Bridging a gap between static analysis and ontology-based reasoning

Filip Murlak
University of Warsaw

Workshop on Foundations of Databases and AI 2025 @ TU Wien

Databases

Data	(tables, trees, graphs) – relational structures/models
Queries	(SQL, XPath, Cypher) – formulas with free variables
Metadata	(schemas, integrity constraints) – theories
Goal	Evaluate queries over data, guided by metadata.

Databases

Data (tables, trees, graphs) – relational structures/models
Queries (SQL, XPath, Cypher) – formulas with free variables
Metadata (schemas, integrity constraints) – theories

Goal Evaluate queries over data, guided by metadata.

Problem (Query containment modulo schema)

Given queries P and Q, and a schema $\mathcal{S}$, decide if $P \subseteq_{\mathcal{S}} Q$; that is,

for every database D, if $D \models \mathcal{S}$ and $D \models P$, then $D \models Q$.

Knowledge representation

Facts (ABoxes) - ground atomic formulas (a CQ with no variables)

Ontologies (TBoxes, rules) - theories

Queries (SPARQL) - formulas with free variables

Goal Answer queries based on facts derived using ontologies.

Knowledge representation

Facts (ABoxes) - ground atomic formulas (a CQ with no variables)
Ontologies (TBoxes, rules) - theories
Queries (SPARQL) - formulas with free variables

Goal Answer queries based on facts derived using ontologies.

Problem (Query entailment)

Given facts $\mathcal{A}$, an ontology $\mathcal{T}$, and a query Q, decide if $\mathcal{T}, \mathcal{A} \models Q$; that is,

for every interpretation I, if $I \models \mathcal{T}$ and $I \models \mathcal{A}$, then $I \models Q$.

Two problems, or one?

Problem (Query containment modulo schema)
Given queries P *and* Q, *and a schema* $\mathcal{S}$, *decide if* $P \subseteq_{\mathcal{S}} Q$; *that is,*

for every database D, *if* $D \models \mathcal{S}$ *and* $D \models P$, *then* $D \models Q$.

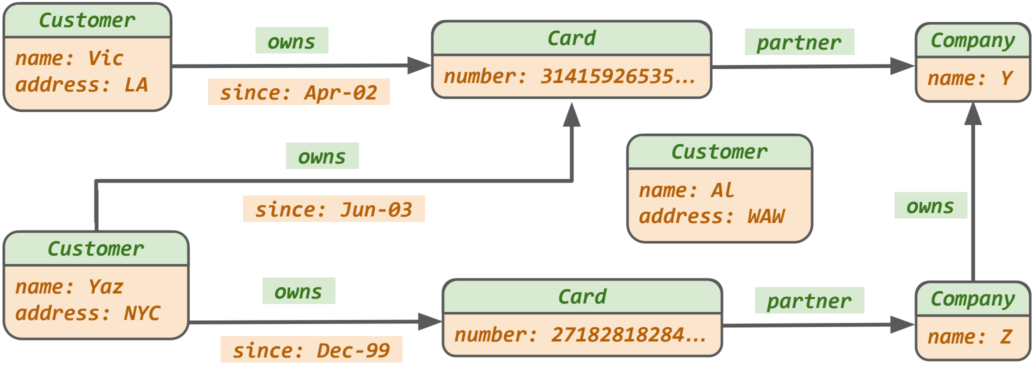
Problem (Query entailment)
Given facts $\mathcal{A}$, *an ontology* $\mathcal{T}$, *and a query* Q, *decide if* $\mathcal{T}, \mathcal{A} \models Q$; *that is,*

for every interpretation I, *if* $I \models \mathcal{T}$ *and* $I \models \mathcal{A}$, *then* $I \models Q$.

$$D \approx I \qquad P \approx \mathcal{A} \qquad \mathcal{S} \approx \mathcal{T}$$

[Calvanese, De Giacomo, Lenzerini '98]

