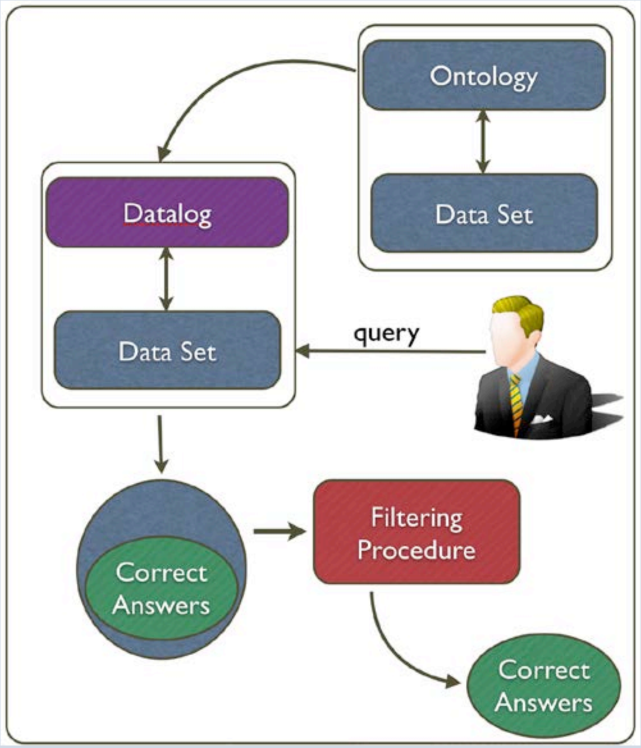
- **Over-approximate** O into Datalog program D
- **Evaluate** Q over D + DB



OWL 2 EL and Combined Approach

Given (RDF) data DB, EL ontology O and query Q:

- **Over-approximate** O into Datalog program D
- **Evaluate** Q over D + DB
- **Filter** result to eliminate spurious answers



Combined Approach — Issues

1 **Expressiveness**

- OWL 2 EL still relatively weak
- Lacks, e.g., counting, inverse, negation, disjunction

UNIVERSITY OF OXFORD DEPARTMENT OF COMPUTER SCIENCE

Information Systems Group

EPSRC Engineering and Physical Sciences Research Council

Optique

SEVENTH FRAMEWORK PROGRAMME

Combined Approach — Issues

1. **Expressiveness**
   - OWL 2 EL still relatively weak
   - Lacks, e.g., counting, inverse, negation, disjunction
2. **Scalability**
   - Dependent on performance of Datalog engine and/or blowup in size of data

University of Oxford Department of Computer Science
Information Systems Group
EPSRC Engineering and Physical Sciences Research Council
Optique
Seventh Framework Programme

OWL 2 RL and Materialisation

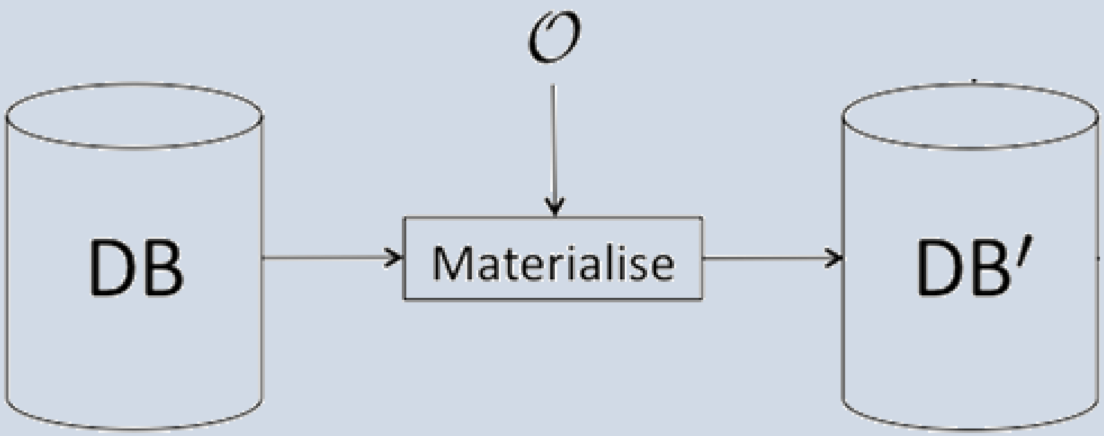
Given (RDF) data DB, RL ontology **O** and query **Q**:

OWL 2 RL and Materialisation

Given (RDF) data DB, RL ontology **O** and query **Q**:

