Existential Rules:
a Study Through Chase Termination, FO-Rewritability and Boundedness

Marie-Laure Mugnier

University of Montpellier
Query (SQL, SPARQL, MongoDB ...)

« find all patients affected by a lung disease due to a bacteria »

??

ID Patient  Diagnosis
p  « legionella »

Database (relational, RDF, NoSQL, ...)

« find all patients affected by a lung disease due to a bacteria »
A legionella is bacterial pneumonia
A bacterial pneumonia is a pneumonia
A pneumonia is a lung disease
A bacterial pneumonia is caused by a bacteria
If x is caused by y then x is due to y
If the diagnosis of a patient x contains a disease y then x is affected by y

The answers to the query are inferred from the KB
Adding an Ontological Layer on Top of Data (2)

q(x) = ∃y. ∃z. Patient(x) ∧ isAffectedBy(x,y) ∧ LungDisease(y) ∧ dueTo(y,z) ∧ Bacteria(z)

Factbase = { Patient(p), Diagnosis(p,d), Legionella(d) }

∀ x. Legionella(x) → LungDisease(x) ∧ BacterialDisease(x)  
    hence LungDisease(d) and BacterialDisease(d)

∀ x. LungDisease(x) → Disease(x)  
    hence Disease(d)

∀ x. BacterialDisease(x) → ∃ z. hasCausativeAgent(x,z) ∧ Bacteria(z)  
    hence hasCausativeAgent(d,z₀) and Bacteria(z₀)

∀ x. ∀ y. hasCausativeAgent(x,y) → dueTo(x,y)  
    hence dueTo(d,z₀)

∀ x. ∀ y. Diagnosis(x,y) ∧ Disease(y) → isAffectedBy(x,y)  
    hence isAffectedBy(p,d)

Answer : x = p

Implicit assumption here: relevant data naturally seen as a factbase on the application vocabulary

« Find all patients affected by a lung disease due to a bacteria »

« The diagnosis for the patient p is legionella »
**Ontology-Based Data Access (OBDA)**

- **Conceptual level**
  - Query
  - Ontology
  - Factbase
  - Mappings from data to facts
  - Relational Database
  - Preexisting data source

- **Query using the vocabulary of the ontology**

- **Description of the application domain with a high abstraction level**

- **Factbase (possibly virtual)** using the vocabulary of the ontology

---

[RuleML+RR] [Poggi et al. JoDS 2008]
**Mappings: Linking Data to Ontologies**

| Patient_T [ID_PATIENT, NAME, SSN] | Patient /1  
|                                  | Legionella /1  
| Diagnosis_T [ID_PATIENT, DISORDER] | Diagnosis /2 |

**Mapping: database query(X) \( \sim \) conjunction of atoms with free variables \( X \)**

\[
q(x): \exists n. \exists s. \text{Patient}_T (x,n,s) \sim \text{Patient}(x)
\]

\[
q'(x): \exists n. \exists s. \text{Patient}_T (x,n,s) \land \text{Diagnosis}_T (x,y) \land y = \text{« Legionella »} \sim \exists z. \text{diagnosis}(x,z) \land \text{Legionella}(z)
\]

<table>
<thead>
<tr>
<th>Patient_T</th>
<th>Diagnosis_T</th>
<th>GLAV</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>id</td>
<td></td>
</tr>
<tr>
<td>name</td>
<td>dis</td>
<td></td>
</tr>
<tr>
<td>ssn</td>
<td></td>
<td></td>
</tr>
<tr>
<td>p</td>
<td>p</td>
<td></td>
</tr>
<tr>
<td>..</td>
<td>« Leg. »</td>
<td></td>
</tr>
<tr>
<td>..</td>
<td>..</td>
<td></td>
</tr>
<tr>
<td>..</td>
<td>..</td>
<td></td>
</tr>
</tbody>
</table>

RuleML+RR
GENERALIZED OBDA: HETEROGENEOUS DATA SOURCES

Conceptual level

Query

Ontology

Factbase

Mappings from data to facts
{ Database native query $\leadsto$ Facts }

Independent and heterogeneous data sources
**EXISTENTIAL RULES AS AN ONTOLOGICAL LANGUAGE**

∀X. ∀Y. Body [X,Y] → ∃ Z. Head [X,Z]

sets of variables

X, Y, Z :

any positive conjunction (without functional symbols except constants)

∀x. ∀y. siblingOf(x,y) → ∃ z. parentOf(z,x) ∧ parentOf(z,y)

(Universal quantifiers will be omitted)

Key point: ability to assert the existence of unknown entities

Crucial for representing ontological knowledge in open domains

∀x. ∀y. siblingOf(x,y) → ∃ z. parentOf(z,x) ∧ parentOf(z,y)