Graphs



Customer
name: Vic
address: LA
owns
since: Apr-02
Card
number: 31415926535...
partner
Company
name: Y
owns
since: Jun-03
Customer
name: Al
address: WAW
owns
Customer
name: Yaz
address: NYC
owns
since: Dec-99
Card
number: 27182818284...
partner
Company
name: Z

Graphs

We work with labelled graphs, modelled as relational structures:

- unary predicates = node labels = concept names $A, B, \ldots$
- binary predicates = egde labels = role names $r, s, \ldots$

That is,

- nodes have multiple labels;
- edges have single labels;
- parallel edges with different labels are allowed;
- in a subgraph, nodes may have fewer labels.

Queries

- Conjunctive queries (CQs), unions of CQs (UCQs)

$$\exists x\, \exists y\; A(x) \wedge r(x,y) \wedge \bar{A}(y) \quad \vee \quad \exists x\, \exists y\, \exists z\; r(x,y) \wedge r(y,z) \wedge r(z,x)$$

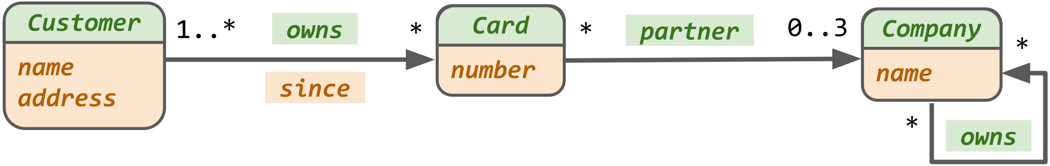
  The core of relational query languages, such as SQL.

- Conjunctive regular path queries (CRPQs), unions of CRPQs (UCRPQs)

$$\exists x\; r^{+}(x,x) \quad \vee \quad \exists x\, \exists y\; A(x) \wedge (r^{*} \cup s)(x,y) \wedge (p \cdot B)^{*}(y,x)$$

  Graph query languages have reachability/RPQs at the core (SPARQL, Neo4j's Cypher, SQL/PGQ, GQL).

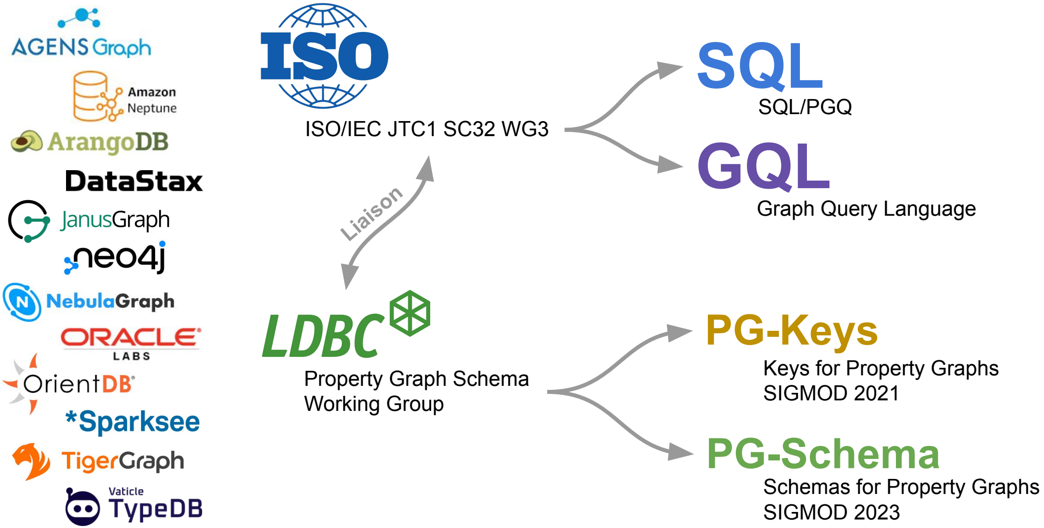
Schemas



Customer
name
address
1..*
owns
since
*
Card
number
*
partner
0..3
Company
name
*
*
owns

What kind of schemas for graph data?

