Reasoning on Data

with Existential Rules

Marie-Laure Mugnier

University of Montpellier
PART OF A KNOWLEDGE GRAPH

∃ x (Prof(Bob) ∧ PHS(#1) ∧ Comp(C) ∧ Pest(b) ∧ involvedIn(Bob,#1) ∧ fundedBy(Bob,C) ∧ about(#1,P) ∧ produces(C,b) ∧ contains(b,P))

Facts

Rules

∀ x (PHS(x) \rightarrow PIS(x))

∀ x ∀ y (fundedBy(x,y) \rightarrow relatedTo(x,y))

+ Basic ontological knowledge

PublicHealthStudy subclass of PublicInterestStudy

fundedBy subproperty of relatedTo

Marie-Laure Mugnier – Reasoning on Data with Existential Rules
**How to Infer Conflicts of Interest (CoI)?**

**Query:** “Find all \(x, y, z\) such that \(x\) has a conflict for study \(y\) because of its relationships with company \(z\)”
DEFINING CONFLICTS OF INTEREST

\[ R_1: \forall x \forall y \forall z \ (\text{produces}(x,y) \land \text{contains}(y,z)) \rightarrow \text{hasInterest}(x,z) \]

\[ R_2: \forall x \forall y \forall z \forall u \ (\text{involvedIn}(x,y) \land \text{PIS}(y) \land \text{about}(y,u) \land \text{relatedTo}(x,z) \land \text{Company}(z) \land \text{hasInterest}(z,u)) \rightarrow \text{Col}(x,y,z) \]

What if we only have unary and binary predicates ie graphs and not hypergraphs?

Reification: new object of type Col

\[ R_2: \forall x \forall y \forall z \forall u \ (\text{body}[x,y,z,u] \rightarrow \exists o \ (\text{Col}(o) \land \text{in}(x,o) \land \text{on}(o,y) \land \text{with}(o,z))) \]

Interest of creating a new object:

- **Flexible** description of Col (different patterns) instead of a fixed arity predicate
- Ability to talk about Col

Marie-Laure Mugnier – Reasoning on Data with Existential Rules
INFERRING CONFLICTS OF INTEREST

**Facts**

\[
\exists x \ (\text{Prof}(Bob), \text{PHS}(#1), \text{Comp}(C), \text{Pest}(x) \land \text{involvedIn}(Bob,#1), \text{fundedBy}(Bob,C) \land \text{about} (#1,P) \land \text{produces}(C,x) \land \text{contains}(x,P))
\]

**Rules** (universal quantifiers omitted)

\[
\text{PHS}(x) \rightarrow \text{PIS}(x) \\
\text{fundedBy}(x,y) \rightarrow \text{relatedTo}(x,y)
\]

\[
R_1: \text{produces}(x,y) \land \text{contains}(y,z) \\
\rightarrow \text{hasInterest}(x,z)
\]

\[
R_2: \text{involvedIn}(x,y) \land \text{PIS}(y) \land \text{about}(y,u) \land \text{relatedTo}(x,z) \land \text{Company}(z) \land \text{hasInterest}(z,u) \\
\rightarrow \exists o \ \text{CoI}(o) \land \text{in}(x,o) \land \text{on}(o,y) \land \text{with}(o,z)
\]

**Inferred facts**

\[
\text{PIS}(#1), \text{relatedTo}(Bob,C), \text{hasInterest}(C,P) \\
\text{Col}(o_1), \text{in}(Bob,o_1), \text{on}(o_1,#1), \text{with}(o_1,C)
\]

**Query:** find \((x,y,z)\) such that

\[
\exists o \ \text{CoI}(o) \land \text{in}(x,o) \land \text{on}(o,y) \land \text{with}(o,z)
\]

**Answer:** \((Bob,#1,C)\)
**EXISTENTIAL RULES AS AN ONTOLOGICAL LANGUAGE**

∀X. ∀Y. Body [X,Y] \(\rightarrow\) \(\exists\) Z. Head [X,Z]

**X, Y, Z : sets of variables**

any **positive conjunction** (without functional symbols except constants)

∀x. ∀y. siblingOf(x,y) \(\rightarrow\) \(\exists\) z parentOf(z,x) \(\land\) parentOf(z,y)

*(Universal quantifiers will be omitted in examples)*

Key point: ability to assert **the existence of unknown entities**

Crucial for representing ontological knowledge in **open domains**
**Rule Application**

*R is applicable to* $F$ *if there is a homomorphism* $h$ *from* $\text{body}(R)$ *to* $F$

*ie* a substitution $h$ of the variables in $\text{body}(R)$ by terms in $F$ such that $h(\text{body}(R)) \subseteq F$

$$R = \text{siblingOf}(x,y) \rightarrow \exists z \text{ parentOf}(z,x) \land \text{parentOf}(z,y)$$

$$F = \{ \text{siblingOf}(a,b) \}$$

$h$: $x \mapsto a$

$y \mapsto b$

The application produces $h(\text{head}(R))$

where a fresh variable (a null) is created for each existential variable in $R$

$$F' = \{ \text{siblingOf}(a,b), \text{parentOf}(z_0,a), \text{parentOf}(z_0,b) \}$$
THEORETICAL FOUNDATIONS

