ONTOLOGIES FOR KNOWLEDGE GRAPHS?

Markus Krötzsch†

reporting on joint work with Stefan Bischoff, Fredo Erxleben, Michael Günther, Maximilian Marx†, Julian Mendez, Ana Ozaki†, Axel Polleres, Sebastian Rudolph, Veronika Thost†, and Denny Vrandečić

† Knowledge-Based Systems
TU Dresden

Full paper: https://iccl.inf.tu-dresden.de/web/DL2017-keynote

DL Workshop 2017
2012: The Knowledge Graph

Google Inside Search

See it in action

“... one of the key breakthroughs behind the future of search”

Markus Krötzsch

Ontologies for Knowledge Graphs?

slide 3 of 33
What is a Knowledge Graph?

More than “a database used in an AI application”?

Characteristics of today’s KGs:
- Normalised: Data decomposed into small units (“edges”)
- Connected: Knowledge represented by relationships between these units
- Annotated: Enriched with contextual information to record meta-data and auxiliary details
What is a Knowledge Graph?

More than “a database used in an AI application”? 

**Characteristics of today’s KGs:**

**Normalised:** Data decomposed into small units (“edges”)

**Connected:** Knowledge represented by relationships between these units

**Annotated:** Enriched with contextual information to record meta-data and auxiliary details
What is a Knowledge Graph?

More than “a database used in an AI application”?

Characteristics of today’s KGs:

Normalised: Data decomposed into small units ("edges")

Connected: Knowledge represented by relationships between these units

Annotated: Enriched with contextual information to record meta-data and auxiliary details

• Typical for many KG applications
• Often comes with a promise of declarative processing
Summary

Knowledge graphs

- introduce graph-based data models
- requiring declarative analytics
- that make non-local connections
Summary

Knowledge graphs

- introduce graph-based data models
- requiring declarative analytics
- that make non-local connections
Summary

Knowledge graphs

• introduce graph-based data models
• requiring declarative analytics
• that make non-local connections

Conclusion

Symbolic KR is the key technology in modern data management especially in AI applications
Summary

Knowledge graphs
• introduce graph-based data models
• requiring declarative analytics
• that make non-local connections

Conclusion
Symbolic KR is the key technology in modern data management especially in AI applications

Not really happening
A Free Knowledge Graph

**Wikidata**

- Wikipedia’s knowledge graph
- Free, community-built database
- Large graph
  (July 2017: >165M statements on >29M entities)
- Large, active community
  (July 2017: >175,000 logged-in human editors)
- Many applications

Freely available, relevant, and active knowledge graph

[Vrandečić & K; Comm. ACM 2014]
I’m in ur phone ...

Who is Grover Cleveland
Tap to Edit

OK. Check it out:

 knowledge

Grover Cleveland
22nd and 24th president of the United States

Stephen Grover Cleveland was an American politician and lawyer who was the 22nd and 24th President of the United States. He won the popular vote for three presidential elections – in 1884, 1888, and 1892 – and was one of two Democrats to be elected president during the era of Republican political domination dating from 1861 to 1933. He was also the first and to date only President in American history to serve two non-consecutive terms in office.

See More on Wikipedia

Date of birth
March 18, 1837

Birthplace
Caldwell

Date of death
June 24, 1908

Deathplace
Princeton
Tim Berners-Lee (Q80)

British computer scientist

TimBL | Sir Tim Berners-Lee | Timothy John Berners-Lee | TBL | Tim Berners Lee | T. Berners-Lee | T Berners-Lee | Tim Berners-Lee | T.J. Berners-Lee
Tim Berners-Lee

British computer scientist

instance of human

› 1 reference
Tim Berners-Lee

instance of
  human
  1 reference

employer
  CERN
  start time 1984
  end time 1994
  position held Fellow
  0 references

award received
  Queen Elizabeth Prize for Engineering
    point in time 2013
    together with
      Robert Kahn
      Vint Cerf
      Louis Pouzin
      Marc Andreessen
    1 reference
Statements in Wikidata

Wikidata’s basic information units

• Built from Wikidata items (“CERN”, “Vint Cerf”), Wikidata properties (“award received”, “end time”), and data values (“2013”)

• Based on directed edges (“Tim Berners-Lee employer → CERN”)

• Annotated with property-value pairs (“end time: 1994”)
  – same property can have multiple annotation values (“together with: Robert Kahn, Vint Cerf, . . .”)
  – same properties/values used in directed edges and annotations