RDF has SHACL and ShEx, for property graphs the picture is less clear.



Common Foundations for SHACL, ShEx, and PG-Schema

Shqiponja Ahmetaj
shqiponja.ahmetaj@tuwien.ac.at
TU Wien
Vienna, Austria

Iovka Boneva
iovka.boneva@univ-lille.fr
Univ. Lille, CNRS, Inria, Centrale Lille, UMR 9189 CRIStAL
Lille, France

Jan Hidders
j.hidders@bbk.ac.uk
Birkbeck, University of London
London, UK

Katja Hose
katja.hose@tuwien.ac.at
TU Wien
Vienna, Austria

Maxime Jakubowski
maxime.jakubowski@tuwien.ac.at
TU Wien
Vienna, Austria

Jose Emilio Labra Gayo
labra@uniovi.es
University of Oviedo
Oviedo, Spain

Wim Martens
wim.martens@uni-bayreuth.de
University of Bayreuth
Bayreuth, Germany

Fabio Mogavero
fabio.mogavero@unina.it
Università di Napoli Federico II
Naples, Italy

Filip Murlak
f.murlak@uw.edu.pl
University of Warsaw
Warsaw, Poland

Cem Okulmus
okulmus@mail.uni-paderborn.de
Paderborn University
Paderborn, Germany

Axel Polleres
axel.polleres@wu.ac.at
WU Wien
Vienna, Austria
CSH Vienna
Vienna, Austria

Ognjen Savković
ognjen.savkovic@unibz.it
Free University of Bolzano
Bolzano, Italy

Mantas Šimkus
mantas.simkus@tuwien.ac.at
TU Wien
Vienna, Austria

Dominik Tomaszuk
d.tomaszuk@uwb.edu.pl
TU Wien
Vienna, Austria
University of Bialystok
Bialystok, Poland

ABSTRACT

Graphs have emerged as an important foundation for a variety of applications, including capturing and reasoning over factual knowledge, semantic data integration, social networks, and providing factual knowledge for machine learning algorithms. To formalise certain properties of the data and to ensure data quality, there is a need to describe the *schema* of such graphs. Because of the breadth of applications and availability of different data models, such as RDF and property graphs, both the Semantic Web and the database community have independently developed *graph schema languages*: SHACL, ShEx, and PG-Schema. Each language has its unique approach to defining constraints and validating graph data, leaving potential users in the dark about their commonalities and differences. In this paper, we provide formal, concise definitions of the *core components* of each of these schema languages. We employ a uniform framework to facilitate a comprehensive comparison between the languages and identify a common set of functionalities, shedding light on both overlapping and distinctive features of the three languages.

ACM Reference Format:
Shqiponja Ahmetaj, Iovka Boneva, Jan Hidders, Katja Hose, Maxime Jakubowski, Jose Emilio Labra Gayo, Wim Martens, Fabio Mogavero, Filip Murlak, Cem Okulmus, Axel Polleres, Ognjen Savković, Mantas Šimkus, and Dominik Tomaszuk. 2025. Common Foundations for SHACL, ShEx, and PG-Schema. In *Proceedings of the ACM Web Conference 2025 (WWW '25), April 28–May 2, 2025, Sydney, NSW, Australia.* ACM, New York, NY, USA, 23 pages. https://doi.org/10.1145/XXXXXX.XXXXXX

WWW '25, April 28–May 2, 2025, Sydney, NSW, Australia

ACM ISBN 979-8-4007-1274-6/25/04
https://doi.org/10.1145/XXXXXX.XXXXXX

1 INTRODUCTION

Driven by the unprecedented growth of interconnected data, *graph-based data representations* have emerged as an expressive and versatile framework for modelling and analysing connections in data sets [46]. This rapid growth however, has led to a proliferation of diverse approaches, each with its own identity and perspective.

