Knowledge Graphs, Rule Extraction and Data Access

M. Thomazo

What is a Knowledge Graph ?

"Since Google started an initiative called Knowledge Graph in 2012, a substantial amount of research has used the phrase knowledge graph as a generalized term.
Although there is no clear definition for the term knowledge graph, it is sometimes used as synonym for ontology.
One common interpretation is that a knowledge graph represents a collection of interlinked descriptions of entities – real-world objects, events, situations or abstract concepts.
Unlike ontologies, knowledge graphs, such as Google's Knowledge Graph, often contain large volumes of factual information with less formal semantics.
In some contexts, the term knowledge graph is used to refer to any knowledge base that is represented as a graph."
(Wikipedia, 28 Oct. 2019)

Examples of Knowledge Graphs

WIKIDATA

https://www.wikidata.org/wiki/Q937

yago select knowledge

https://tinyurl.com/yxnfrzc6

DBpedia

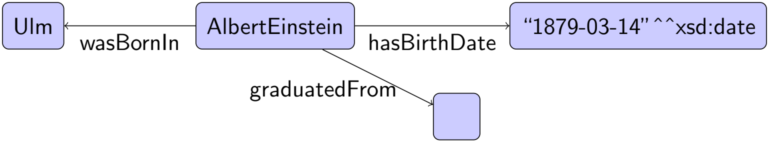
http://fr.dbpedia.org/page/Albert_Einstein

Google Knowledge Graph https://tinyurl.com/y53nzrqv

RDF (Resource Description Framework)

RDF graph : set of triples of the form (subject, predicate, object)
- subject : IRI (Internationalized Resource Identifier) or blank node
- predicate : IRI
- object : IRI, blank node, or literal

- W3C standard for exchanging graphs
- Directed labelled (multi-) graphs
- Nodes are entities (vertices labelled with IRIs), data values (vertices labelled with literals), or blank nodes (vertices without labels)



SPARQL (SPARQL Protocol and RDF Query Language)

W3C standard query (and update) language for RDF data
We focus on SELECT queries

- PREFIX declaration: specifies namespaces
- SELECT clause: output variables (strings that begin with ?)
- WHERE clause:
  - basic graph patterns (BGP): sets of triple patterns: $\langle s, p, o \rangle$ where s and o are RDF terms or variables and p is an IRI or variable, written as a whitespace-separated list in the query
  - possibly property path patterns instead of triple patterns: p can be a property path $\sim$ regular expression built on IRIs
  - possibly FILTER, UNION, OPTIONAL
- Solution set modifiers (DISTINCT, LIMIT, ORDER BY...)

SPARQL Examples

SELECT Query

```
PREFIX dc10:  <http://purl.org/dc/elements/1.0/>
PREFIX dc11:  <http://purl.org/dc/elements/1.1/>

SELECT ?title ?author
   WHERE  {
     { ?book dc10:title ?title .
       ?book dc10:creator ?author }
     UNION
     { ?book dc11:title ?title .
       ?book dc11:creator ?author }
   }
```

SPARQL Examples

Property paths

Some property paths constructors

path1/path2	path1 followed by path 2
ˆpath1	backwards path (object to subject)
path1|path2	path1 or path2
path1*	path1 repeated zero or more times
path1+	path1 repeated one or more times

SPARQL Examples

Property paths

```
{
  ?x foaf:mbox <mailto:alice@example> .
  ?x foaf:knows/foaf:name ?name .
 }
```

```
{
  ?x foaf:mbox <mailto:alice@example> .
  ?x foaf:knows+/foaf:name ?name .
 }
```

```
 { ?x rdf:type/rdfs:subClassOf* ?type }
```

Hands-on Session: Querying Wikidata

Wikidata

- Free knowledge base that anyone can edit
- Wikipedia's knowledge graph
- Large graph: >890M statements on >70M entities on Jan. 2020
- Large, active community (Jan. 2020: >3M registered users)
- Launched in 2012
- Many applications
  - Wikipedia: inter-language links, auto-generated info boxes
  - Application-specific data-excerpts
  - Data integration and quality control
  - ...

