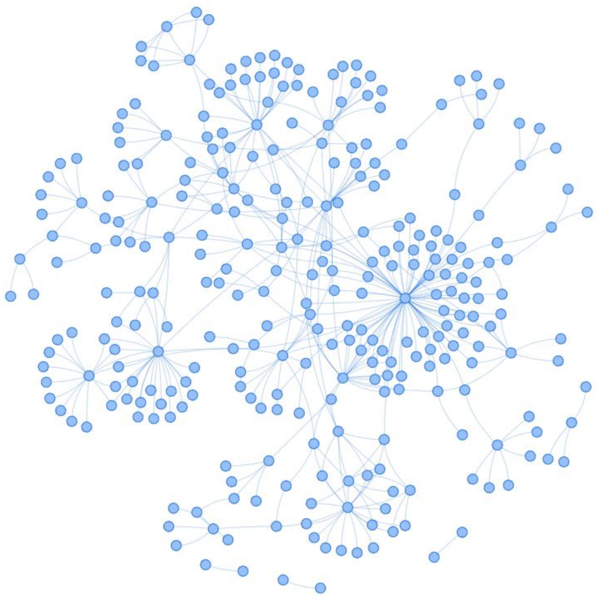
Explainable AI: Rule-aware Datalog Fact Explanation Using Group-SAT Solver

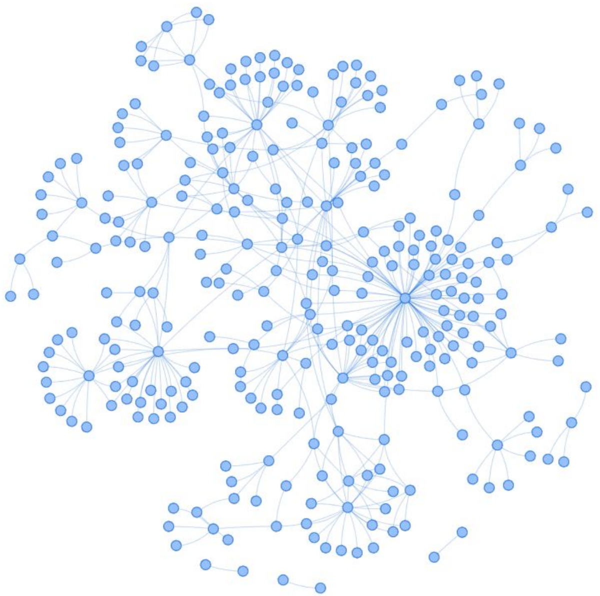
Akira Charoensit, David Carral, Pierre Bisquert, Lucas Rouquette, Federico Ulliana
Inria, LIRMM, Univ Montpellier, CNRS, France
IATE, Univ Montpellier, INRAE, Institut Agro, Montpellier, France

Introduction: making sense of data



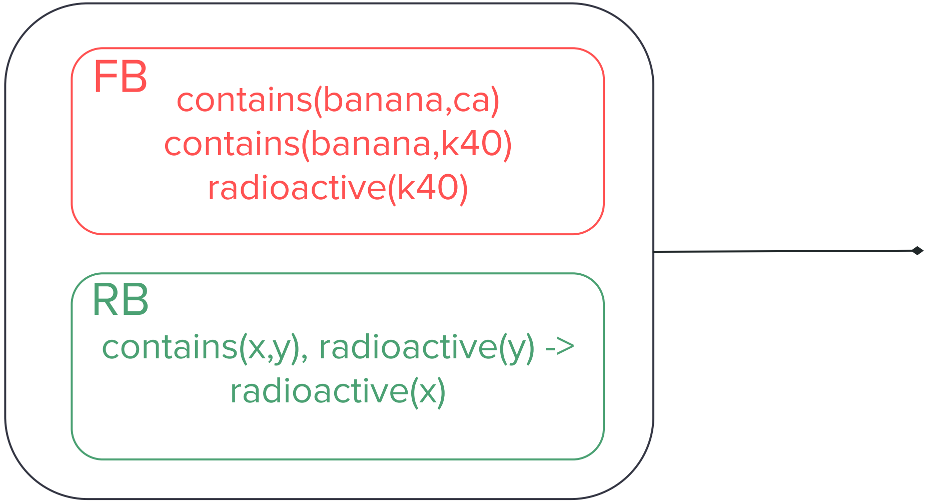
? radioactive

Introduction: making sense of data



Bananas
are
radioactive!