∃ z₀. siblingOf(a,b) ∧ parentOf(z₀,a) ∧ parentOf(z₀,b)
THEORETICAL FOUNDATIONS

Graph-based KR
[Chein Mugnier 1992, 2009]

Datalog (70-80s)

\( \forall \exists \) -rules, existential Rules [Baget+ IJCAI 2009]

Datalog+/- [Cali+ PODS 2009]

Logical translation of graph rules

- « value invention »
- unrestricted cycles on variables
- unbounded arity

Lightweight Description Logics, e.g. OWL 2 tractable profiles
More generally, Horn DLs

- Same logical form as « Tuple-Generating Dependencies » (TGDs)
  long studied in database theory
MAPPINGS CAN ALSO BE SEEN AS EXISTENTIAL RULES

Patient_T [ID_PATIENT, NAME, SSN]

Diagnosis_T [ID_PATIENT, DISORDER]

q(x): \( \exists n \exists s \) Patient_T (x, n, s) \( \leadsto \) Patient(x)

q'(x): \( \exists n \exists s \) Patient_T (x, n, s) \( \land \) Diagnosis_T (x, y) \( \land \) y = « Legionella »
\( \leadsto \) \( \exists z \) diagnosis(x, z) \( \land \) Legionella(z)

More generally: \( q_1(X) \leadsto q_2(X) \) where \( q_1 \) is expressed in a native query language

Decomposition of a mapping into 2 mappings

low level : \( q_1(X) \leadsto \text{view}(X) \)

high level : \( \text{view}(X) \rightarrow q_2(X) \) existential rule
EXISTENTIAL RULES AS A UNIFORM LANGUAGE FOR OBDA

Conceptual level

Query

Ontology

Factbase

(Union of) Conjunctive Queries

Existential Rules

Existentially closed conjunction of atoms

Mappings

1. source-to-target existential rules
   (G)LAV mappings
2. low level mappings

Data

Data

Data
**FUNDAMENTAL QUERY ANSWERING DECISION PROBLEM**

**Knowledge Base** $\mathcal{K} = (F, \mathcal{R})$

**Query** $q$

**CQ Entailment**

*Input:* 
- a Boolean conjunctive query $q$
- a knowledge base $\mathcal{K}$

*Question:* 
is $q$ entailed by $\mathcal{K}$ ($\mathcal{K} \models q$)?

Undecidable for general existential rules but many decidable subclasses are known.
1. **FORWARD CHAINING (CHASE)**

For any (Boolean) conjunctive query $q$

$K \models q$ iff

$\text{chase}(K) \models q$ iff

$q$ maps to $\text{chase}(K)$ by homomorphism

Of course the chase may no halt
2. **QUERY REWRITING**

Backward chaining decomposed into 2 steps

[DL-Lite]

Rewriting into a **first-order query** (= core SQL query), typically a union of conjunctive queries independently from any factbase

For any $F$, $F, \mathcal{R} \models q$ iff $F \models Q$

Of course there may be no finite rewriting of $q$

A set of rules $\mathcal{R}$ is **FO-rewritable**

if for any query $q$, there is a (finite) FO-rewriting of $q$ with $\mathcal{R}$
(VERY PARTIAL) MAP OF DECIDABLE CLASSES

chase termination

FO-rewritability

acyclic Graph of Rule Dependencies

w-sticky-join

w-sticky

sticky-join

sticky

w-sticky-guarded

jointly-fg

weakly-guarded

frontier-guarded

wa-GRD

super-wa

wa-GRD

datalog

RMFA

MFA

FO-rewritability

DL-Lite_R

linear

EC

RuleML+RR
DIFFERENT VARIANTS OF THE CHASE

All variants compute **universal models** of the KB
but they differ on how they handle **redundancies** possibly caused by nulls

\[
p(a,b), p(b,c)\]

\[
p(a,b), p(b,c), \exists z_0 \ p(a,z_0) \ p(z_0,c)\]

**Core**: set of atoms without homomorphism to one of its strict subsets
OBLIVIOUS / SEMI-OBLIVIOUS = SKOLEM CHASE

Oblivious (or naive): all homomorphisms from rule bodies

\[ R = p(x, y) \rightarrow \exists z \ p(x, z) \]

\[ F = p(a, b) \]

\[ p(a, z_0) \]

\[ p(a, z_1) \]

\[ p(a, z) \]

\[ \exists z \ p(x, z) \]

\[ \forall x \exists z p(x, z) \]

Semi-oblivious: homomorphisms that differ on the rule frontier \((x)\)

« isomorphic » to the Skolem chase:
(1) skolemize rules
(2) perform oblivious chase on skolemized rules

\[ R = p(x, y) \rightarrow p(x, f(x)) \]

Stupid rules to keep examples simple!