Hands-on Session: Querying Wikidata

Principles of Wikidata

- Open editing: Anyone can extend or modify content;
- Community control: The users decide what is stored and how it is represented;
- Plurality: There might not be one truth but several co-existing views; such complexity must be supported;
- Secondary data: All content should be supported by external, primary sources;
- Multi-lingual data: One site serves all languages; labels are translated: content is the same for all;
- Easy access: Technical and legal barriers for data re-use are minimized;
- Wikidata remains work in progress; active community.

Hands-on Session: Querying Wikidata

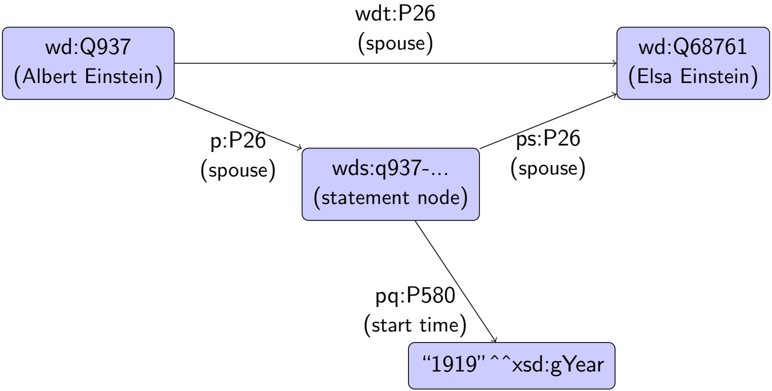
Wikidata data (simplified)

- Statements: Wikidata's basic information units, sourced claims for several properties that an entity might have
  - Built from Wikidata items ("Albert Einstein"), Wikidata properties ("date of birth", "spouse"), and data values ("1879")
  - Items and properties can be subjects/values in statements
  - Annotated with property-value pairs ("start time: 1919")
- Entities identified by language-independent ids, starting by Q for items and P for properties (e.g. Q937, P40)
- Wikidata is internally stored using a JSON format but is converted in RDF for several purposes, and in particular for export for external use and for importing data into Wikidata's SPARQL query service

Hands-on Session: Querying Wikidata

RDF encoding of Wikidata statements

We present the basics needed for today's hands-on session.



Hands-on Session: Querying Wikidata

RDF encoding of Wikidata statements

- Each statement is represented by a resource in RDF ("wds:q937-881C4FA7-075C-4D48-8182-77D69CA6309C")
- Direct single-triple links from subject to value are added ("wd:Q937 wdt:P26 wd:Q68761")
- Each Wikidata property turns into several RDF properties for different uses in encoding ("wdt:P26", "wd:P26"...)
- Order of qualifiers or statements is not represented in RDF

The complete Wikidata-to-RDF documentation is available online:
https://www.mediawiki.org/wiki/Wikibase/Indexing/RDF_Dump_Format

Hands-on Session: Querying Wikidata

- Wikidata:
  https://www.wikidata.org/wiki/Wikidata:Main_Page
- Wikidata Query Service: https://query.wikidata.org/
- Queries:
  - List of Albert Einstein's children with their birth date and place.
  - Subproperties of the property student.
  - List of students of Einstein or of one of his students.
  - List of singers (occupation singer) having French and German citizenship.
  - List of singers having French or German citizenship.
  - List of paintings from European painters that are located in France.
  - List of French presidents with the start date of their presidency.
  - List of presidents of the French Fifth Republic with the start date of their presidency.
  - Number of presidents of the French Fifth Republic.
  - List of proteins encoded by some gene located on chromosome Y.