Introduction: making sense of data



Bananas are radioactive!

Introduction: making sense of data

FB

contains(banana,ca)
contains(banana,k40)
radioactive(k40)

RB

contains(x,y), radioactive(y) -> radioactive(x)

Bananas are radioactive!

Contribution

1. Explanations for Datalog

2. Using a group-MUS solver

3. Optimisation through filtering of relevant facts

Research Questions

1. **What is a useful form of explanations?**

2. **What is an efficient way to compute them?**

Research Questions

1. **What is a useful form of explanations?**

   -> KB-support Explanation:

   An explanation of a fact *p* with regard to KB *K* is a ***minimal*** subset of *K* that entails *p*.

Research Questions

1. **What is a useful form of explanations?**

   -> KB-support explanation:

   An explanation of a fact *p* with regard to KB *K* is a ***minimal*** subset of KB that entails *p*.

2. **What is an efficient way to compute KB-support explanations?**

   -> Hint: Explanations resemble MUS in SAT formula

Computing explanations: MUS

MUS: minimal unsatisfiable subsets of clauses

$p \wedge (p\text{->}q) \wedge (\text{NOT } q) \wedge s \wedge (q \text{ v NOT } s) \wedge t$

Computing explanations: MUS

MUS: minimal unsatisfiable subsets of clauses

$\boxed{p \wedge (p\text{->}q) \wedge \boxed{(\text{NOT } q)}} \wedge s \wedge (q \vee \text{NOT } s) \wedge t$

Research Questions

1. **What is a useful form of explanations?**

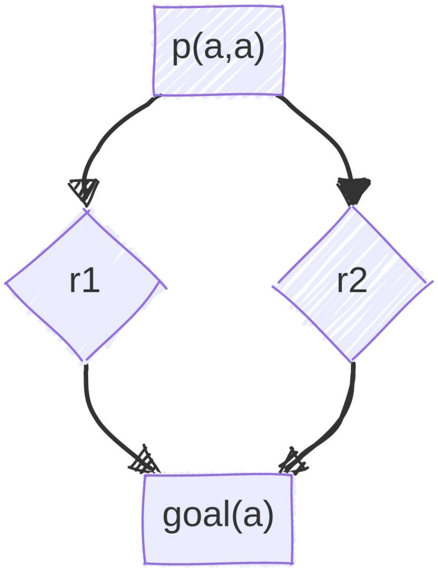
   -> A KB-support explanation:

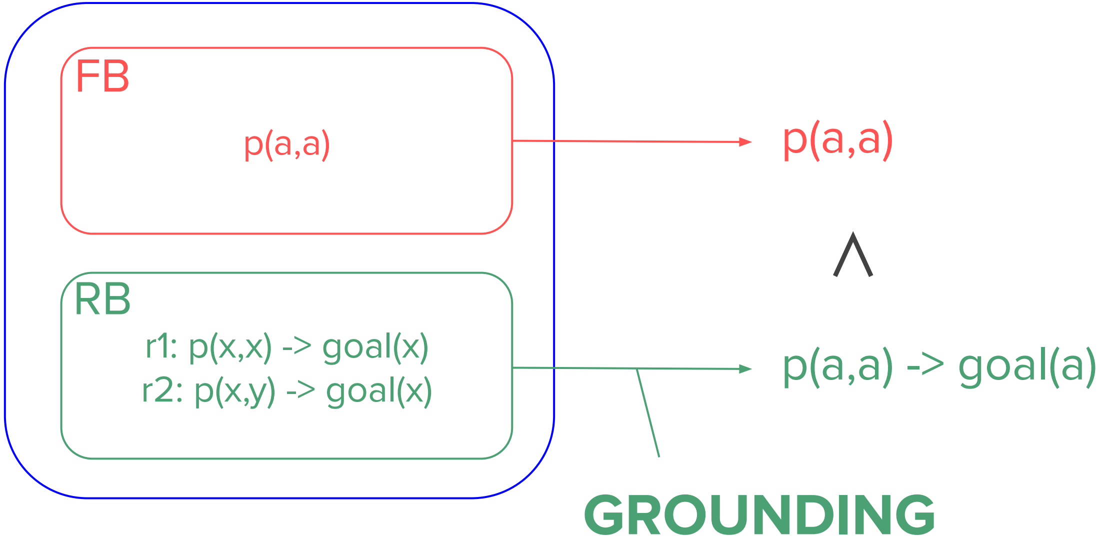
   An explanation of a fact *p* is a ***minimal*** subset of KB that entails *p*.