• Items and properties can be subjects/values in statements

• Multi-graph
Elizabeth Taylor (Q34851)
Elizabeth Rosemond Taylor | Liz Taylor | Dame Elizabeth Rosemond Taylor
British-American actress

instance of: Elizabeth Taylor is a(n) human

Fig.: Taylor standing in multiple relations; from https://tools.wmflabs.org/sqid/#/view?id=Q34851

Markus Krötzsch  Ontologies for Knowledge Graphs?  slide 13 of 33
Wikidata Statements in Terms of Graphs

Elizabeth Taylor (Q34851)

spouse

Richard Burton

start time

10 October 1975

end time

29 July 1976
Wikidata Statements in Terms of Graphs

“Property Graph”:

Elizabeth Taylor (Q34851)

spouse

Richard Burton

start time: 10 October 1975
end time: 29 July 1976

Taylor

spouse

Burton

start time: 1975-10-10
end time: 1976-07-29
Wikidata Statements in Terms of Graphs

“Property Graph”:

Taylor --spouse--> Burton

start time: 1975-10-10
end time: 1976-07-29

“RDF”:

Taylor <spouse_in> 1975-10-10

Taylor <spouse_out> 1976-07-29

Richard Burton

start time: 10 October 1975
end time: 29 July 1976
Ontological Modelling in Wikidata

Classification

• 25,298,346 instance of statements (for 84.9% of entities)
• 2,056,181 subclass of statements (for 4.5% of entities)

Property characteristics/constraints

• symmetric property (17 instances)
• transitive property (8 instances)
• 12,595 statements specifying other constraints (domain, range, disjointness, . . . )
Ontological Modelling in Wikidata

Classification

- 25,298,346 instance of statements (for 84.9% of entities)
- 2,056,181 subclass of statements (for 4.5% of entities)

Property characteristics/constraints

- symmetric property (17 instances)
- transitive property (8 instances)
- 12,595 statements specifying other constraints (domain, range, disjointness, ...)

Markus Krötzsch
Queries on Wikidata

**SPARQL query service:** [https://query.wikidata.org](https://query.wikidata.org)

- officially maintained, live data
- based on RDF mapping [Erxleben et al., ISWC 2014]
- heavily used: 60M–135M queries per month

Initial analysis of the non-public logs:

- ≤ 1% queries from human traffic (400–500K per month)
- ≥ 99% service calls from tools and robots
- Irregular distributions and biases – hard to analyse

Property paths used for transitivity reasoning

- used in about 50% of human subclass-of queries (20K)
- over 500K queries with subclass-of paths overall

(statistics for May 2017)
Queries on Wikidata

**SPARQL query service:** https://query.wikidata.org

- officially maintained, live data
- based on RDF mapping [Erxleben et al., ISWC 2014]
- heavily used: 60M–135M queries per month

**Initial analysis** of the non-public logs:

- ≤1% queries from human traffic (400–500K per month)
- ≥99% service calls from tools and robots
- Irregular distributions and biases – hard to analyse
Queries on Wikidata

**SPARQL query service:** https://query.wikidata.org

- officially maintained, live data
- based on RDF mapping [Erxleben et al., ISWC 2014]
- heavily used: 60M–135M queries per month

**Initial analysis** of the non-public logs:

- \( \leq 1\% \) queries from human traffic (400–500K per month)
- \( \geq 99\% \) service calls from tools and robots
- Irregular distributions and biases – hard to analyse

**Property paths used for transitivity reasoning**

- used in about 50% of human subclass-of queries (20K)
- over 500K queries with subclass-of paths overall

(statistics for May 2017)
OBQA via SPARQL

SPARQL is actually powerful enough for OWL QL reasoning [Bischoff et al., ISWC 2014]

... but the queries then are getting lengthy ...

```
x (sCO | eqC | ^eqC | intListMember | owl:someValuesFrom |
   (owl:onProperty / (INV | SPOEqP)^ / (^owl:onProperty | rdfs:domain | rdfs:range))^) ^C . |
   {?C subClassOf owl:Nothing} UNION
   {?C subClassOf ?D1 {{?C subClassOf ?D2} UNION univClass[?D2]} |
     {?D1 disjointClasses ?D2} UNION
     {?V rdfs:type owl:AllDisjointClasses . twoMembers[?V, ?D1, ?D2]}
   } UNION
   {?C (owl:onProperty / (INV | SPOEqP)^) ^P . |
    {?P subPropertyOf owl:bottomObjectProperty} UNION
    {?P subPropertyOf ?Q1 {{?P subPropertyOf ?Q2} UNION univProperty[?Q2]} |
     {?Q1 (owl:propertyDisjointWith | ^owl:propertyDisjointWith) ?Q2} UNION
     {?V rdfs:type owl:AllDisjointProperties . twoMembers[?V, ?Q1, ?Q2]}
    } |
   }
```

Fig.: A query that checks if \( x \) is equivalent to \( \bot \) (abbreviated)
Beyond OWL QL

SPARQL cannot support arbitrary OWL reasoning:

- computing power limited by data complexity
- SPARQL can only perform reasoning in NL
Beyond OWL QL

SPARQL cannot support arbitrary OWL reasoning:

- computing power limited by data complexity
- SPARQL can only perform reasoning in NL

Queries with higher data complexities?