- **Materialise** (RDF) data DB $\rightarrow$ DB^0 s.t. evaluating Q w.r.t. DB^0 equivalent to answering Q w.r.t. DB and O

  nb: Closely related to **chase** procedure used with DB dependencies



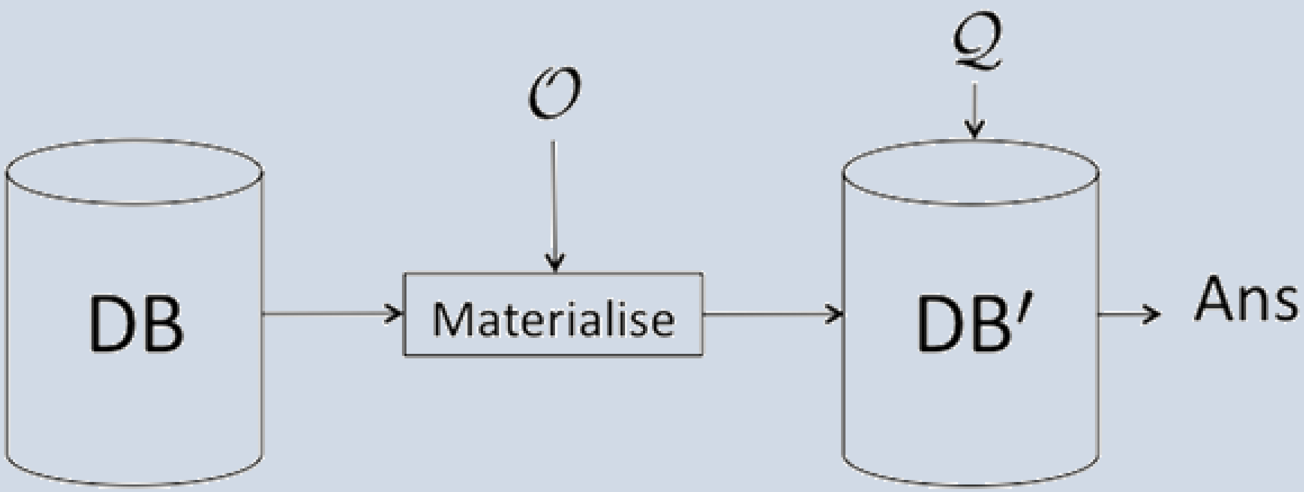
OWL 2 RL and Materialisation

Given (RDF) data DB, RL ontology **O** and query **Q**:

- **Materialise** (RDF) data DB $\rightarrow$ DB^0 s.t. evaluating Q w.r.t. DB^0 equivalent to answering Q w.r.t. DB and O

  nb: Closely related to **chase** procedure used with DB dependencies

- **Evaluate** Q against DB^0



Materialisation — Example

$$\mathcal{O} \left\{ \begin{array}{r} \exists \mathsf{treats}.\mathsf{Patient} \sqsubseteq \mathsf{Doctor} \\ \mathsf{Consulatant} \sqsubseteq \mathsf{Doctor} \end{array} \right.$$

$$\mathrm{DB} \left\{ \begin{array}{l} \mathsf{treats}(d_1, p_1) \\ \mathsf{Patient}(p_1) \\ \mathsf{Doctor}(d_2) \\ \mathsf{Consultant}(c_1) \end{array} \right.$$

Materialisation — Example

$$\mathcal{O}\left\{\begin{array}{r}\exists\mathsf{treats}.\mathsf{Patient} \sqsubseteq \mathsf{Doctor}\\ \mathsf{Consulatant} \sqsubseteq \mathsf{Doctor}\end{array}\right.$$

$$\text{DB}\left\{\begin{array}{l}\mathsf{treats}(d_1, p_1)\\ \mathsf{Patient}(p_1)\\ \mathsf{Doctor}(d_2)\\ \mathsf{Consultant}(c_1)\end{array}\right.$$

$$\text{DB}^0\left\{\begin{array}{l}\mathsf{treats}(d_1, p_1)\\ \mathsf{Patient}(p_1)\\ \mathsf{Doctor}(d_2)\\ \mathsf{Consultant}(c_1)\\ \mathsf{Doctor}(d_1)\\ \mathsf{Doctor}(c_1)\end{array}\right.$$

Materialisation — Example

$$\mathcal{O}\left\{\begin{array}{r}\exists\mathsf{treats}.\mathsf{Patient} \sqsubseteq \mathsf{Doctor}\\ \mathsf{Consulatant} \sqsubseteq \mathsf{Doctor}\end{array}\right.$$

$$\text{DB}\left\{\begin{array}{l}\mathsf{treats}(d_1, p_1)\\ \mathsf{Patient}(p_1)\\ \mathsf{Doctor}(d_2)\\ \mathsf{Consultant}(c_1)\end{array}\right.$$

$$\text{DB}^0\left\{\begin{array}{l}\mathsf{treats}(d_1, p_1)\\ \mathsf{Patient}(p_1)\\ \mathsf{Doctor}(d_2)\\ \mathsf{Consultant}(c_1)\\ \mathsf{Doctor}(d_1)\\ \mathsf{Doctor}(c_1)\end{array}\right.$$

$$\mathcal{Q}_1 \quad Q(x) \leftarrow \mathsf{Doctor}(y)$$

Materialisation — Example

$$\mathcal{O}\left\{\begin{array}{r}\exists\text{treats}.\text{Patient} \sqsubseteq \text{Doctor}\\ \text{Consulatant} \sqsubseteq \text{Doctor}\end{array}\right.$$

$$\text{DB}\left\{\begin{array}{l}\text{treats}(d_1, p_1)\\ \text{Patient}(p_1)\\ \text{Doctor}(d_2)\\ \text{Consultant}(c_1)\end{array}\right.$$

$$\text{DB}^0\left\{\begin{array}{l}\text{treats}(d_1, p_1)\\ \text{Patient}(p_1)\\ \text{Doctor}(d_2)\\ \text{Consultant}(c_1)\\ \text{Doctor}(d_1)\\ \text{Doctor}(c_1)\end{array}\right.$$

$$\mathcal{Q}_1 \quad Q(x) \leftarrow \text{Doctor}(y) \qquad \rightsquigarrow \qquad \{d_2, d_1, c_1\}$$