2. **What is an efficient way to compute explanations?**

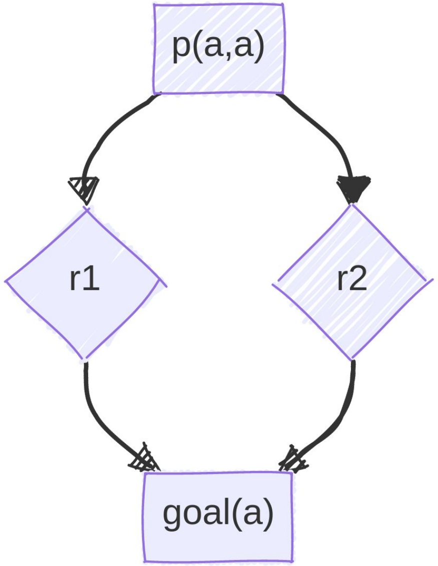
   -> Hint: Explanations resemble MUS in SAT formula

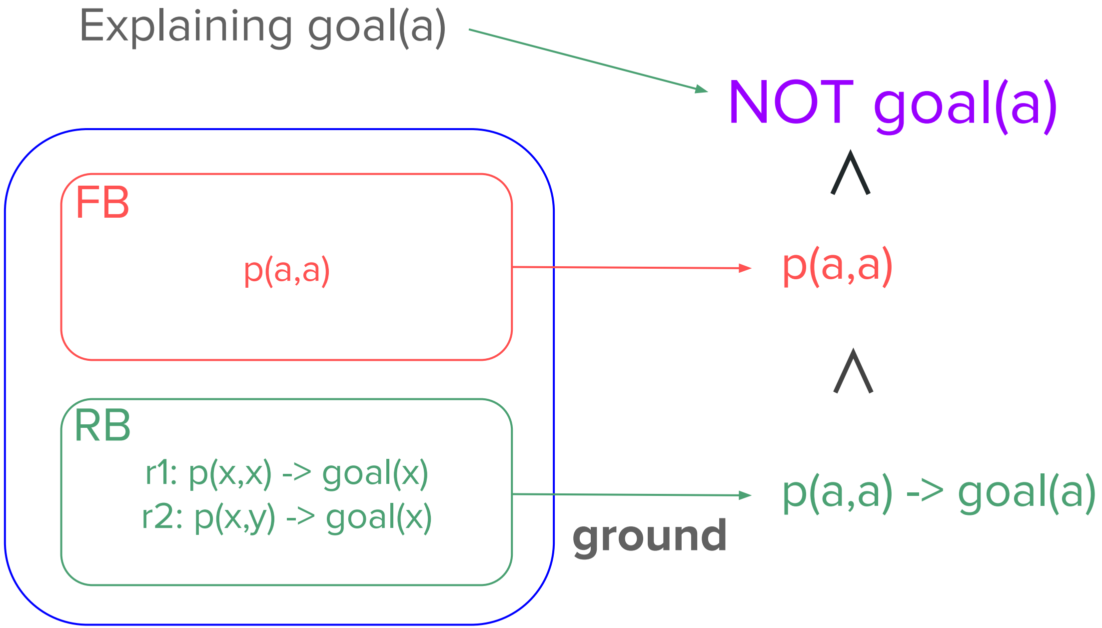
Building a SAT formula from a KB





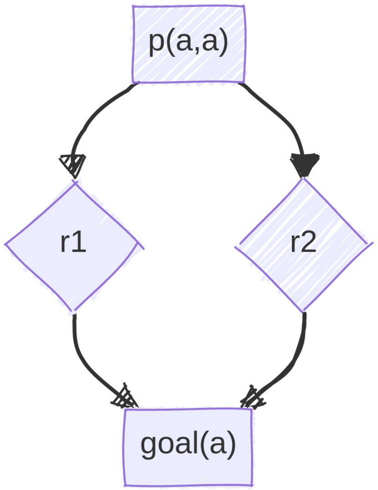
Finally: reducing fact explanation to MUS