Infinite chase

RuleML+RR
\textbf{RESTRICTED (AKA STANDARD) CHASE}

\textbf{Restricted}: do not perform a rule application that brings \textit{only} redundant information

ie do not consider homomorphism \( h: \text{body} \rightarrow \text{facts} \)
if \( h \) can be extended to homomorphism \( h': \text{body} \cup \text{head} \rightarrow \text{facts} \)

\( F: p(a,b) \quad R : p(x,y) \rightarrow \exists z \ p(y,z), \ p(z,y) \)

\begin{align*}
a & \quad b \quad z_0 \quad z_1 \quad \ldots \\
& \quad \downarrow \quad \downarrow \quad \downarrow \\
& \quad z_2
\end{align*}

(semi-) oblivious chase:

infinite

restricted chase:

halts after one rule application
For the same KB, some derivations may halt while others may not

\[ F : p(a,b) \quad R_1 : p(x,y) \rightarrow \exists z \ p(y,z) \]
\[ R_2 : p(x,y) \rightarrow p(y,y) \]

If \( R_1 \) is always applied before \( R_2 \) for a given homomorphism of \( p(x,y) \):

If \( R_2 \) is applied first:

\[ a \rightarrow b \]
CORE CHASE

Iterate:

1. apply some rules as in the restricted chase
2. compute the core of the result

$F : p(a,b), q(b,b), q(a,z)$ where $z$ is a variable

$R_1 : p(x,y) \rightarrow q(x,y)$
$R_2 : q(x,y) \rightarrow \exists z \ q(y,z)$

The restricted chase only checks redundancy of **newly** added atoms $\Rightarrow$ infinite here

The core chase outputs $\{ p(a,b), q(b,b), q(a,b) \}$
IN SHORT

All chase variants compute **universal** models

They can be **strictly ordered wrt termination**: oblivious < semi-oblivious = skolem < restricted < core

Only the **core** chase is ensured to halt if the KB admits a **finite universal model** but it is costly (involves homomorphisms from the whole factbase)

The **restricted** chase seems to achieve a **good tradeoff** but it is **order dependent**

→ Is there an ordering strategy that terminates more often than the others? Maybe for some rule classes?
Chase termination problem: given a ruleset, does the chase terminate on all factbases?

(VERY PARTIAL) MAP OF DECIDABLE CLASSES

- super-wa
- wa-GRD
- weakly-acyclic
- datalog

Chase termination guaranteed

- SO-chase
- R-chase
- RMFA
- MFA

- (S)O-chase
- linear
- sticky
- guarded

acyclic Graph of Rule Dependencies

Core-chase

linear with atomic head
**BOUNDEDNESS PROBLEM (BREADTH-FIRST VERSION)**

Deeply studied in the 90’s for **datalog**

Key property for optimizations

**datalog boundedness:**
Given a datalog program, is there \( k \) such that breadth-first forward chaining halts within **at most** \( k \) **steps** on any factbase?

\[
\begin{align*}
 p(x,y), p(y,z) & \rightarrow p(x,z) \\
 p(x,y), p(y,z) & \rightarrow p(x,z) \\
 p(x,y), p(u,z) & \rightarrow p(x,z)
\end{align*}
\]

- **Unbounded**: \( \# \) steps depends on factbase size
- **Bounded**: \( \# \) steps = 1

**datalog boundedness is undecidable**
**BOUNDEDNESS ON EXISTENTIAL RULES**

**Existential rule C-boundedness** (parametrized by a chase variant C):
Given a set of rules \( \mathcal{R} \), is there \( k \) such that the breadth-first C-chase with \( \mathcal{R} \) halts within at most \( k \) steps on any factbase?

\[
\begin{align*}
R_1: & \quad p(x,y) \rightarrow \exists z \ q(x,z) \\
R_2: & \quad q(x,z) \rightarrow \exists w \ p(x,w)
\end{align*}
\]

not O-bounded (even infinite): \( p(a,b), q(a,z_0), p(a,w_0), q(a,z_1), p(a,w_1), \ldots \)

SO-bounded with \( k=2 \): e.g. \( p(a,b), q(a,z_0), p(a,w_0) \)

R-bounded with \( k=1 \): e.g. \( p(a,b), q(a,z_0) \)
For **datalog**: boundedness is equivalent to FO-rewritability

For **existential rules**:

C-boundedness $\Rightarrow$ FO-rewritability, for any chase C but the reciprocal is false:

$$p(x,y) \rightarrow \exists z \ p(y,z)$$

FO-rewritable but unbounded (even not core finite)

C-boundedness $\Rightarrow$ FO-rewritability and C-termination

What about the other direction?

