

Semantic Web

MPRI 2.26.2: Web Data Management

Antoine Amarilli

Friday, January 11th

Motivation

- Information on the Web is not structured

[List of joint winners of the **Hugo and Nebula** awards - Wikipedia, the ...](https://en.wikipedia.org/wiki/List_of_joint_winners_of_the_Hugo_and_Nebula_awards)
[en.wikipedia.org/.../List_of_joint_winners_of_the_Hugo_and_Nebul...](https://en.wikipedia.org/wiki/List_of_joint_winners_of_the_Hugo_and_Nebula_awards)

This is a list of the works that have won both the **Hugo Award** and the **Nebula Award**, **awarded** annually to works of science fiction literature. The **Hugo** Awards ...

Motivation

- Information on the Web is not **structured**

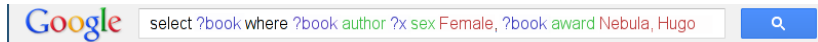
[List of joint winners of the **Hugo and Nebula** awards - Wikipedia, the ...](https://en.wikipedia.org/wiki/List_of_joint_winners_of_the_Hugo_and_Nebula_awards)
[en.wikipedia.org/.../List_of_joint_winners_of_the_Hugo_and_Nebul...](https://en.wikipedia.org/wiki/List_of_joint_winners_of_the_Hugo_and_Nebula_awards)

This is a list of the works that have won both the **Hugo Award** and the **Nebula Award**, **awarded** annually to works of science fiction literature. The **Hugo Awards** ...

- This makes it difficult to:
 - **Combine information** from multiple sources
 - **Integrate** different services
 - **Reason** with Web data

Goal

What if we could write:



Goal

What if we could write:



```
SELECT ?book ?bookLabel WHERE {  
  ?book wdt:P166/(wdt:P361*|wdt:P31)  
         wd:Q194285, wd:Q188914 .  
  ?book wdt:P50/wdt:P21 wd:Q6581072 .  
  SERVICE wikibase:label { bd:serviceParam  
    wikibase:language "[AUTO_LANGUAGE],en" . }  
}
```

Applications

- **Most visible application** today: rich search results
 - **Bing** <https://www.bing.com/webmaster/help/marking-up-your-site-with-structured-data-3a93e731>
 - **Google Search** <https://developers.google.com/search/docs/guides/search-gallery>
 - **Yandex Search**
<https://yandex.ru/support/webmaster/site-content/data-transmit.html?ncrnd=4299&lang=en>
- **Also:** SPARQL endpoints, e.g.,
<https://query.wikimedia.org/>
- **Indirectly:** Graph databases

Key concepts

- **Semantic Web:** put **structured data** on the Web
- **Entities:** the things about which we want to express data
- **Ontology:** a **vocabulary** to talk about entities, specifying **relations, classes**, etc.
- **Knowledge base:** a set of assertions about entities expressed following an ontology
- **Linked data:** use URIs to create **links** between datasets
- **Information extraction:** creating structured information out of existing Web Data (later)

Resources (or entities)

- A resource is anything that can be referred to by a **URI**
 - a **web page**, identified by a URL
 - a **fragment of an XML document**, identified by an element node of the document,
 - a **web service**,
 - an **identifier**, e.g., an ISBN,
 - a thing, an object, a concept, a property, etc.
- The URI does not need to be **dereferenceable**
- A **cool URI** is one that can be dereferenced to obtain information about the entity
 - `https://www.wikidata.org/wiki/Q42`
 - `https://wiki.openstreetmap.org/wiki/Key:opening_hours`

Ontologies

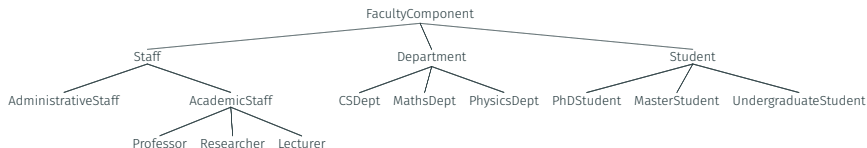
- Formal descriptions providing **human** users a shared understanding of a given domain
 - A controlled vocabulary
- Formally defined so that it can also be processed by **machines**
- **Logical semantics** that can be useful for **reasoning**
 - to answer queries (over possibly distributed data)
 - to relate objects in different data sources and integrate them
 - to detect inconsistencies or redundancies
 - to refine queries with too many answers
 - to relax queries with no answer

Where Do Ontologies Come From?