Computing explanations: reduction to (standard) MUS

r1: p(x,x) -> goal(x)
r2: p(x,y) -> goal(x)



Example 1

Encoding

NOT goal(a)
p(a,a)
p(a,a)->goal(a)

MUS Solver

MUS

NOT goal(a)
p(a,a)
p(a,a)->goal(a)

Issue with standard MUS:
1 MUS for 2 explanations

Computing explanations: group MUS

Group MUS: minimal unsatisfiable subsets of *groups of* clauses

0		1		2		3
[NOT g(a)]	∧	[p(a,a)]	∧	[p(a,a)->g(a)]	∧	[p(a,a)->g(a)]

group-MUS Solver

Group MUS:
2 MUS for 2 explanations

0, 1, 2 => FB: p(a,a) RB: r1

0, 1, 3 => FB: p(a,a) RB: r2

Computing explanations: group MUS

Group MUS: minimal unsatisfiable subsets of *groups of* clauses

0 1 2 3

[NOT g(a)] ∧ [p(a,a)] ∧ [p(a,a)->g(a)] ∧ [p(a,a)->g(a)]

group-MUS Solver

Group MUS:
2 MUS for 2 explanations

0, 1, 2 => FB: p(a,a) RB: r1

0, 1, 3 => FB: p(a,a) RB: r2

KB-support explanations = group-MUSes

Issue: MUS solving is expensive

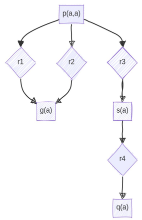
1. KB can be very large
   - so is the group-SAT formula: how to reduce it?

2. The complexity of group-MUS enumeration

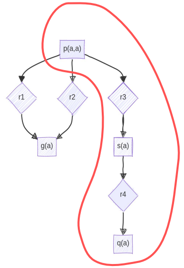
Filtering: Find the relevant subset of the KB

2 steps:

1. Static step
   - entailment graph building



2. Dynamic step
   - fact relevance tracing

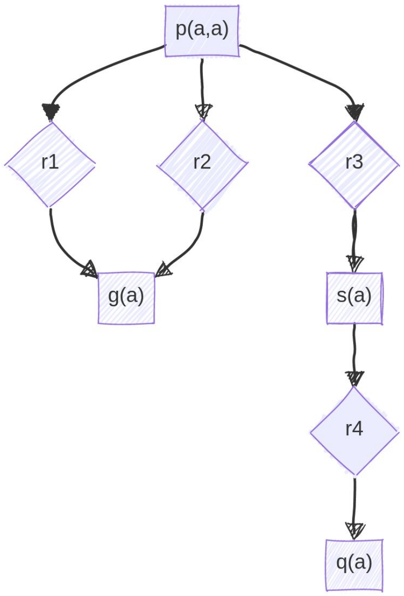


Static Step: Entailment Graph Building

Desiderata: a data structure to trace the lineage of an atom

Static Step: Entailment Graph Building

Desiderata: a data structure to trace the lineage of an atom



Static Step: Entailment Graph Building

Desiderata: a data structure to trace the lineage of an atom

Solution: use a rule-based encoding to build this data structure

FB

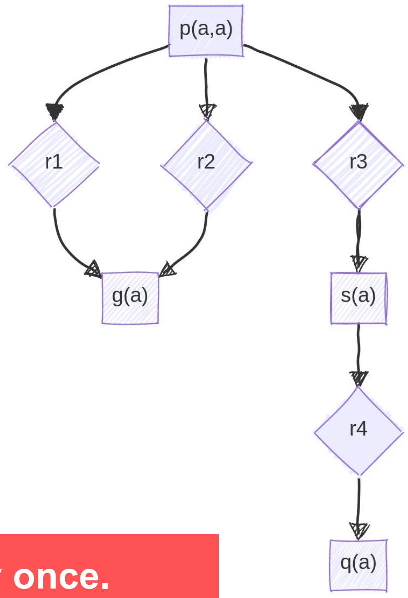
p(a,a)