Materialisation — Issues

1. **Expressiveness**

- RL still relatively weak
- Asymmetrical – problematical for definitions (bi-implications)
- Many realistic ontologies use (at least) existentials on RHS

Materialisation — Issues

1 **Expressiveness**
- RL still relatively weak
- Asymmetrical – problematical for definitions (bi-implications)
- Many realistic ontologies use (at least) existentials on RHS

2 **Updates**
- Additions relatively easy (continue materialisation)
- Retraction more difficult – but incremental reasoning possible using view maintenance techniques

Materialisation — Issues

1 **Expressiveness**

- RL still relatively weak
- Asymmetrical – problematical for definitions (bi-implications)
- Many realistic ontologies use (at least) existentials on RHS

2 **Updates**

- Additions relatively easy (continue materialisation)
- Retraction more difficult – but incremental reasoning possible using view maintenance techniques

3 **Scalability**

- Dependent on performance of Datalog engine

Scalability: RDFox Datalog Engine

- Efficient **Datalog/RL** engine critical for both RL and EL

Scalability: RDFox Datalog Engine

- Efficient **Datalog/RL** engine critical for both RL and EL
- Existing approaches mainly focus on **map reduce**
  - high communication overhead
  - redundant computation
  - query answering over (distributed) materialized data is problematic

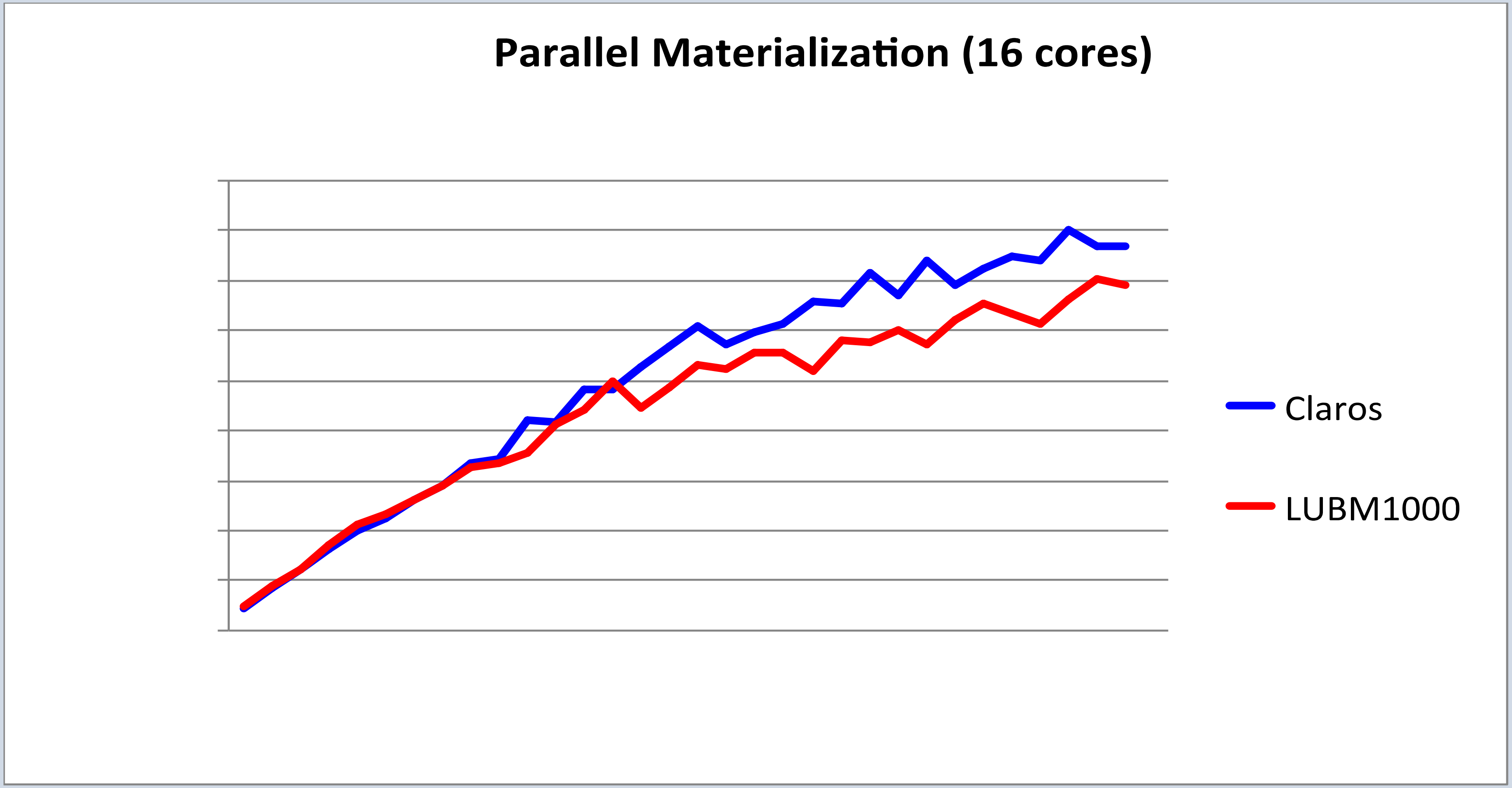
Scalability: RDFox Datalog Engine

- Efficient **Datalog/RL** engine critical for both RL and EL
- Existing approaches mainly focus on **map reduce**
  - high communication overhead
  - redundant computation
  - query answering over (distributed) materialized data is problematic
- **RDFox** is a new Datalog/RL engine with novel features
  - parallel materialization with fine-grained load balancing
  - highly optimized in-memory data storage with ‘mostly’ lock-free parallel inserts
  - £4,000 desktop with 128 GB can store around 2×10^9 triples