Learning Rules from Knowledge Graphs

Horn Rule:

$$B_1 \wedge B_2 \ldots \wedge B_n \rightarrow r(x, y)$$

- $B_1, \ldots, B_n$ are atoms, $r(x, y)$ is an atom as well. $r(x, y)$ is called the *head*, r the *head relation.*
- instantiation of a rule: copy of a rule where each variable is replaced by a constant;

Learning Rules from Knowledge Graphs

Horn Rule:

$$B_1 \wedge B_2 \ldots \wedge B_n \rightarrow r(x, y)$$

- $B_1, \ldots, B_n$ are atoms, $r(x, y)$ is an atom as well. $r(x, y)$ is called the *head*, r the *head relation.*
- instantiation of a rule: copy of a rule where each variable is replaced by a constant;

Examples:

$$\text{isMarriedTo}(x, z) \wedge \text{hasChild}(z, y) \rightarrow \text{hasChild}(x, y)$$

Learning Rules from Knowledge Graphs

Horn Rule:

$$B_1 \wedge B_2 \ldots \wedge B_n \rightarrow r(x, y)$$

- $B_1, \ldots, B_n$ are atoms, $r(x, y)$ is an atom as well. $r(x, y)$ is called the *head*, r the *head relation.*
- instantiation of a rule: copy of a rule where each variable is replaced by a constant;

Examples:

$$\text{isMarriedTo}(x, z) \wedge \text{hasChild}(z, y) \rightarrow \text{hasChild}(x, y)$$

$$\text{directed}(x, z) \wedge \mathit{hasActor}(z, y) \rightarrow \text{isMarried}(x, y)$$

Learning Rules from Knowledge Graphs

Horn Rule:

$$B_1 \wedge B_2 \ldots \wedge B_n \rightarrow r(x, y)$$

- $B_1, \ldots, B_n$ are atoms, $r(x, y)$ is an atom as well. $r(x, y)$ is called the *head*, r the *head relation.*
- instantiation of a rule: copy of a rule where each variable is replaced by a constant;

Examples:

$$\text{isMarriedTo}(x, z) \wedge \text{hasChild}(z, y) \rightarrow \text{hasChild}(x, y)$$

$$\text{directed}(x, z) \wedge \textit{hasActor}(z, y) \rightarrow \text{isMarried}(x, y)$$

Question: how to learn *interesting* rules from a knowledge graph?

Interesting Rule?

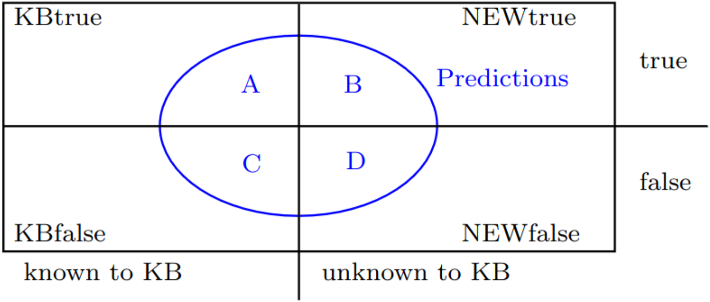


Figure taken from Fast rule mining in ontological knowledge bases with AMIE, Galárraga et al., VLDB 2015.

Quantitative Measures

Support

- quantifies the number of correct predictions in the existing data
- one wants *monotonicity*

Quantitative Measures
Support

- quantifies the number of correct predictions in the existing data
- one wants *monotonicity*
  - more specific rules has smaller support
  - why is this property interesting?

Quantitative Measures

Support

- quantifies the number of correct predictions in the existing data
- one wants *monotonicity*
  - more specific rules has smaller support
  - why is this property interesting?
- Which notion of support can you think of?

Quantitative Measures

Support

- quantifies the number of correct predictions in the existing data
- one wants *monotonicity*
  - more specific rules has smaller support
  - why is this property interesting?
- Which notion of support can you think of?
- defined by authors as the number of pairs (x, y) such that B and $r(x, y)$ hold in the database.

Head Coverage

Support is an absolute measure.

- not suited to define thresholds independently of the knowledge base
- wants to take it into account
- normalizing by the size of the KB not adapted to small size relations
- normalizing the support by the size of the head relation.

Impact of the Closed World Assumption vs Open World Assumption

Standard confidence of a rule is the ratio:

$$\frac{\text{support}}{\#\text{instantiation}(\vec{B})}$$

This is well adapted in a *closed-world* scenario, where the absence of a fact is equivalent to its negation.

Impact of the Closed World Assumption vs Open World Assumption

Standard confidence of a rule is the ratio:

$$\frac{\text{support}}{\#\text{instantiation}(\vec{B})}$$

This is well adapted in a *closed-world* scenario, where the absence of a fact is equivalent to its negation.