- **Manually crafted** to represent the knowledge of a specific domain (e.g., life sciences)
- Exported from **classical Web databases**
- Created **collaboratively** (e.g., Wikidata)
- **Private** to a company or **public**
- Value of the Semantic Web: bits of ontologies can be **re-used** in another, and ontologies can be mapped with **owl:sameAs**, **owl:equivalentClass**, **owl:equivalentProperty**, etc.

Classes and class hierarchy

- A **class** denotes a set of entities; e.g., **Professor**, **Country**, **Person**
- An entity can be an **instance** of one or several classes
- Ex: MPRI **instanceOf** MasterProgram
- Ex: AcademicStaff **subClassOf** Staff (interpreted as set inclusion)



Relations

- A **relation** denotes a binary relation between objects; e.g., **locatedIn**, **father**
- Often convenient to express with a **signature** (classes for the subject and object)
- **TeachesIn**(AcademicStaff, Course)
 - if one states that “X **TeachesIn** Y”, then X belongs to AcademicStaff and Y to Course
- **TeachesTo**(AcademicStaff, Student)
- **Leads**(Staff, Department)

Namespaces

- Using full URLs for URIs is often **cumbersome**
 - `https://www.wikidata.org/entity/Q42`
- **Idea**: declare a **namespace** like `wd` to stand for `https://www.wikidata.org/entity/`
- Then you can write a **Compact URI** (aka CURIE) `wd:Q42` to mean `https://www.wikidata.org/entity/Q42`
- This is like a simplification of **XML namespaces**
- Also a **default namespace**, to write `:Q42`
- In the slides I will just write entities like “MPRI” and ignore the issue

Ontologies and knowledge bases

To summarize:

- **Ontology** puts the focus on the **schema**, or TBox
 - The set of class and relation names (= the **vocabulary**)
 - The **signatures** of relations and also **constraints**
 - The constraints can be used to:
 - check data consistency (like dependencies in databases)
 - infer new facts
- **Knowledge base** puts the focus on the **instance**, or ABox
 - **Entities** (instances of a class), and **facts** about them (see next)
- Many **knowledge bases** provide their own **ontology** (but may also use terms from other ontologies)

Ontology Languages for the Web

3 ontology languages for the Web

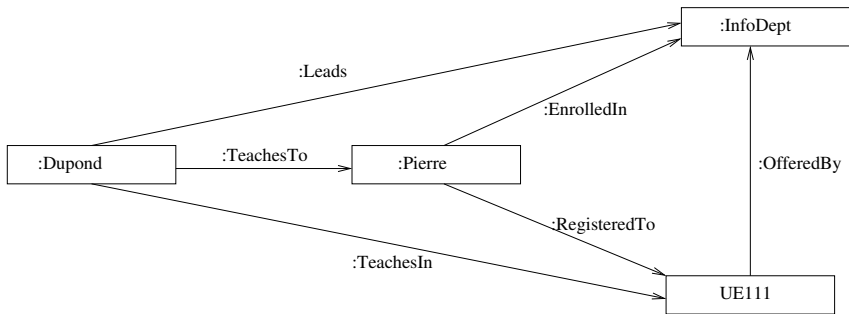
- **RDF**: not really an ontology language (only ABox facts)
- **RDFS**: schema for RDF, but very basic
- **OWL**: a much richer ontology language