RB

r1: p(x,x) -> g(x)
r2: p(x,y) -> g(x)
r3: p(x,x) -> s(x)
r4: s(x) -> q(x)

Datalog programme

```
#FB
E_r1(f_p(a,a),f_g(a))
E_r2(f_p(a,a),f_g(a))
E_r3(f_p(a,a),f_s(a))
E_r4(f_s(a),f_q(a))

#RB
[REL(Y) ^ E_r1(X1,Y) ->
REL(X1) ^ REL_E_r1(X1,Y),
…
REL(Y) ^ E_r4(X1,Y) ->
REL(X1) ^ REL_E_r4(X1,Y)]
```



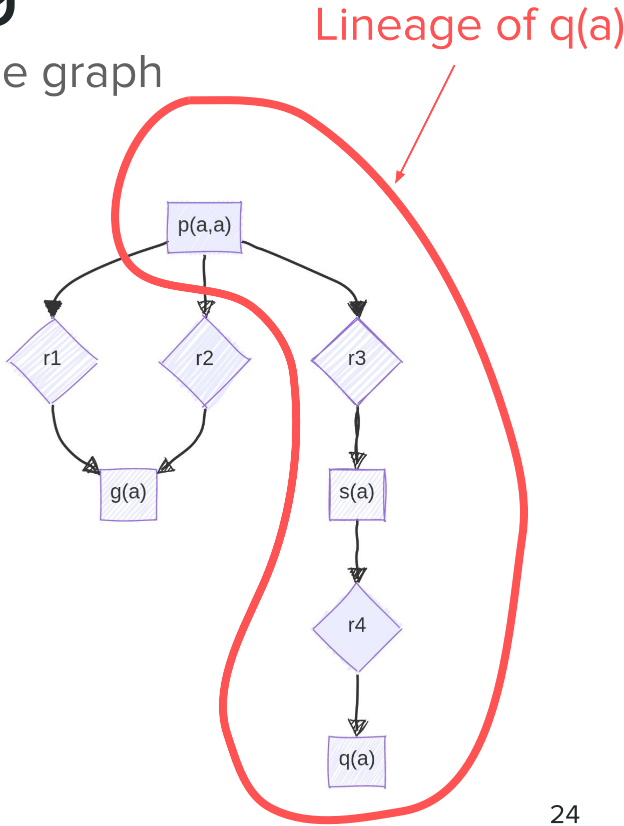
This step is computed only once.