IJCAI 2019
A CLOSER LOOK AT EXISTENTIAL RULES

Any existential rule can be decomposed into:

- a **fully-existential** rule: all head atoms have at least one existential variable
- a (set of) **datalog** rule(s): no existential in head atoms

\[
p(x,y) \rightarrow \exists z_1 \exists z_2 p(x,z_1), p(z_1,z_2), q(z_2), q(x), r(x,y)
\]

\[
p(x,y) \rightarrow \exists z_1 \exists z_2 p(x,z_1), p(z_1,z_2), q(z_2)
\]

\[
p(x,y) \rightarrow q(x)
\]

\[
p(x,y) \rightarrow r(x,y)
\]

preserves chase termination, FO-rewritability and boundedness
**Oblivious Boundedness**

For **fully-existential rulesets**:  
O-chase termination = O-boundedness

Hence, for fully existential rules, O-chase termination implies FO-rewritability

For **general existential rulesets**:  
O-chase termination \(\cap\) FO-rewritability = O-boundedness

Moreover, for O-chase terminating rulesets:

FO-rewritability on fully-atomic queries \(\Rightarrow\) FO-rewritability

[fully-atomic: 1 atom on answer variables]

O-chase termination \(\cap\) FO-rewritability[fully-atomic] = O-boundedness
**SEMI-OBLIVIOUS (= SKOLEM) BOUNDEDNESS**

SO-chase termination $\cap$ FO-rewritability = SO-boundedness

- Here FO-rewritability for any conjunctive query is required
- Furthermore, fully-existential rules behave as general rules

*Explanation:*
for the SO-chase, even a fully-existential rule has an « underlying datalog part »

$R: \text{Body} \rightarrow \text{Head}$

$\Psi$

$\text{Body} \rightarrow p_R(\text{frontier}(R))$

$p_R(\text{frontier}(R)) \rightarrow \text{Head}$

$R: p(x,y) \rightarrow \exists z \ p(x,z)$

$p(x,y) \rightarrow p_R(x)$

$p_R(x) \rightarrow \exists z \ p(x,z)$

$R$ ensures SO-chase termination iff $\Psi(R)$ ensures O-chase termination

$R$ is SO-bounded iff $\Psi(R)$ is O-bounded
**Complexity of Boundedness for Specific Rule Classes**

- Intrinsically bounded class: acyclic graph of rule dependencies (aGRD)

<table>
<thead>
<tr>
<th></th>
<th>datalog</th>
<th>Linear</th>
<th>Sticky</th>
<th>Guarded</th>
</tr>
</thead>
<tbody>
<tr>
<td>(S)O-chase</td>
<td>Undecidable</td>
<td>PSpace-c</td>
<td>PSpace-c</td>
<td>in 2Exptime</td>
</tr>
<tr>
<td>Restricted chase</td>
<td>(*)</td>
<td>in co-N2ExpTime</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>Core chase</td>
<td>(*)</td>
<td>in 2Exptime</td>
<td>?</td>
<td>?</td>
</tr>
</tbody>
</table>

Calautti & al. PODS 2015: complexity (S)O-chase termination for linear and guarded
Calautti & al. ICDT 2019: complexity (S)O-chase termination for sticky

Barcello & al. IJCAI 2018: complexity of FO-rewritability of a single CQ for guarded
+ Bourhis & al. IJCAI 2019: bound on # queries to be considered

(*) Leclère & al. ICDT 2019: complexity restricted- and core chase termination for linear restricted to atomic heads
From Hernich ICDT 2012: decidability core chase termination for linear rules
**RELATED PROBLEM: k-BOUNDEDNESS**

**k-C-boundedness** (parametrized by a chase variant C):
Given a set of rules \( \mathcal{R} \) and an integer \( k \), is \( \mathcal{R} \) C-bounded with bound \( k \)?

Results so far

- decidability and upper complexity bounds for (semi-)oblivious and restricted chases [Delivorias & al., Rule-ML+RR 2018]

- complexity for datalog co-NExptime-complete [Bourhis & al. IJCAI 2019]

→ complexity for specific existential rules classes?

→ is k-boundedness decidable for the core chase?
Conclusions

- **Existential rules** are an expressive and flexible formalism particularly well suited to OBDA.

- **Chase termination** and **FO-rewritability** are essential properties for query answering.

- **Boundedness / k-Boundedness** are key properties for optimizations:
  - Many open questions from a theoretical viewpoint
  - The exploitation of these properties in practice is just beginning