RDF: Resource Description Framework

- RDF facts are **triplets**
- Each triplet is of the form: $\langle \text{Subject Predicate Object} \rangle$
- The subject is a URI, referencing an **entity**
(from the same KB or a different one)
- The predicate is a URI, referencing a **relation**
(from some ontology)
- The object is either a URI, referencing an **entity**, or a **literal**
 - $\langle \text{:Dupond :Leads :CSDept} \rangle$
 - $\langle \text{:Dupond :HasName "Paul Dupond"} \rangle$
 - $\langle \text{:Dupond :TeachesIn :UE111} \rangle$
 - $\langle \text{:Dupond :TeachesTo :Pierre} \rangle$
 - $\langle \text{:Pierre :EnrolledIn :CSDept} \rangle$
 - $\langle \text{:Pierre :RegisteredTo :UE111} \rangle$
 - $\langle \text{:UE111 :OfferedBy :CSDept} \rangle$
- The linked data cloud contains **dozens of billions** of RDF triples

RDF graph

- A set of RDF facts defines
 - a set of relations between objects
 - an **RDF graph** where the nodes are objects:



- A triplet $\langle s \ P \ o \rangle$ is interpreted in first-order logic (FOL) as a fact $P(s, o)$
- Example:
 - Leads(Dupond, CSDept)
 - TeachesIn(Dupond, UE111)
 - TeachesTo(Dupond, Pierre)
 - EnrolledIn(Pierre, CSDept)
 - RegisteredTo(Pierre, UE111)
 - OfferedBy(UE111, CSDept)

Literals

- Sometimes the **object** of a statement is a literal value, e.g., a name, number, date
- Literals can come with a **language**, i.e., an ISO language name
`"France"@en`
- They can have a **data type**, which is just a URI
`"28753"^^<http://www.w3.org/2001/XMLSchema#nonNegativeInteger>`
`"6.96E10"^^<http://dbpedia.org/datatype/euro>`
- Default data type, and default for literals with a language code
- The data type also indicates how the value is **interpreted**, how **comparisons** work, etc.

RDF Serialization

Several serialization formats for RDF data, see alternate formats of http://live.dbpedia.org/page/%C3%89lectricit%C3%A9_de_France:

- **RDF/XML**, structured XML representation, allowing for nesting
- **N-Triples**, **Turtle**, **N3**, text-based formats
- **RDFa** and **JSON-LD** to integrate RDF annotations into HTML

N-Triples

Simplest text-based format

- Each **fact** is a triple of a subject, predicate, object, followed by a full stop.

```
<http://live.dbpedia.org/resource/\u00C9lectricit\u00E9_de_France>  
  <http://live.dbpedia.org/property/netIncome>  
  "3.2E9"^^<http://dbpedia.org/datatype/euro> .
```

- Possibility of **comments**
- Support for **literals** with **language** and **datatype**
- All usual questions of quoting, escaping, character encodings...

Turtle

Turtle is a superset of N-Triples

- Adds **prefixes** (for CURIEs) and **default prefix**

```
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
```

- Adds **factoring** of common subjects, common predicates

```
:s1 :p1 :o1, :o2, :o3 ;
```

```
    :p2 :o4 .
```

```
:s2 :p3 :o5 .
```

- Adds a alias for the type relation

Turtle (cont'd) and N3

- Adds **square brackets** for blank nodes (see later)

```
ex:coursenotes ex:author [  
  foaf:name "Amarilli" ; foaf:givenname "Antoine"  
] .
```

stands for

```
ex:coursenotes ex:author []:b1 .  
[]:b1 foaf:name "Amarilli" ;  
      foaf:givenname "Antoine" .
```

- Adds **brackets** for **linked lists**

Notation3 (N3) is a superset of Turtle with additional features for semantic assertions (not just RDF data).