Knowledge base are in a setting where we discard the closed world assumption.

Impact of the Closed World Assumption vs Open World Assumption

Standard confidence of a rule is the ratio:

$$\frac{\text{support}}{\#\text{instantiation}(\vec{B})}$$

This is well adapted in a *closed-world* scenario, where the absence of a fact is equivalent to its negation.

Knowledge base are in a setting where we discard the closed world assumption.

How to assess the confidence of a rule?

Partially Closed World Assumption (PCA)

Negative examples are generated in AMIE using the *partially closed world assumption*:

> "If $r(x, y)$ is in the knowledge base, then for any y', $r(x, y')$ is in the knowledge base if and only if holds in the real world."

Partially Closed World Assumption (PCA)

Negative examples are generated in AMIE using the *partially closed world assumption*:

> "If $r(x, y)$ is in the knowledge base, then for any y', $r(x, y')$ is in the knowledge base if and only if holds in the real world."

How to adapt confidence with this assumption in mind?

Problem: Search Space is Huge

- (infinitely) many Horn rules;
- AMIE focuses only on:
  - connected Horn rules
  - closed Horn rules
  - rules containing atoms of the shape $r(x, x)$
- but AMIE allows for recursive rules (common relation between body and head)
- parameters: minimum head coverage, maximum number of atoms in rules, minimum confidence.
  - reduce the search space, while (hopefully) preserving all interesting rules.

AMIE Pseudo-code

Algorithm 1 Rule Mining

function AMIE(KB $\mathcal{K}$, $minHC$, $maxLen$, $minConf$)
    $q = [r_1(x, y), r_2(x, y) \ldots r_m(x, y)]$
    $out = \langle\rangle$
    while $\neg q.isEmpty()$ **do**
        $r = q.dequeue()$
        if $AcceptedForOutput(r, out, minConf)$ **then**
            $out.add(r)$
        end if
        if $length(r) < maxLen$ **then**
            $R(r) = Refine(r)$
            for all rules $r_c \in R(r)$ **do**
                if $hc(r_c) \geq minHC$ & $r_c \notin q$ **then**
                    $q.enqueue(r_c)$
                end if
            end for
        end if
    end while
    return out
end function

Further Reading

To know more about AMIE(+), and notably optimizations:

- Fast rule mining in ontological knowledge bases with AMIE, Galárraga et al., VLDB 2015.

The Challenge of Accessing Big Data

The Statoil Example

Experts in geology and geophysics develop stratigraphic models of unexplored areas on the basis of data acquired from previous operations at nearby geographical locations.

Fact:

- 1000TB of relational data
- Using diverse schemata
- spread over 2000 tables, over mutiple individual databases

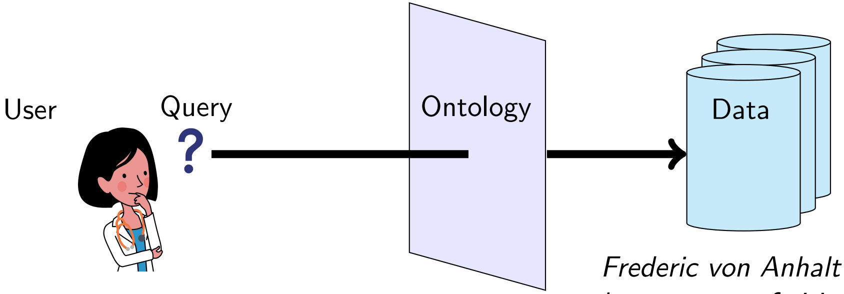
Data Access for Exploration::

- 900 experts in Statoil Exploration
- Up to 4 days for new data access queries – assistance by IT experts
- 30 - 70 % of time spent on data gathering

Accessing Data through an Ontology

- Knowledge graphs and SPARQL queries allow us to get answers to complex queries
- But SPARQL queries may become very (too) complex
- Need for a way of formulating simpler queries, closer to the natural language of the user, and still get all the answers from the data
- Ontologies allow to formalize knowledge and delegate the reasoning to the machine

Accessing Data through an Ontology



Which European citizens were married to Zsa Zsa Gabor?