- Datalog: PTime-complete data complexity
- Datalog can be used for “query-based” EL reasoning

[K, IJCAI 2011]
Beyond OWL QL

SPARQL cannot support arbitrary OWL reasoning:
- computing power limited by data complexity
- SPARQL can only perform reasoning in NL

Queries with higher data complexities?
- Datalog: PTime-complete data complexity
- Datalog can be used for “query-based” EL reasoning
  [K, IJCAI 2011]

Query-Based Reasoning:
- ontological information as part of data
- logic for meta-reasoning on top
- same data can be viewed under different semantics
Ontologies for Wikidata?
Wikidata declares the *spouse* property to be symmetric:

A Simple Example

\[
\forall x, y, z_1, z_2, v. \text{spouse} \text{ in } (x, v) \land \text{spouse} \text{ out } (v, y) \land \text{start} (v, z_1) \land \text{end} (v, z_2) \rightarrow \exists w. \text{spouse} \text{ in } (y, w) \land \text{spouse} \text{ out } (x, y) \land \text{start} (w, z_1) \land \text{end} (w, z_2)
\]
A Simple Example

Wikidata declares the **spouse** property to be symmetric:

```
\begin{align*}
\text{Taylor} & \xrightarrow{\text{spouse}_{\text{in}}} \text{Burton} \\
& \xleftarrow{\text{spouse}_{\text{out}}} \\
\text{1975-10-10} & \xrightarrow{\text{start time}} \text{1976-07-29} \\
\Rightarrow \\
\text{Taylor} & \xrightarrow{\text{spouse}_{\text{out}}} \text{Burton} \\
& \xleftarrow{\text{spouse}_{\text{in}}} \\
\text{1975-10-10} & \xleftarrow{\text{start time}} \text{1976-07-29}
\end{align*}
```

**ABox:**

\[
\begin{align*}
\text{spouse}_{\text{in}}(\text{taylor, } s) & \quad \text{spouse}_{\text{out}}(s, \text{burton}) \\
\text{start}(s, 1975-10-10) & \quad \text{end}(s, 1976-07-29)
\end{align*}
\]
A Simple Example

Wikidata declares the `spouse` property to be symmetric:

\[ \forall x, y, z_1, z_2, v.\text{spouse}_\text{in}(x, v) \land \text{spouse}_\text{out}(v, y) \land \text{start}(v, z_1) \land \text{end}(v, z_2) \rightarrow \exists w.\text{spouse}_\text{in}(y, w) \land \text{spouse}_\text{out}(x, y) \land \text{start}(w, z_1) \land \text{end}(w, z_2) \]
Existential rules

\[
\text{spouse}_\text{in}(x, v) \land \text{spouse}_\text{out}(v, y) \land \text{start}(v, z_1) \land \text{end}(v, z_2) \\
\rightarrow \exists w. \text{spouse}_\text{in}(y, w) \land \text{spouse}_\text{out}(x, y) \land \text{start}(w, z_1) \land \text{end}(w, z_2)
\]

This axiom is an existential rule
Existential rules

\[
\begin{align*}
\text{spouse}_{\text{in}}(x, v) \land \text{spouse}_{\text{out}}(v, y) \land \text{start}(v, z_1) \land \text{end}(v, z_2) \\
\rightarrow \exists w. \text{spouse}_{\text{in}}(y, w) \land \text{spouse}_{\text{out}}(x, y) \land \text{start}(w, z_1) \land \text{end}(w, z_2)
\end{align*}
\]

This axiom is an existential rule

- it is not expressible in Datalog
Existential rules

\[
\text{spouse}_{\text{in}}(x, v) \land \text{spouse}_{\text{out}}(v, y) \land \text{start}(v, z_1) \land \text{end}(v, z_2) \\
\rightarrow \exists w. \text{spouse}_{\text{in}}(y, w) \land \text{spouse}_{\text{out}}(x, y) \land \text{start}(w, z_1) \land \text{end}(w, z_2)
\]