Graph-based KR
[Chein Mugnier 1992, 2009]

Datalog (70-80s)

logical translation of graph rules

∀ ∃-rules, existential Rules [Baget+ IJCAI 2009]

Datalog+/- family [Cali+ PODS 2009]

+ existential variables in rule heads

+ complex relationships between objects
  + unbounded predicate arity

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

• Same logical form as « Tuple-Generating Dependencies » (TGDs)
  long studied in database theory
**QUERY ANSWERING PROBLEM**

**Conjunctive query (CQ)**

- SELECT ... FROM ... WHERE <join conditions>
- SELECT ... WHERE <basic graph pattern>

Existentially quantified conjunction of atoms, where free variables are answer variables

\[ \exists o (\text{Col}(o) \land \text{in}(x,o) \land \text{on}(o,y) \land \text{with}(o,z)) \]

**Query answering decision problem:**

Given a KB $K$ and CQ $q$, does $K$ provide an answer to $q$?

Undecidable

but many decidable and tractable classes are known

---

Marie-Laure Mugnier – Reasoning on Data with Existential Rules
1. **FORWARD CHAINING (CHASE) - MATERIALISATION**

For any CQ $q$, $\text{Answers}(q,K) = \text{Answers}(q, \text{chase}(K))$

Of course $\text{chase}(K)$ may be infinite

*e.g.* $\text{Human}(x) \rightarrow \exists z \ \text{parentOf}(z,x) \land \text{Human}(z)$ with $F = \text{Human}(Bob)$
2. BACKWARD CHAINING BY QUERY REWRITING

1. $q$ is rewritten into a union of CQs, and more generally into a first-order query (core SQL query)

2. The rewriting is evaluated on the facts

Query rewriting is independent from any set of facts
For any $F$, $\text{Answers}(q,(F, R)) = \text{Answers}(\text{rew}(q, R), F)$

Of course, there may be no finite $\text{rew}(q, R)$
Finite chase

Finite query rewriting

Finite chase

Finite query rewriting

Greedy Bounded Treewidth

- glut-fg
- jointly-fg
- weakly-frontier-guarded
- warded
- weakly-guarded
- frontier-guarded
- guarded
- frontier-1

RDFS

DL-Lite

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic

Datalog

MFA

jointly-acyclic

weakly-acyclic
Finite chase

Finite query rewriting

Greedy Bounded Treewidth

Datalog

DL-Lite

RDFS

POLYNOMIAL DATA COMPLEXITY

Marie-Laure Mugnier
ONTOMETRY-BASED DATA ACCESS (OBDA)

Query using the ontology vocabulary

Description of the application domain with a high abstraction level

Materialized or virtual Facts using the ontology vocabulary

Mappings from data to facts

{ Database query ~ Facts }

Independent and heterogeneous data sources
OBDA with Virtual Facts

1. Rewrite $q$ with the ontology

2. Rewrite $Q_o$ with the mappings

3. Evaluate $Q_{o,M}$ with a mediator query engine
**Existential Rules as a Uniform Language for OBDA**

**Ontology**
- Query
- Facts

**Mappings from data to facts**
- «Global-Local-as-View» mappings: able to create new objects, which increases the data integration power
CONCLUSION

- Existential rules are able to express **complex structures** and create **new objects**

- These features can be exploited for both expressing **ontological knowledge** and **integrating data**

- A wide range of rule classes offer various **expressivity/complexity** tradeoffs

- The framework has been **extended** in several ways:
  - other rules: negative constraints and Equality Generating Dependencies
  - existential rules extended to stratified negation and disjunctive heads

- Efficient **systems** are available
SOME SYSTEMS

Query answering with existential rules


**RDFox**  fast chase-based engine  [https://www.cs.ox.ac.uk/isg/tools/RDFox/](https://www.cs.ox.ac.uk/isg/tools/RDFox/)

**Vadalog**  fast chase-based engine for warded Datalog+/-  [Commercial]

**Graal**  toolkit with fast query rewriting, rule set analysis, chase algorithms, ...  [http://graphik-team.github.io/graal/](http://graphik-team.github.io/graal/)

and many other usable tools developed for other purposes:

**Llunatic**  data exchange and data cleaning (chase on tuple-generating dependencies)

**DLV**  Answer Set Programming (DLV³: for shy existential rules)  [Commercial]

...

**OBDA systems**  are still restricted to lightweight description logics or (extended) RDFS: OnTop, Mastro, UltraWrap[OBDA] ...
**FURTHER READING ON THE EXISTENTIAL RULE FRAMEWORK**

**Introductions and surveys**


**Seminal papers (journal versions)**


**Latest results**

See conferences in AI (IJCAI, KR), databases (PODS, ICDT, VLDB) and the semantic web (ISWC)