Scalability: RDFox Datalog Engine

	Claros	LUBM1000-UB
Memory Usage		
Base Triples	19M	134M
Base Mem	1GB	9GB
Mat. Triples	96M	333M
Mat. Mem	5GB	16GB
Time (16 cores)		
1 thread	2274s	942s
16 thread	180s	96s
32 thread	132s	66s

Scalability: RDFox Datalog Engine



Information Systems Group

EPSRC
Engineering and Physical Sciences Research Council

Optique

SEVENTH FRAMEWORK PROGRAMME

UNIVERSITY OF OXFORD
DEPARTMENT OF COMPUTER SCIENCE

Expressiveness: Chase Materialisation

- Applicable to **acyclic** ontologies
  - Acyclicity can be checked using, e.g., graph based techniques (weak acyclicity, joint acyclicity, etc.)
  - Many realistic ontologies turn out to be acyclic
- Given acyclic ontology **O**, can apply chase materialisation:
  - Ontology translated into existential rules (aka dependencies)
  - Existential rules can introduce fresh Skolem individuals
  - Termination guaranteed for acyclic ontologies

Expressiveness: Chase Materialisation

$$\mathcal{O} \begin{cases} \exists \mathsf{treats}.\mathsf{Patient} \sqsubseteq \mathsf{Doctor} \\ \mathsf{Consulatant} \sqsubseteq \mathsf{Doctor} \end{cases}$$

$$\mathsf{DB} \begin{cases} \mathsf{treats}(d_1, p_1) \\ \mathsf{Patient}(p_1) \\ \mathsf{Doctor}(d_2) \\ \mathsf{Consultant}(c_1) \end{cases} \qquad \mathsf{DB}^0 \begin{cases} \mathsf{treats}(d_1, p_1) \\ \mathsf{Patient}(p_1) \\ \mathsf{Doctor}(d_2) \\ \mathsf{Consultant}(c_1) \\ \mathsf{Doctor}(d_1) \\ \mathsf{Doctor}(c_1) \end{cases}$$

$$\mathcal{Q}_1 \quad Q(x) \leftarrow \mathsf{Doctor}(y) \qquad \rightsquigarrow \qquad \{d_2, d_1, c_1\}$$

Expressiveness: Chase Materialisation

$$\mathcal{O} \begin{cases} \exists \mathsf{treats}.\mathsf{Patient} \equiv \mathsf{Doctor} \\ \mathsf{Consulatant} \sqsubseteq \mathsf{Doctor} \end{cases}$$

$$\text{DB} \begin{cases} \mathsf{treats}(d_1, p_1) \\ \mathsf{Patient}(p_1) \\ \mathsf{Doctor}(d_2) \\ \mathsf{Consultant}(c_1) \end{cases} \qquad \text{DB}^0 \begin{cases} \mathsf{treats}(d_1, p_1) \\ \mathsf{Patient}(p_1) \\ \mathsf{Doctor}(d_2) \\ \mathsf{Consultant}(c_1) \\ \mathsf{Doctor}(d_1) \\ \mathsf{Doctor}(c_1) \end{cases}$$

$$\mathcal{Q}_1 \quad Q(x) \leftarrow \mathsf{Doctor}(y) \qquad \rightsquigarrow \qquad \{d_2, d_1, c_1\}$$

Expressiveness: Chase Materialisation

$$\mathcal{O} \begin{cases} \exists \mathsf{treats}.\mathsf{Patient} \equiv \mathsf{Doctor} \\ \mathsf{Consulatant} \sqsubseteq \mathsf{Doctor} \end{cases}$$

$$\mathrm{DB} \begin{cases} \mathsf{treats}(d_1, p_1) \\ \mathsf{Patient}(p_1) \\ \mathsf{Doctor}(d_2) \\ \mathsf{Consultant}(c_1) \end{cases} \qquad \mathrm{DB}^0 \begin{cases} \mathsf{treats}(d_1, p_1) \\ \mathsf{Patient}(p_1) \\ \mathsf{Doctor}(d_2) \\ \mathsf{Consultant}(c_1) \\ \mathsf{Doctor}(d_1) \\ \mathsf{Doctor}(c_1) \end{cases}$$

$$\mathcal{Q}_1 \quad Q(x) \leftarrow \mathsf{Doctor}(y) \quad \leadsto \quad \{d_2, d_1, c_1\}$$

$$\mathcal{Q}_2 \quad Q(x) \leftarrow \mathsf{treats}(x, y) \wedge \mathsf{Patient}(y) \quad \leadsto \quad \{d_1\}$$