Dynamic Step: Fact Tracing

Desiderata: trace the FB-ancestors of the query atom on the graph

Dynamic Step: Fact Tracing

Desiderata: trace the FB-ancestors of the query atom on the graph



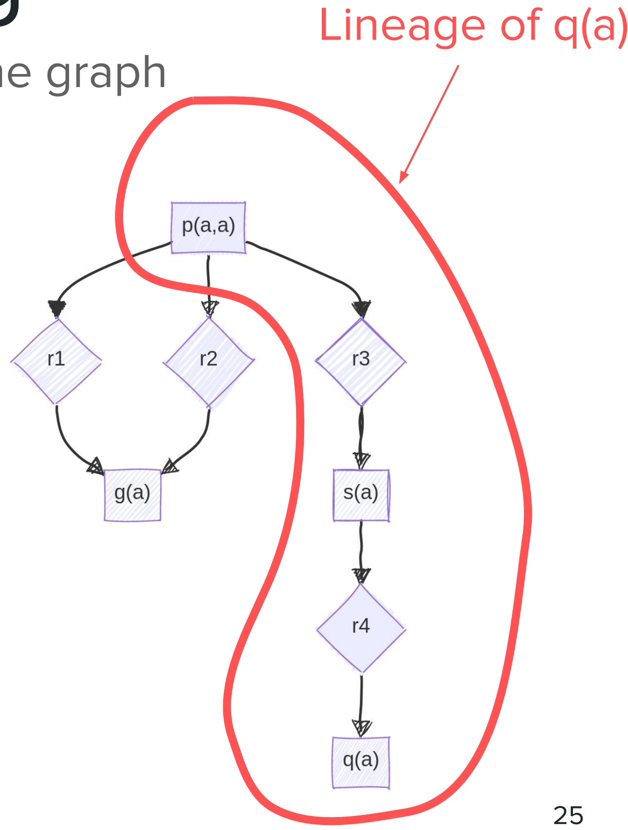
Dynamic Step: Fact Tracing

Desiderata: trace the FB-ancestors of the query atom on the graph

Solution: rule-based encoding of the tracing

This step is computed once for every query.

```
#FB
E_r1(f_p(a,a),f_g(a))
E_r2(f_p(a,a),f_g(a))
E_r3(f_p(a,a),f_s(a))
E_r4(f_s(a),f_q(a))
REL(f_q(a))

#RB
[REL(Y) ^ E_r1(X1,Y) ->
REL(X1) ^ REL_E_r1(X1,Y),
...
REL(Y) ^ E_r4(X1,Y) ->
REL(X1) ^ REL_E_r4(X1,Y)]
```



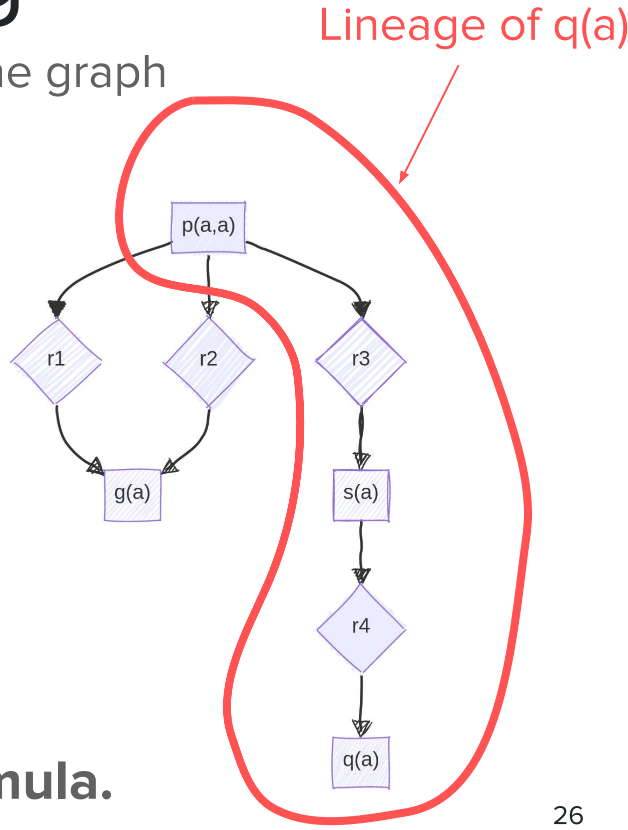
Dynamic Step: Fact Tracing

Desiderata: trace the FB-ancestors of the query atom on the graph

Solution: rule-based encoding of the tracing

This step is computed once for every query.

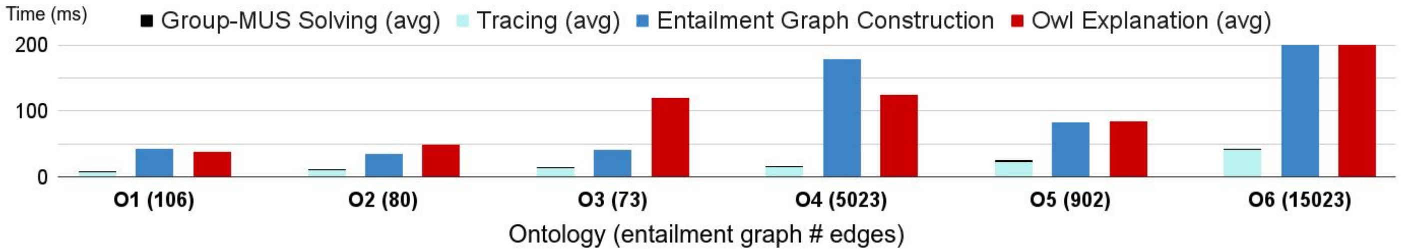
```
#FB
E r1(f p(a,a),f g(a))
E r2(f p(a,a),f g(a))
E r3(f p(a,a),f s(a))
E r4(f s(a),f q(a))
REL(f q(a))

#RB
[REL(Y) ^ E r1(X1,Y) ->
REL(X1) ^ REL E r1(X1,Y)
...
REL(Y) ^ E r4(X1,Y) ->
REL(X1) ^ REL E r4(X1,Y)
```