This axiom is an **existential rule**

- it is **not expressible** in Datalog
- it is **not expressible** in DL
Existential rules

\[
\begin{align*}
\text{spouse}_{\text{in}}(x, v) & \land \text{spouse}_{\text{out}}(v, y) & \land \text{start}(v, z_1) & \land \text{end}(v, z_2) \\
\rightarrow & \exists w. \text{spouse}_{\text{in}}(y, w) & \land \text{spouse}_{\text{out}}(x, y) & \land \text{start}(w, z_1) & \land \text{end}(w, z_2)
\end{align*}
\]

This axiom is an **existential rule**

- it is **not** expressible in Datalog
- it is **not** expressible in DL
- it is **not** linear
Existential rules

\[
\text{spouse}_{\text{in}}(x, v) \land \text{spouse}_{\text{out}}(v, y) \land \text{start}(v, z_1) \land \text{end}(v, z_2) \\
\rightarrow \exists w. \text{spouse}_{\text{in}}(y, w) \land \text{spouse}_{\text{out}}(x, y) \land \text{start}(w, z_1) \land \text{end}(w, z_2)
\]

This axiom is an existential rule

- it is not expressible in Datalog
- it is not expressible in DL
- it is not linear
- it is not (frontier) guarded
Existential rules

This axiom is an existential rule

\[ \text{spouse}_{\text{in}}(x, v) \wedge \text{spouse}_{\text{out}}(v, y) \wedge \text{start}(v, z_1) \wedge \text{end}(v, z_2) \]

\[ \rightarrow \exists w. \text{spouse}_{\text{in}}(y, w) \wedge \text{spouse}_{\text{out}}(x, y) \wedge \text{start}(w, z_1) \wedge \text{end}(w, z_2) \]

- it is not expressible in Datalog
- it is not expressible in DL
- it is not linear
- it is not (frontier) guarded
- it is not acyclic (w.r.t. predicate dependencies)
Existential rules

\[ \text{spouse}_{in}(x, v) \land \text{spouse}_{out}(v, y) \land \text{start}(v, z_1) \land \text{end}(v, z_2) \]
\[ \rightarrow \exists w. \text{spouse}_{in}(y, w) \land \text{spouse}_{out}(x, y) \land \text{start}(w, z_1) \land \text{end}(w, z_2) \]

This axiom is an existential rule

- it is not expressible in Datalog
- it is not expressible in DL
- it is not linear
- it is not (frontier) guarded
- it is not acyclic (w.r.t. predicate dependencies)

but it might be weakly acyclic/frontier guarded
(depends on other axioms)
Observation: Normalisation may destroy syntactic properties
[K & Thost; ISWC 2016]

- **Acyclicity** properties are mostly often preserved
- **Linearity** and **guardedness** are lost (syntactically)
- Can sometimes recover by denormalising rules
Observation: Normalisation may destroy syntactic properties
[K & Thost; ISWC 2016]

- Acyclicity properties are mostly often preserved
- Linearity and guardedness are lost (syntactically)
- Can sometimes recover by denormalising rules

Existential rules are a first step, but:

- Normalised rules are hard to read and write
- Not expressive enough, e.g., cannot copy arbitrary annotation sets
- Loss of structure by flattening annotations, e.g., cannot have closed-world negation on annotation sets
Annotated Logics
Idea: Change from relational structures to “relational structures with annotated tuples” [Marx, K, Thost, IJCAI 2017]
Idea: Change from relational structures to “relational structures with annotated tuples” [Marx, K, Thost, IJCAI 2017]

Multi-Attributed Relational Structures (MARS):

- standard interpretation domain $\Delta^I$
- finite annotation sets $S \in \mathcal{P}_{\text{fin}}(\Delta^I \times \Delta^I)$
- $n$-ary relations $r$ interpreted as $r^I \subseteq (\Delta^I)^n \times \mathcal{P}_{\text{fin}}(\Delta^I \times \Delta^I)$
**MARS**

**Idea:** Change from relational structures to “relational structures with annotated tuples” [Marx, K, Thost, IJCAI 2017]

Multi-Attribute Relational Structures (MARS):
- standard interpretation domain $\Delta^I$
- finite annotation sets $S \in \mathcal{P}_{\text{fin}}(\Delta^I \times \Delta^I)$
- $n$-ary relations $r$ interpreted as $r^I \subseteq (\Delta^I)^n \times \mathcal{P}_{\text{fin}}(\Delta^I \times \Delta^I)$

Multi-Attribute Predicate Logic (MAPL)
- Ground fact:
  spouse(taylor, burton)@{start : 1975, end : 1976}
- Object and set variables:
  $\forall x, y, Z.\text{spouse}(x, y)@Z \rightarrow \text{spouse}(y, x)@Z$
Expressivity of MAPL

Theorem: MAPL is equivalent to Weak Second-Order Logic, hence reasoning is not semi-decidable.
Expressivity of MAPL

Theorem: MAPL is equivalent to Weak Second-Order Logic, hence reasoning is not semi-decidable.