Expressiveness: Chase Materialisation

$$\mathcal{O}\left\{\begin{array}{r}\exists\textsf{treats}.\textsf{Patient} \equiv \textsf{Doctor}\\ \textsf{Consulatant} \sqsubseteq \textsf{Doctor}\end{array}\right.$$

$$\text{DB}\left\{\begin{array}{l}\textsf{treats}(d_1, p_1)\\ \textsf{Patient}(p_1)\\ \textsf{Doctor}(d_2)\\ \textsf{Consultant}(c_1)\end{array}\right.$$

$$\text{DB}^0\left\{\begin{array}{l}\textsf{treats}(d_1, p_1)\\ \textsf{Patient}(p_1)\\ \textsf{Doctor}(d_2)\\ \textsf{Consultant}(c_1)\\ \textsf{Doctor}(d_1)\\ \textsf{Doctor}(c_1)\\ \textsf{treats}(d_2, f(d_2))\\ \textsf{Patient}(f(d_2))\\ \textsf{treats}(c_1, f(c_1))\\ \textsf{Patient}(f(c_1))\end{array}\right.$$

Skolems

$$\mathcal{Q}_1 \quad Q(x) \leftarrow \textsf{Doctor}(y) \quad \rightsquigarrow \quad \{d_2, d_1, c_1\}$$

$$\mathcal{Q}_2 \quad Q(x) \leftarrow \textsf{treats}(x, y) \wedge \textsf{Patient}(y) \quad \rightsquigarrow \quad \{d_1\}$$

Expressiveness: Chase Materialisation

$$\mathcal{O}\left\{\begin{array}{rcl} \exists\mathsf{treats}.\mathsf{Patient} & \equiv & \mathsf{Doctor} \\ \mathsf{Consulatant} & \sqsubseteq & \mathsf{Doctor} \end{array}\right.$$

$$\text{DB}\left\{\begin{array}{l} \mathsf{treats}(d_1, p_1) \\ \mathsf{Patient}(p_1) \\ \mathsf{Doctor}(d_2) \\ \mathsf{Consultant}(c_1) \end{array}\right.$$

$$\text{DB}^0\left\{\begin{array}{l} \mathsf{treats}(d_1, p_1) \\ \mathsf{Patient}(p_1) \\ \mathsf{Doctor}(d_2) \\ \mathsf{Consultant}(c_1) \\ \mathsf{Doctor}(d_1) \\ \mathsf{Doctor}(c_1) \\ \mathsf{treats}(d_2, f(d_2)) \\ \mathsf{Patient}(f(d_2)) \\ \mathsf{treats}(c_1, f(c_1)) \\ \mathsf{Patient}(f(c_1)) \end{array}\right.$$

Skolems

$$\mathcal{Q}_1 \quad Q(x) \leftarrow \mathsf{Doctor}(y) \quad \rightsquigarrow \quad \{d_2, d_1, c_1\}$$

$$\mathcal{Q}_2 \quad Q(x) \leftarrow \mathsf{treats}(x, y) \wedge \mathsf{Patient}(y) \quad \rightsquigarrow \quad \{d_1, d_2, c_1\}$$

Expressiveness: Lower and Upper Bounds

- RL reasoning w.r.t. RL+ ontology **O** gives lower bound answer ***L***

Expressiveness: Lower and Upper Bounds

- RL reasoning w.r.t. RL+ ontology **O** gives lower bound answer ***L***

- Transform **O** into strictly stronger OWL RL ontology $\mathbf{O}^{\mathbf{0}}$
  - Transform ontology into Datalog$^{\pm,\vee}$ rules
  - Eliminate $\vee$ by transforming to $\wedge$
  - Eliminate existentials by replacing with Skolem constants
  - Discard rules with empty heads (assuming **O** satisfiable)
  - Transform rules into OWL 2 RL ontology $\mathbf{O}^{\mathbf{0}}$

Expressiveness: Lower and Upper Bounds

- RL reasoning w.r.t. RL+ ontology **O** gives lower bound answer ***L***

- Transform **O** into strictly stronger OWL RL ontology $\mathbf{O}^{\mathbf{0}}$
  - Transform ontology into Datalog$^{\pm,\vee}$ rules
  - Eliminate $\vee$ by transforming to $\wedge$
  - Eliminate existentials by replacing with Skolem constants
  - Discard rules with empty heads (assuming **O** satisfiable)
  - Transform rules into OWL 2 RL ontology $\mathbf{O}^{\mathbf{0}}$

- Datalog/RL reasoning w.r.t. $\mathbf{O}^{\mathbf{0}}$ gives upper bound answer ***U***