Filtering reduces the size of the group-SAT formula.

Experiment

Preprocessing:
- 24 ontologies from MOWL corpus translated to Datalog (via Carral et al.)
- Choose 5 facts from deepest reasoning level
- Run against OWL Explanation (owl api) tool



Conclusion: our approach is generally more competitive as the number of queries grows (and sometimes even for one query!)

Related works

Datalog:

- *Calautti et al., 2024*: SAT-solvers for computing why-provenance in Datalog.
- *Elhalawati et al., 2022*: On-demand computation of Datalog provenance using rule-based approaches.

Description Logic:

TBoxes (without data):

- *Baader et al., 2007*: Axiom pinpointing for EL TBoxes.
- *Manthey et al., 2020*: SAT-based axiom pinpointing for lightweight Description Logics.

Provenance in EL Description Logic:

- *Borgwardt et al., 2023*: ABox justifications via Datalog rewriting.

Answer Set Programming:

- *Eiter & Geibinger, 2023*: Explaining answer-set programs with abstract constraint atoms.
- *Alviano et al., 2023*: Graph-explanations extending fact-support in ASP.

Conclusion

1. Study of **KB-support** explanations for datalog
2. Reduction of explanation to **group-MUS** problem
3. Filtering of **relevant** facts and rules exploiting datalog programme
4. Implementation & Evaluation

Reference

Abiteboul, S., Hull, R., Vianu, V.: Foundations of databases. Add. Wesley (1995)

Baget, J.F., Bisquert, P., Leclère, M., Mugnier, M.L., Pérution-Kihli, G., Tornil, F., Ulliana, F.: InteGraal: a Tool for Data-Integration and Reasoning on Heterogeneous and Federated Sources. In: BDA (2023)

Beeri, C., Vardi, M.Y.: A proof procedure for data dependencies. J. ACM (1984)

Bourgaux, C., Bourhis, P., Peterfreund, L., Thomazo, M.: Revisiting semiring provenance for datalog. In: KR (2022)

Calautti, M., Livshits, E., Pieris, A., Schneider, M.: Computing the why-provenance for datalog queries via SAT solvers. In: AAAI. pp. 10459–10466 (2024)

Carral, D., Zalewski, J., Hitzler, P.: An efficient algorithm for reasoning over OWL EL ontologies with nominal schemas. J. Log. Comput 33(1), 136–162 (2022)

Elhalawati, A., Krötzsch, M., Mennicke, S.: An existential rule framework for computing why-provenance on-demand. In: RuleML+RR. pp. 146–163 (2022)

Horridge, M., Bechhofer, S.: The OWL Api. Semantic Web (2011)

Liffiton, M.H., Previti, A., Malik, A., Marques-Silva, J.: Fast, flexible mus enumeration. Constraints 21(2), 223–250 (2016)

Liffiton, M.H., Sakallah, K.A.: Algorithms for computing minimal unsatisfiable subsets of constraints. Journal of Automated Reasoning 40, 1–33 (2008)

Xiao, G., Calvanese, D., Kontchakov, R., Lembo, D., Poggi, A., Rosati, R., Zakharyaschev, M.: Ontology-based data access: A survey. In: IJCAI (2018)

Zhapa-Camacho, F., Kulmanov, M., Hoehndorf, R.: mOWL: Python library for machine learning with biomedical ontologies. Bioinformatics (2022)