Ahmetaj, Boneva, Hidders, Hose, Jakubowski, Labra Gayo, Martens, Mogavero, Murlak, Okulmus, Polleres, Savković, Šimkus, Tomaszuk,

Common Foundations for SHACL, ShEx, and PG-Schema,

WWW 2025 (to appear).

Description logics

Basic description logic $\mathcal{ALC}$ has statements of the form

$$C \sqsubseteq D$$

where C, D are *complex concepts* build according to the following grammar by

$$C, D ::= \bot \mid \top \mid A \mid C \sqcup D \mid C \sqcap D \mid \neg C \mid \exists r.C \mid \forall r.C\,.$$

For example, $\mathsf{Person} \sqsubseteq \exists\, \mathsf{childOf}.\, \mathsf{Person}$ and $\mathsf{Male} \sqcap \exists\, \mathsf{childOf}.\, \mathsf{Person} \sqsubseteq \mathsf{Son}$.

$\mathcal{ALC}$ in normal form:

$$\begin{aligned} A_1 \sqcap A_2 \sqcap \cdots \sqcap A_n &\sqsubseteq B_1 \sqcup B_2 \sqcup \cdots \sqcup B_m \\ A &\sqsubseteq \exists r.B \\ A &\sqsubseteq \forall r.B \end{aligned}$$

empty $\sqcap = \top$

empty $\sqcup = \bot$

A zoo of description logics

Features can be added to $\mathcal{ALC}$, giving logics like $\mathcal{ALCHOIQ}$ or $\mathcal{SOQ}$ ($\mathcal{S}$ is shorthand for $\mathcal{ALCS}$).

$\mathcal{O}$: use constants as singleton concepts $\{a\}$	$A \sqsubseteq \exists r.\{a\}$
$\mathcal{I}$: use inverse roles r^- anywhere	$A \sqsubseteq \exists r^-.B$
$\mathcal{F}$: declare role r to be a partial function	$\mathsf{fun}(r)$
$\mathcal{Q}$: use counting quantifiers $\exists^{\leq n}$, $\exists^{\geq n}$	$A \sqsubseteq \exists^{\leq 5} r.B$
$\mathcal{S}$: declare role r to be transitive	$\mathsf{tra}(r)$
$\mathcal{H}$: declare role r to be contained in role s	$r \sqsubseteq s$

$\mathcal{ALCQI}$ can express EER, and a lot of SHACL and PG-Schema.

Finite vs unrestricted models

Problem (Query containment modulo schema)
Given queries P and Q, and a schema $\mathcal{S}$, decide if $P \subseteq_{\mathcal{S}} Q$; that is,

for every finite database D, if $D \models \mathcal{S}$ and $D \models P$, then $D \models Q$.

Problem (Query entailment)
Given facts $\mathcal{A}$, an ontology $\mathcal{T}$, and a query Q, decide if $\mathcal{T}, \mathcal{A} \models Q$; that is,

for every possibly infinite interpretation I, if $I \models \mathcal{T}$ and $I \models \mathcal{A}$, then $I \models Q$.

A paradox

Does $\mathcal{T} = \{\mathsf{Person} \sqsubseteq \exists\, \mathsf{childOf}.\, \mathsf{Person}\}$ model reality well?

A paradox

Does $\mathcal{T} = \{\mathsf{Person} \sqsubseteq \exists\, \mathsf{childOf}.\, \mathsf{Person}\}$ model reality well?

Suppose we know that at least one person exists: $\mathcal{A} = \{\mathsf{Person}(\mathsf{filip})\}$.

A paradox

Does $\mathcal{T} = \{\mathsf{Person} \sqsubseteq \exists\,\mathsf{childOf}.\,\mathsf{Person}\}$ model reality well?