Someone with country of citizenship in Europe is a European citizen

Frederic von Anhalt has country of citizenship Germany

Germany is in Europe

Frederic von Anhalt was married to Zsa Zsa Gabor

Accessing Data through an Ontology

```
SELECT DISTINCT ?spouse WHERE
{   ZsaZsaGabor hasSpouse ?spouse.
    ?spouse hasCountryOfCitizenship ?country.
    ?country hasContinent Europe. }
```

Accessing Data through an Ontology

```
SELECT DISTINCT ?spouse WHERE
{   ZsaZsaGabor hasSpouse ?spouse.
    ?spouse hasCountryOfCitizenship ?country.
    ?country hasContinent Europe. }
```

Express relationships : $\mathit{hasContinent}(x, \mathit{Europe}) \implies \mathit{EuropeanCountry}(x)$

$\mathit{hasCountryOfCitizenship}(x, y) \wedge \mathit{EuropeanCountry}(y) \implies \mathit{EuropeanCitizen}(x)$

as an OWL (Web Ontology Language) ontology

```
EuropeanCountry rdfs:subClassOf owl:Restriction
EuropeanCountry owl:onProperty hasContinent
EuropeanCountry owl:ObjectHasValue Europe
EuropeanCitizen rdfs:subClassOf owl:Restriction
EuropeanCitizen owl:onProperty hasCountryOfCitizenship
EuropeanCitizen owl:someValuesFrom EuropeanCountry
```

Accessing Data through an Ontology

```
SELECT DISTINCT ?spouse WHERE
{   ZsaZsaGabor hasSpouse ?spouse.
    ?spouse hasCountryOfCitizenship ?country.
    ?country hasContinent Europe. }
```

Express relationships : $hasContinent(x, Europe) \implies EuropeanCountry(x)$

$hasCountryOfCitizenship(x, y) \wedge EuropeanCountry(y) \implies EuropeanCitizen(x)$

as an OWL (Web Ontology Language) ontology

```
EuropeanCountry rdfs:subClassOf owl:Restriction
EuropeanCountry owl:onProperty hasContinent
EuropeanCountry owl:ObjectHasValue Europe
EuropeanCitizen rdfs:subClassOf owl:Restriction
EuropeanCitizen owl:onProperty hasCountryOfCitizenship
EuropeanCitizen owl:someValuesFrom EuropeanCountry
```

```
SELECT DISTINCT ?spouse WHERE
{   ZsaZsaGabor hasSpouse ?spouse.
    ?spouse rdf:type EuropeanCitizen. }
```

Ontology-Based Data Access: An Example

Ontology – logical representation of the domain of interest

$$\forall X(\mathrm{Researcher}(X) \rightarrow \exists Y(\mathrm{worksFor}(X, Y) \land \mathrm{Project}(Y)))$$
$$\forall X(\mathrm{Project}(X) \rightarrow \exists Y(\mathrm{worksFor}(Y, X) \land \mathrm{Researcher}(Y)))$$
$$\forall X, Y(\mathrm{worksFor}(X, Y) \rightarrow \mathrm{Researcher}(X) \land \mathrm{Project}(Y))$$
$$\forall X(\mathrm{Project}(\mathrm{X}) \rightarrow \exists Y(\mathrm{PrName}(X, Y)))$$

Ontology-Based Data Access: An Example

Relational Database D – a single database that represents the sources

worksIn

SSN	Name
100	AAA
200	BBB
300	CCC

Intuitively, represents "The research with SSN 100 works for project AAA".

Ontology-Based Data Access: An Example

Mappings M – semantically link data at the sources with the ontology

$$\begin{array}{lcl} & & \mathrm{Researcher}(\mathrm{person}(\mathsf{SSN}))\wedge \\ \mathsf{SELECT\ SSN,\ Name} & \rightsquigarrow & \mathrm{Project}(\mathrm{proj}(\mathsf{Name}))\ \wedge \\ \mathsf{FROM}\ \mathrm{worksIn} & & \mathrm{worksFor}(\mathrm{person}(\mathsf{SSN}), \mathrm{proj}(\mathsf{Name}))\wedge \\ & & \mathrm{PrName}(\mathrm{proj}(\mathsf{Name}),\mathsf{Name}) \end{array}$$

- person and proj are constructors to create objects of the ontology from tuple of values from the database
- they are Skolem functions

Mappings could have varying expressivity, including negation, inequality,...