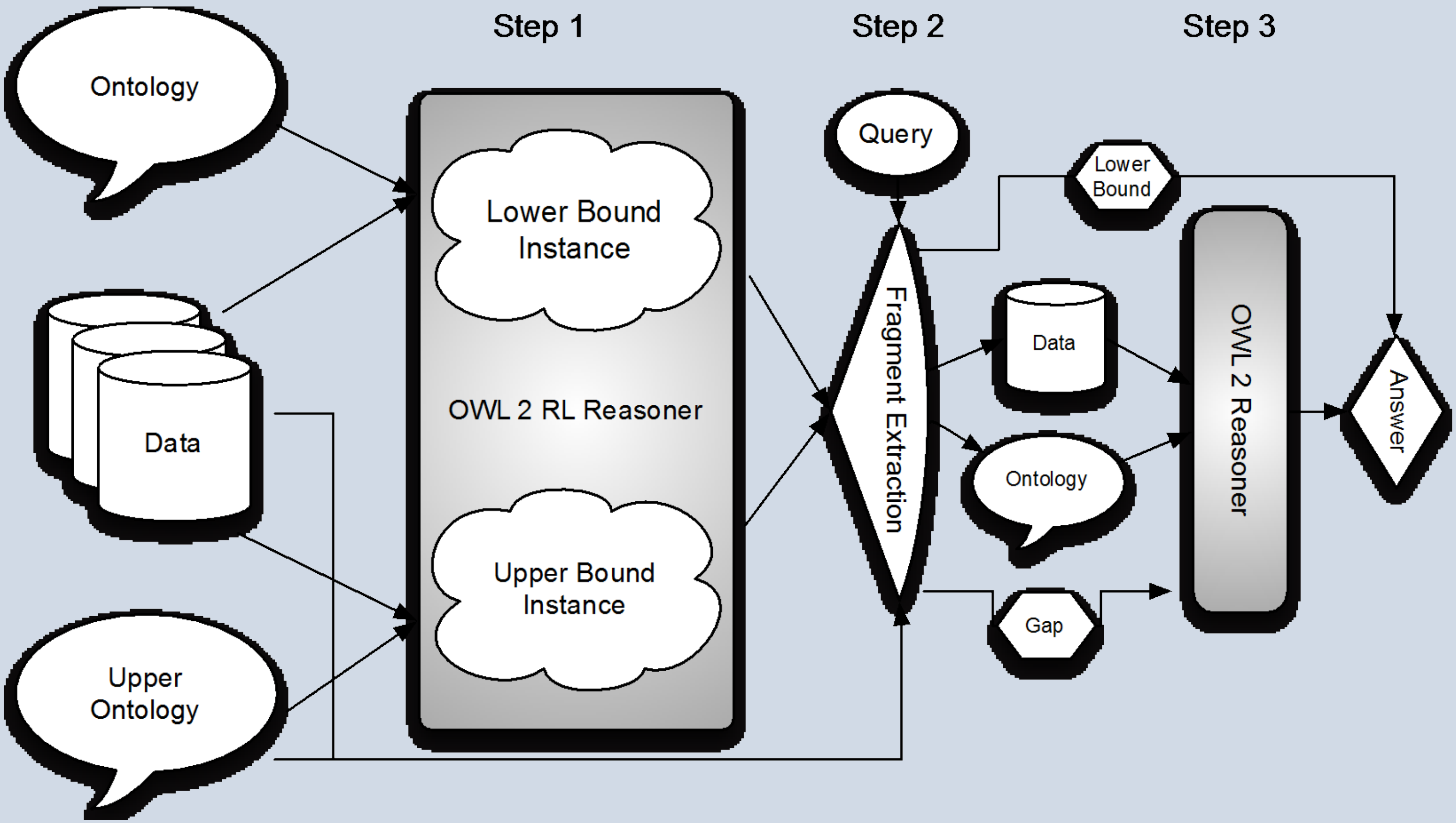
Expressiveness: Lower and Upper Bounds

- If ***L*** = ***U***, then both answers are sound and complete

Expressiveness: Lower and Upper Bounds

- If $L = U$, then both answers are sound and complete
- If $L \neq U$, then $U \setminus L$ identifies a (small) set of "possible" answers
  - Delineates range of uncertainty
  - Can more efficiently check possible answers using, e.g., HermiT (but still infeasible if dataset is large)
  - Can use $U \setminus L$ to identify small(er) "relevant" subset of axioms/data needed to check possible answers

Expressiveness: Lower and Upper Bounds



Performance on LUBM 40

time to extract the fragment

Time to check all tuples in U\L by an OWL 2 reasoner using the fragment

triples in the query

Query	$\lvert V \rvert$	n	$\lvert G \rvert$	t_f	$\lvert \mathcal{O}_f \rvert$	$\lvert \mathcal{D}_f \rvert$	t_{check}	t_{total}
M_1	2	3	39	36.4	6	29041	H: 23.3	H: 60.7
M_2	3	4	1	37.1	6	29004	H: 4.0	H: 42.1
M_3	4	6	16	38.2	6	29054	H: 8.4	H: 47.6
M_4	2	3	30	36.0	6	29032	H: 23.3	H: 60.2
M_5	3	4	4	39.4	6	29010	H: 24.0	H: 64.3
M_6	4	6	29	2,845.8	10	87209	H: 483.0	H: 3339.4
M_7	3	5	15	38.0	6	29033	H: 10.3	H: 49.3
M_8	3	5	14	39.3	6	29038	H: 11.9	H: 52.2
M_9	3	4	10	328.9	12	86785	H: 556.2	H: 886.7
S	1	2	39	310.0	12	86802	H: 1,780.0 P: 16,592.1	H: 2,126.5 P: 16,870.0

answers in the gap U\L

variables in the query

total time to compute answers for the query

DEPARTMENT OF COMPUTER SCIENCE
UNIVERSITY OF OXFORD
Information Systems Group
EPSRC Engineering and Physical Sciences Research Council
Optique
SEVENTH FRAMEWORK PROGRAMME