Suppose we know that at least one person exists: $\mathcal{A} = \{\mathsf{Person}(\mathsf{filip})\}$.

Then, over finite models we can conclude that somebody is their own ancestor...

$$\mathcal{T}, \mathcal{A} \models_{\mathsf{fin}} \exists x\ \mathsf{childOf}^{+}(x, x)$$

Over unrestricted models we are safe:

$$\mathcal{T}, \mathcal{A} \not\models \exists x\ \mathsf{childOf}^{+}(x, x)$$

Query entailment

Landscape of query entailment

Entailment of CQs is EXPTIME-complete for

- $\mathcal{ALCH}$ [Ortiz, Šimkus, Eiter 2008], $\mathcal{ALCHQ}$ [Lutz 2008]

2EXPTIME-complete for

- $\mathcal{ALCI}$ [Lutz 2008], $\mathcal{ALCO}$ [Ngo, Ortiz, Šimkus 2016]
- $\mathcal{SH}$ [Eiter, Lutz, Ortiz, Šimkus '10], $\mathcal{S}$ [Ibáñez-García, Jung, Michielini, M. '25],
- $\mathcal{SHIQ}^r$ [Calvanese, Eiter, Ortiz 2007] [Glimm, Lutz, Horrocks, Sattler 2008]
- $\mathcal{SHOQ}^r$ [Glimm, Horrocks, Sattler 2008]
- $\mathcal{SOQ}^u$ [Gogacz, Gutiérrez-Basulto, Ibáñez-García, Jung, Murlak 2019]

decidable for

- $\mathcal{ALCHOIQ}b$ [Glimm, Rudolph 2010]

undecidable for

- $\mathcal{SHQ}^u$ [Horrocks, Sattler, Tobies 2000], $\mathcal{SIQ}^u$ [Kazakov, Sattler, Zolin 2007]
- $\mathcal{SHOIQ}^r$ and $\mathcal{SHOIF}$ [Rudolph 2016]

Landscape of query entailment: finite controllability

Finite controllability: $\models \; = \; \models_{\mathsf{fin}}$

$\mathcal{GF}$ [Bárány, Gottlob, Otto 2014] (covers $\mathcal{ALCHOI}b$)

$\mathcal{ALCOF}$ [Gogacz, Ibáñez-García, Murlak 2018]

Landscape of query entailment: finite controllability

Finite controllability: $\models \; = \; \models_{\text{fin}}$

$\mathcal{GF}$ [Bárány, Gottlob, Otto 2014] (covers $\mathcal{ALCHOI}b$)

$\mathcal{ALCOF}$ [Gogacz, Ibáñez-García, Murlak 2018]

No finite controllability: $\models \; \neq \; \models_{\text{fin}}$

$\mathcal{ALCIF}$

$A(a)$
$\top \sqsubseteq \exists r.\neg A$
$\top \sqsubseteq \exists^{\leq 1} r^{-}.\top$

$\mathcal{S}$

$A(a)$
$\top \sqsubseteq \exists r.\neg A$
tra(r)
$\exists x \, r(x,x)$

What about $\mathcal{ALCOQ}$ and up to $\mathcal{ALCHOQ}b$?

Landscape of finite entailment: conjunctive queries

Finite entailment of CQs is 2EXPTIME-complete for

- $\mathcal{GC}^2$ [Pratt-Hartmann 2009]
- $\mathcal{GF}$ [Bárány, Gottlob, Otto 2014]
- $\mathcal{SOI}$ and $\mathcal{SIF}$ [Gogacz, Ibáñez-García, Murlak 2018]
- $\mathcal{SHOI}^{\sqcap}$ [Danielski, Kieroński 2019]
- $\mathcal{SOQ}^u$ [Gogacz, Gutiérrez-Basulto, Ibáñez-García, Jung, Murlak 2019]

undecidable for

- $\mathcal{SHQ}^u$ and $\mathcal{SIQ}^u$ [Kazakov, Sattler, Zolin 2007]
- $\mathcal{SHOIF}$ [Rudolph 2016]