Research goal: Identify practical fragments
Expressivity of MAPL

Theorem: MAPL is equivalent to Weak Second-Order Logic, hence reasoning is not semi-decidable.

Research goal: Identify practical fragments

A decidable fragment:

MAPL Rules (MARPL)
- Horn rules, with all variables universally quantified
- all set variables bound in body atoms

Example: $\forall x, y, Z.\text{spouse}(x, y)@Z \rightarrow \text{spouse}(y, x)@Z$
MARPL: Additional Features

We really need more expressive features
MARPL: Additional Features

We really need more expressive features

Conditions on annotation sets $Z$

- $[\text{start} : 1975, \text{end} : \ast](Z)$: “$Z$ has given start and some end, but nothing more”
- $[\text{start} : 1975](Z)$: “$Z$ has given start, and possibly more”

$\rightarrow$ supported in MARPL rule bodies
We really need more expressive features

**Conditions on annotation sets** $Z$

- $[\text{start} : 1975, \text{end} : \ast](Z)$: “$Z$ has given start and some end, but nothing more”
- $\lfloor \text{start} : 1975 \rfloor(Z)$: “$Z$ has given start, and possibly more”

$\Rightarrow$ supported in MARPL rule bodies

**Inferring new annotation sets**

- Support declarative definition of deterministic functions that derive new sets
- Example:

  $$\text{employer}(x, \text{cern})@Z \land [\text{pos} : \text{fellow}](Z) \Rightarrow \text{cernFellow}(x)@[\text{start} : Z.\text{start}, \text{end} : Z.\text{end}]$$

$\Rightarrow$ supported in MARPL rule heads
Theorem: Conjunctive query answering over MARPL ontologies is ExpTime-complete, both for combined complexity and for data complexity.
Theorem: Conjunctive query answering over MARPL ontologies is ExpTime-complete, both for combined complexity and for data complexity.

Problem?
Theorem: Conjunctive query answering over MARPL ontologies is ExpTime-complete, both for combined complexity and for data complexity.

Problem?

- **Not really**: hardness requires annotation sets of unbounded size (not a practical concern)
- **Actually, it’s a feature**: high data complexity enables powerful meta-reasoning in query-based approaches
Attributed Description Logics

MARPL is a simple rule language ("Datalog for annotated logic")

How about DLs?
Attributed Description Logics

MARPL is a simple rule language ("Datalog for annotated logic")

How about DLs?

Attributed Description Logics
see DL talk later today [K et al., ISWC 2017]
Attributed Description Logics

MARPL is a simple rule language ("Datalog for annotated logic")

How about DLs?

Attributed Description Logics
see DL talk later today [K et al., ISWC 2017]

How about attributed existential rules?

~ future work
The Future of KR
Problem solved?

So all we need to marry KG and KR are attributed logics?
Problem solved?

So all we need to marry KG and KR are attributed logics?

Surely not – many other areas need more work!
Problem solved?

So all we need to marry KG and KR are attributed logics?

Surely not – many other areas need more work!

We also need to change some of our premises:

<table>
<thead>
<tr>
<th>Traditional KR View</th>
<th>vs.</th>
<th>Knowledge Graphs View</th>
</tr>
</thead>
<tbody>
<tr>
<td>schema first</td>
<td></td>
<td>data first</td>
</tr>
<tr>
<td>unique purpose</td>
<td></td>
<td>multi-purpose</td>
</tr>
<tr>
<td>fixed application</td>
<td></td>
<td>emerging applications</td>
</tr>
<tr>
<td>closed expert team</td>
<td></td>
<td>open community/many teams</td>
</tr>
</tbody>
</table>
Still Looking for the “Unifying Logic”?

- User Interface & Applications
- Trust
- Proof
- Unifying Logic
- Query: SPARQL
- Ontology: OWL
- Rule: RIF
- RDFS
- Data interchange: RDF
- XML
- URI/IRI

Markus Krötzsch
Ontologies for Knowledge Graphs?
slide 31 of 33
Conclusions

Summary

- Knowledge Graphs are enriched graphs
- Wikidata: large ABox / “ontology” / path queries
- Query-based reasoning: plug’n’play semantics for data
- Existential rules & DLs: struggling with annotations
- Attributed logics: MAPL & MARPL (& attributed DLs)

What next?

View KR as a declarative computing paradigm & start facing the competition in this space!

Revisit “Computing in Logic” (but don’t go back to Prolog!)
References