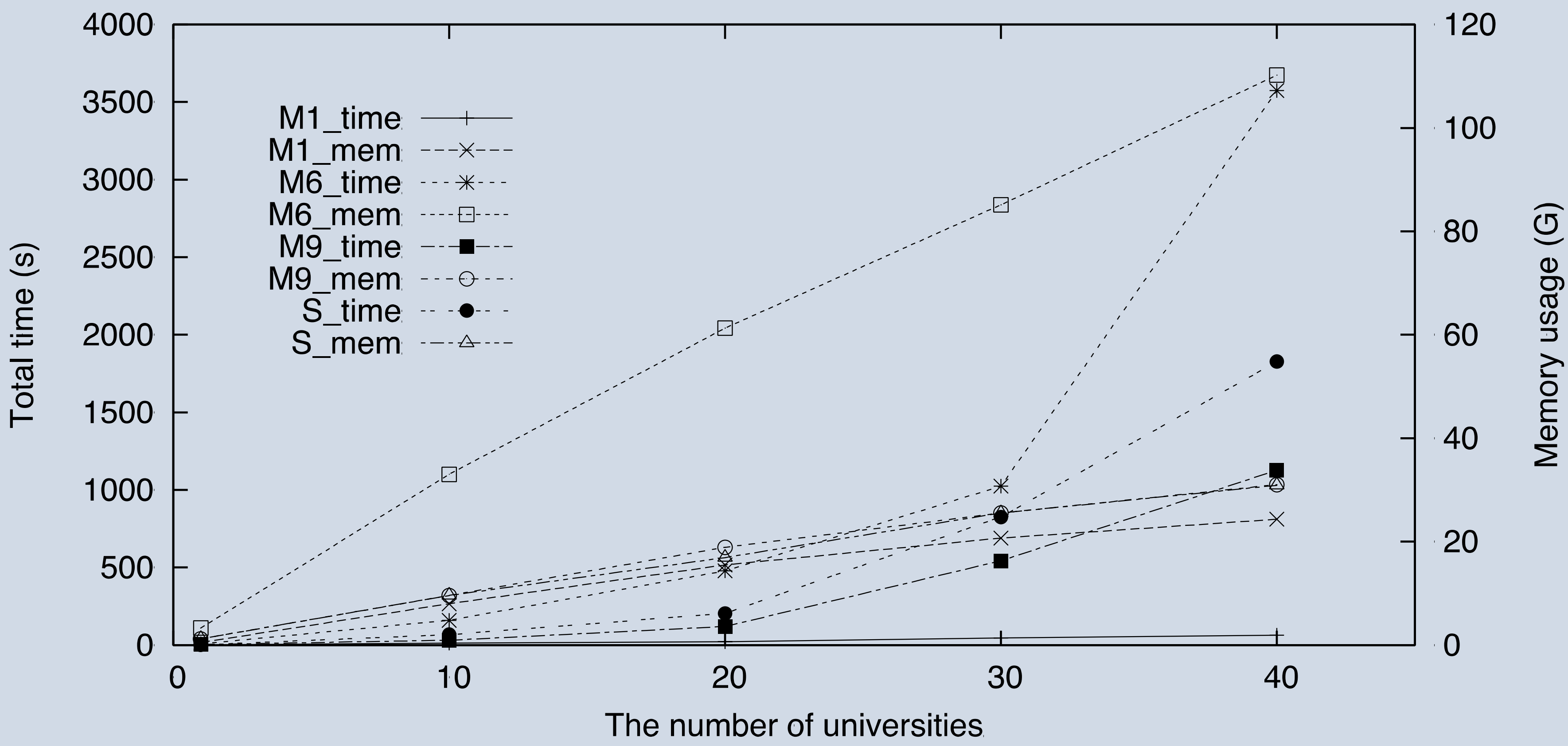
Performance on Fly

Query	$\lvert V\rvert$	n	$\lvert G\rvert$	t_f	$\lvert \mathcal{O}_f\rvert$	$\lvert \mathcal{D}_f\rvert$	t_{check} (H)	t_{total}	t_{HermiT}
Q_1	2	3	803	108.9	224	4515	45.9	155.2	3,465.9
Q_2	3	5	342	97.7	224	4054	16.0	114.0	3,179.0
Q_3	1	1	28	91.0	217	3712	0.9	92.3	5,863.3
Q_4	2	3	25	94.3	233	3762	4.7	99.2	2,944.3
Q_5	2	2	518	100.3	222	3712	24.0	124.6	3,243.7

Time to check all tuples in U\L by an OWL 2 reasoner using the original ontology

UNIVERSITY OF OXFORD DEPARTMENT OF COMPUTER SCIENCE
Information Systems Group
EPSRC Engineering and Physical Sciences Research Council
Optique
SEVENTH FRAMEWORK PROGRAMME

Scalability on LUBM



UNIVERSITY OF OXFORD DEPARTMENT OF COMPUTER SCIENCE
Information Systems Group
EPSRC Engineering and Physical Sciences Research Council
Optique
SEVENTH FRAMEWORK PROGRAMME

Future Work

- Tighten lower and upper bounds
- Use Datalog reasoner to compute relevant subsets
- Hybrid approaches, e.g., exploiting ELHO filtering