Ontology-Based Data Access: An Example

An interpretation is a model of an OBDA system if it is a model of Σ and satisfies M with respect to D.

worksIn

SSN	Name
100	AAA
200	BBB
300	CCC

SELECT SSN, Name FROM worksIn $\rightsquigarrow$ Researcher(person(SSN))$\wedge$ Project(proj(Name)) $\wedge$ worksFor(person(SSN), proj(Name))$\wedge$ PrName(proj(Name),Name)

Ontology-Based Data Access: An Example

An interpretation is a model of an OBDA system if it is a model of Σ and satisfies M with respect to D.

worksIn

SSN	Name
100	AAA
200	BBB
300	CCC

SELECT SSN, Name
FROM worksIn $\rightsquigarrow$ $\mathrm{Researcher}(\mathrm{person}(\mathsf{SSN})) \wedge$
$\mathrm{Project}(\mathrm{proj}(\mathsf{Name})) \wedge$
$\mathrm{worksFor}(\mathrm{person}(\mathsf{SSN}), \mathrm{proj}(\mathsf{Name})) \wedge$
$\mathrm{PrName}(\mathrm{proj}(\mathsf{Name}), \mathsf{Name})$

$$\mathrm{Researcher}(\mathrm{person}(100)) \wedge \mathrm{Project}(\mathrm{proj}(AAA)) \wedge$$
$$\mathrm{worksFor}(\mathrm{person}(100), \mathrm{proj}(AAA)) \wedge \mathrm{PrName}(\mathrm{proj}(AAA), AAA)$$

Ontology-Based Data Access: An Example

An interpretation is a model of an OBDA system if it is a model of Σ and satisfies M with respect to D.

worksIn

SSN	Name
100	AAA
200	BBB
300	CCC

SELECT SSN, Name FROM worksIn $\rightsquigarrow$ $\mathrm{Researcher}(\mathrm{person}(\mathsf{SSN})) \wedge \mathrm{Project}(\mathrm{proj}(\mathsf{Name})) \wedge \mathrm{worksFor}(\mathrm{person}(\mathsf{SSN}), \mathrm{proj}(\mathsf{Name})) \wedge \mathrm{PrName}(\mathrm{proj}(\mathsf{Name}), \mathsf{Name})$

$$\mathrm{Researcher}(\mathrm{person}(100)) \wedge \mathrm{Project}(\mathrm{proj}(AAA)) \wedge \mathrm{worksFor}(\mathrm{person}(100), \mathrm{proj}(AAA)) \wedge \mathrm{PrName}(\mathrm{proj}(AAA), AAA)$$

$$\mathrm{Researcher}(\mathrm{person}(200)) \wedge \mathrm{Project}(\mathrm{proj}(BBB)) \wedge \mathrm{worksFor}(\mathrm{person}(200), \mathrm{proj}(BBB)) \wedge \mathrm{PrName}(\mathrm{proj}(BBB), BBB)$$

Ontology-Based Data Access: An Example

An interpretation is a model of an OBDA system if it is a model of Σ and satisfies M with respect to D.

worksIn

SSN	Name
100	AAA
200	BBB
300	CCC

SELECT SSN, Name FROM worksIn $\rightsquigarrow$ $\mathrm{Researcher}(\mathrm{person}(\mathit{SSN}))\wedge \mathrm{Project}(\mathrm{proj}(\mathsf{Name}))\wedge \mathrm{worksFor}(\mathrm{person}(\mathsf{SSN}), \mathrm{proj}(\mathsf{Name}))\wedge \mathrm{PrName}(\mathrm{proj}(\mathsf{Name}),\mathsf{Name})$

$$\mathrm{Researcher}(\mathrm{person}(100)) \wedge \mathrm{Project}(\mathrm{proj}(AAA))\wedge \mathrm{worksFor}(\mathrm{person}(100), \mathrm{proj}(AAA)) \wedge \mathrm{PrName}(\mathrm{proj}(AAA), AAA)$$