Blank nodes

- Some entities may be **blank nodes**, i.e., nodes with no URI.
- They are written `_:xyz` with `xyz` being a **local identifier**
- Common usage: *n*-ary relations

```
ex:speaker ex:gaveSeminar _:seminar .
```

```
_:seminar ex:date "2019-01-11"^^xsd:date .
```

```
_:seminar ex:room ex:Room101 .
```

```
_:seminar ex:title "Example seminar title"@en .
```

Reification

Another common usage of blank nodes is reification:

```
dbp:Earth dbprop:creationDate "-4003-10-23"^^xsd:date
```

can be written as

```
[ ]:stmt rdf:type rdf:Statement ;  
        rdf:subject dbp:Earth ;  
        rdf:predicate dbprop:creationDate ;  
        rdf:object "-4003-10-23"^^xsd:date .
```

allowing us to say, e.g.,

```
[ ]:stmt dbp:author dbp:James_Ussher .
```

RDF built-ins

- Some **classes**, i.e., literals, properties, statements, etc.
 - Some **datatypes**, i.e., unordered lists, ordered lists, bags,
 - **rdf:type** to say that something is an instance of a class
- To say more about the **schema**, we need **RDF Schema**

RDF Schema (RDFS)

- **RDF Schema** is a language to describe the schema of RDF documents
- Do not get confused: RDFS can use RDF as syntax, i.e., RDFS statements are themselves expressed as RDF triplets with some specific **properties** and **objects**
- Declaration of classes and subclass relationships
 - `< Staff rdf:type rdfs:Class >`
 - `< JavaCourse rdfs:subClassOf CSCourse >`
- Declaration of instances
 - `< Dupond rdf:type AcademicStaff >`

RDF Schema - continued

- Declaration of relations (properties in RDFS terminology)
 - `< RegisteredTo rdfs:type rdfs:Property >`
- Declaration of subproperty relationships
 - `< LateRegisteredTo rdfs:subPropertyOf RegisteredTo >`
- Declaration of domain and range for predicates (used for inference)
 - `< TeachesIn rdfs:domain AcademicStaff >`
 - `< TeachesIn rdfs:range Course >`
 - `TeachesIn(AcademicStaff, Course)`

RDFS logical semantics

RDF and RDFS statements	FOL translation	DL notation
$\langle i \text{ rdf:type } C \rangle$	$C(i)$	$i : C$ or $C(i)$
$\langle i P j \rangle$	$P(i, j)$	$i P j$ or $P(i, j)$
$\langle C \text{ rdfs:subClassOf } D \rangle$	$\forall X (C(X) \Rightarrow D(X))$	$C \sqsubseteq D$
$\langle P \text{ rdfs:subPropertyOf } R \rangle$	$\forall X \forall Y (P(X, Y) \Rightarrow R(X, Y))$	$P \sqsubseteq R$
$\langle P \text{ rdfs:domain } C \rangle$	$\forall X \forall Y (P(X, Y) \Rightarrow C(X))$	$\exists P \sqsubseteq C$
$\langle P \text{ rdfs:range } D \rangle$	$\forall X \forall Y (P(X, Y) \Rightarrow D(Y))$	$\exists P^- \sqsubseteq D$

DL: Description logics, a specialized logical formalism

OWL: Web Ontology Language

- OWL extends RDFS to express **richer constraints**
- Main OWL constructs
 - **Disjointness** between classes
 - Constraints of **functionality** and **symmetry** on predicates
 - Class **union** and **intersection**
- Inspired by **description logics**
- Several **profiles**: OWL Full, OWL DL, OWL Lite, OWL 2 EL, OWL 2 QL, OWL 2 RL.
- Different profiles include different **features**, and have different **(in)tractability**

Slide acknowledgements

- Material reused from Pierre Senellart's class <http://pierre.senellart.com/talks/romatre-20130923.pdf> itself adapted from [Web data management](#), S. Abiteboul, I. Manolescu, P. Rigaux, M.-C. Rousset, P. Senellart, Cambridge University Press, 2012. Also available at <http://webdam.inria.fr/Jorge/>