Discussion

- QL-Rewriting has many advantages
  - Data can be left untouched and in legacy storage
  - Exploits existing DB infrastructure and scalability
  - …

UNIVERSITY OF OXFORD DEPARTMENT OF COMPUTER SCIENCE
Information Systems Group
EPSRC Engineering and Physical Sciences Research Council
Optique
SEVENTH FRAMEWORK PROGRAMME

Discussion

- QL-Rewriting has many advantages
  - Data can be left untouched and in legacy storage
  - Exploits existing DB infrastructure and scalability
  - …
- But what if more expressiveness is needed?
  - Query answering for EL and RL still tractable (polynomial)
  - Critically depend on Datalog scalability – RDFox to the rescue!
  - Chase and LB/UB techniques offer potential for empirical scalability beyond EL and RL fragments

Acknowledgements

EPSRC Engineering and Physical Sciences Research Council

SIXTH FRAMEWORK PROGRAMME

SEVENTH FRAMEWORK PROGRAMME

DARPA

UNIVERSITY OF OXFORD DEPARTMENT OF COMPUTER SCIENCE

Information Systems Group

EPSRC Engineering and Physical Sciences Research Council

Optique

SEVENTH FRAMEWORK PROGRAMME

References

[1] Kontchakov, Lutz, Toman, Wolter, Zakharyaschev: The Combined Approach to Ontology-Based Data Access. IJCAI 2011.

[2] Stefanoni, Motik, Horrocks: Small Datalog Query Rewritings for EL. DL 2012

[3] Rodriguez-Muro, Calvanese: High Performance Query Answering over DL-Lite Ontologies. KR 2012

[4] Motik, Horrocks, and Kim. Delta-reasoner: a semantic web reasoner for an intelligent mobile platform. In Proc. of WWW 2012.

[5] Cuenca Grau et al. Acyclicity Notions for Existential Rules and Their Application to Query Answering in Ontologies. JAIR, 47:741-808, August 2013.

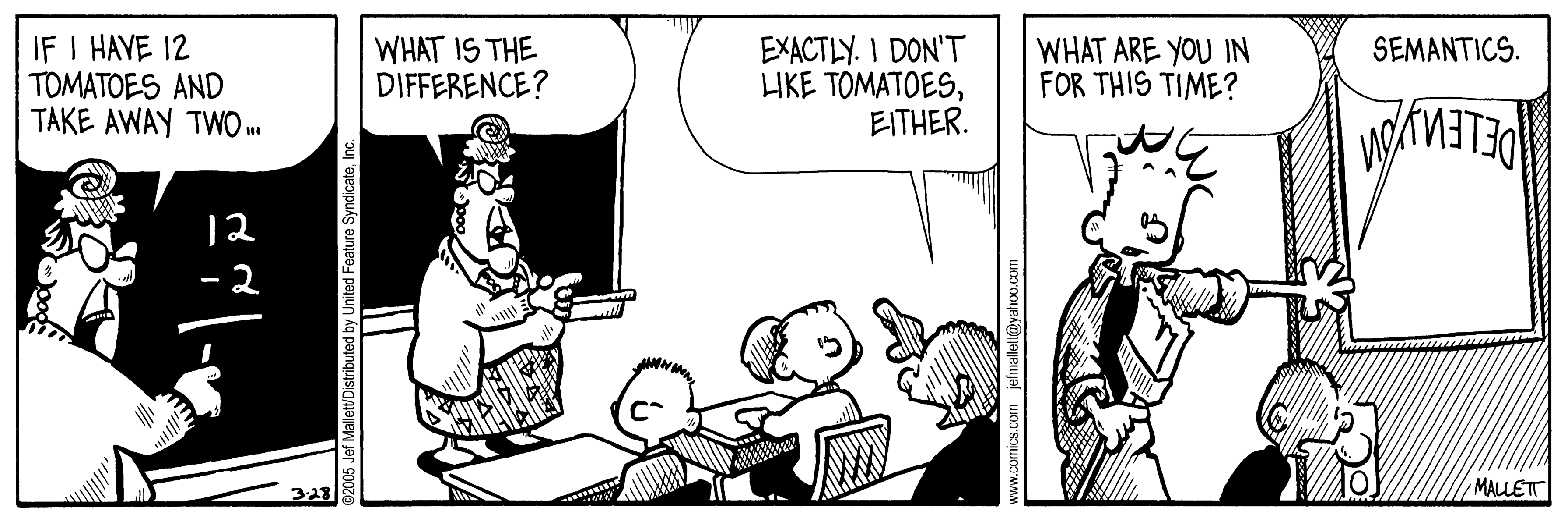
[6] Zhou, Cuenca Grau, and Horrocks. Making the Most of your Triple Store: Query Answering in OWL 2 Using an RL Reasoner. In Proc. of WWW 2013.

[7] Yujiao Zhou, Yavor Nenov, Bernardo Cuenca Grau, and Ian Horrocks. Complete Query Answering Over Horn Ontologies Using a Triple Store. ISWC 2013.

[8] Giorgio Stefanoni, Boris Motik, and Ian Horrocks. Introducing Nominals to the Combined Query Answering Approaches for EL. AAAI 2013.

[9] RDFox http://www.cs.ox.ac.uk/isg/tools/RDFox/

Thank you for listening



Any questions?