$$\mathrm{Researcher}(\mathrm{person}(200)) \wedge \mathrm{Project}(\mathrm{proj}(BBB))\wedge \mathrm{worksFor}(\mathrm{person}(200), \mathrm{proj}(BBB)) \wedge \mathrm{PrName}(\mathrm{proj}(BBB), BBB)$$

$$\mathrm{Researcher}(\mathrm{person}(300)) \wedge \mathrm{Project}(\mathrm{proj}(CCC))\wedge \mathrm{worksFor}(\mathrm{person}(300), \mathrm{proj}(CCC)) \wedge \mathrm{PrName}(\mathrm{proj}(CCC), CCC)$$

Ontology-Based Data Access: An Example

A database is a model of an OBDA system if it is a model of Σ and satisfies M with respect to D.

$$\forall X(\mathrm{Researcher}(X) \rightarrow \exists Y(\mathrm{worksFor}(X, Y) \wedge \mathrm{Project}(Y)))$$
$$\forall X(\mathrm{Project}(X) \rightarrow \exists Y(\mathrm{worksFor}(Y, X) \wedge \mathrm{Researcher}(Y)))$$
$$\forall X, Y(\mathrm{worksFor}(X, Y) \rightarrow \mathrm{Researcher}(X) \wedge \mathrm{Project}(Y)$$
$$\forall X(\mathrm{Project}(\mathrm{X}) \rightarrow \exists Y(\mathrm{PrName}(X, Y)))$$

$$\mathrm{Researcher}(\mathrm{person}(100)) \wedge \mathrm{Project}(\mathrm{proj}(AAA)) \wedge$$
$$\mathrm{worksFor}(\mathrm{person}(100), \mathrm{proj}(AAA)) \wedge \mathrm{PrName}(\mathrm{proj}(AAA), AAA)$$
$$\mathrm{Researcher}(\mathrm{person}(200)) \wedge \mathrm{Project}(\mathrm{proj}(BBB)) \wedge$$
$$\mathrm{worksFor}(\mathrm{person}(200), \mathrm{proj}(BBB)) \wedge \mathrm{PrName}(\mathrm{proj}(BBB), BBB)$$
$$\mathrm{Researcher}(\mathrm{person}(300)) \wedge \mathrm{Project}(\mathrm{proj}(CCC)) \wedge$$
$$\mathrm{worksFor}(\mathrm{person}(300), \mathrm{proj}(CCC)) \wedge \mathrm{PrName}(\mathrm{proj}(CCC), CCC)$$

Semantics... again :)

Meaning of a query is again based on first-order semantics:

- based on interpretations, as last week;
- no closed-world assumption as in relational databases;
- notion of entailment: if $\mathcal{A}$ is an ABox, $\mathcal{T}$ is a TBox, and q is a Boolean query, then $\mathcal{A}, \mathcal{T} \models q$ if and only if every model of $\mathcal{A}$ and $\mathcal{T}$ is a model of q.

Influence of Ontology and Query Languages

What is the/a right ontology language?

- there is a wide spectrum of languages that differ in expressive power and computational complexity
- an important aspect is the scalability to large amounts of data

What is the/a right query language?

- "core" fragment of traditional database query languages?
- navigational components, such as can be seen in SPARQL?
- other practical features (aggregation...)?

Combined vs. Data Complexity

When considering OBQA, there are two classical ways of measuring complexity:

- combined complexity, where the database, the ontology and the query are considered part of the input;
- data complexity, where only the database is considered part of the input, whereas the ontology and the query are considered to be constant.

Take home message

- ontologies are used to structure human knowledge;
- interesting in their own right;
- hard to build, be it manually or (semi-)automatically;
- allows for formal reasoning, helpful for explainability.

Take home message

- ontologies are used to structure human knowledge;
- interesting in their own right;
- hard to build, be it manually or (semi-)automatically;
- allows for formal reasoning, helpful for explainability.

Optional course on:

- reasoning techniques for description logics
- challenges and techniques for ontology-based data access
- given by Camille Bourgaux and myself.