a challenge for

- $\mathcal{ALCOIF}$

Landscape of finite entailment: UCRPQs and UC2RPQs

Finite entailment of UCRPQ is 2EXPTIME-complete for

- $\mathcal{ALCI}$ and $\mathcal{ALCQ}$ [Gutiérrez-Basulto, Gutowski, Ibáñez-García, M. '22,'24]
  - should extend to $\mathcal{ALCOI}$ and $\mathcal{ALCOQ}$

a challenge for

- $\mathcal{ALCIQ}$

Finite entailment of UC2RPQs (two-way UCRPQs) is undecidable for

- $\mathcal{ALCOIF}$ [Rudolph 2016]

a challenge for

- $\mathcal{ALC}$

Query containment

Query containment without schema is well understood

UCQs: NP-complete [Chandra, Merlin '77]

UC2RPQs: EXPSPACE-complete [Florescu, Levy, Suciu '98] [Calvanese, De Giacomo, Lenzerini, Vardi '00]

Fragments complete for NP, co-NP^{NP}, PSPACE [Deutsch, Tannen '02] [Figueira, Godbole, Krishna, Martens, Niewerth, Trautner '20]

Dichotomy between EXPSPACE-hard and PSPACE-easy [Figueira '20]

Query containment modulo schema is mostly open

Containment of UCQs modulo full dependencies [Beeri, Vardi '84]

Containment of UCQs w. reachability modulo full dependencies w. reachability [Deutsch, Tannen '01]

Containment of UC2RPQs in acyclic UC2RPQs modulo Horn $\mathcal{ALCIQ}$ [Boneva, Groz, Hidders, Murlak, Staworko '23]

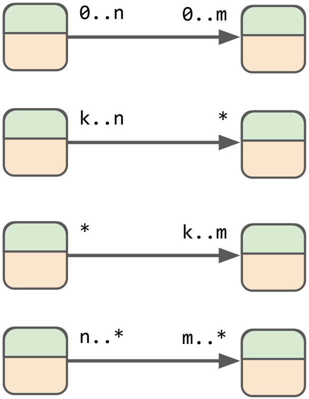
Strong results on query containment modulo schema/constraints over infinite graphs. [Calvanese, De Giacomo, Lenzerini '98,'08] [Calvanese, Ortiz, Šimkus '11]

Step 1: reduce containment to finite entailment

Theorem (Gutiérrez-Basulto, Gutowski, Ibáñez-García, Murlak 2024)

Containment of UC2RPQs modulo one-way schemas reduces to finite entailment.

- One-way schemas do not mix forward at-least and backward at-most constraints (and vice versa).
- Solving one instance of containment requires multiple instances of entailment.
- All instances of entailment involve single-node input graphs.



Corollary (from the proof)

Containment of UC2RPQs modulo schemas without at-least constraints is decidable in 2EXPTIME.

Step 2: use finite entailment

Theorem (Gutiérrez-Basulto, Gutowski, Ibáñez-García, Murlak 2024)

Query containment is 2EXPTIME*-complete for*

(a) *UCRPQs and one-way schemas,*

(b) *simple UC2RPQs and forward schemas.*

- Forward schemas do not use backward at-least and at-most constraints.
- Simple UC2RPQs do not use concatenation in regular expressions.

Conclusion

Summary and take-away

- Query containment and finite entailment are closely related problems.
- Rich body of techniques and results in DLs that can be potentially reused, because DLs are pretty good at capturing relevant schema information.
- Capturing more refined schemas requires fancier logics, new results are needed. Looks challenging, but this is what we like, isn't it?
- Complexity is high in general, but there's space for tractable special cases. The more we know about user's needs, the better we can tailor the algorithms.
- Other ways to build a bridge: closed predicates and mixed models.

Sometimes infinity is an oversimplification