Biomedical semantics, information retrieval and knowledge discovery - (8)

Resource Description Framework RDF

Dr. Dietrich Rebholz-Schuhmann
Dr. Leyla Jael Garcia Castro

December 19, 2020

Outline

Objectives and Learning outcomes

Objectives

- We will introduce RDF and name some advantages and disadvantages, all of it illustrated by practical examples

Learning outcomes

- Stating what the Resource Description Framework is about
- Explaining the main elements in RDF
- Illustrating via diagrams the sort of statements that can be modelled with RDF
- Naming the different serializations for RDF graphs
- Naming different ways to syntactically validate RDF data
- Using RDF-XML serialization (and Turtle) to represent knowledge
- Explaining how graph representations can be separated into RDF triples

Resource Description Framework RDF

Introduction to RDF

- Unifed data formats are great . . . but how do we combine data from several resources?
- The data format must be more general, XML is not good enough, i.e. adjustments and extensions in XML are too difficult due to the nested structure of tags
- One option would be to use a very simple data format, a format that represents single facts
- Statements ('assertion') should communicate with/relate to all other assertions
- Resources should be structured in a way that it can be decomposed into its components which then can be reused
- RDF: Resource Description Framework — standard model for data interchange on the Web

Resource Description Framework

- RDF (the standard model) combines three main specifications: RDF (the controlled vocabulary), RDFS and XSD
- RDF (vocabulary) mainly provides a way to describe resources via statements
- The combination of RDF+RDFS+XSD supports the definition of controlled vocabularies
- RDF and RDFS support each other to define and describe their elements

Namespace prefix	Namespace URI	RDF vocabulary
rdf	`http://www.w3.org/1999/02/22-rdf-syntax-ns#`	The RDF built-in vocabulary
rdfs	`http://www.w3.org/2000/01/rdf-schema#`	The RDF Schema vocabulary
xsd	`http://www.w3.org/2001/XMLSchema#`	The RDF-compatible XSD types

From XML to RDF: in a formal way

- RDF can be expressed as XML, i.e. the syntax is the same
- By the moment we forget about the graph structure (RDFS, classes) and only generate a set, i.e. a collection of statements (RDF)
- Elements are represented by URIs, i.e. all resources are defined via/as Web resources
- The namespace serves as one of the key elements, and this principle is expanded to improve readability with the help of prefixes
- For all the characteristics and all the links between involved elements, we use the same unified schema:
  - Subject - Predicate - Object (S - P - O)
  - S and O can be an rdfs:Resource, an rdfs:Literal or a blank node
  - P this is an rdfs:Resource, which denotes the relation between S and O

Naming resources: qualified URIs

We have seen URIs and namespaces before. When identifying an rdfs:Resource we have mainly two options

```
{
  http://purl.uniprot.org/uniprot/P05067
  http://www.w3.org/2001/XMLSchema#boolean
}
```

- Both are meant to identify a resource for public use
- We can use namespaces with any of them
- Any can be used to denote a node (S or O) or an edge (P)
- A qualified URI has a fragment #

Types of statements

RDF triples denote statements about two nodes (entities/concepts) What do we want to express with them?

- Statements as data elements:
  - Each data element can be identified in a unique way (by its nodes and the labeled edge).
  - URIs can be used to denote nodes and edges in a unique way.
- Apart from naming the data elements, they require a description (characterization of their properties):
  - How big, how long, how heavy, . . .
  - Where does it come from, where does it belong, . . . ?
- Which characteristics do exist between data elements?
  - How are they related?
  - Do some statements apply in a universal way to data elements?

Facts can be described in RDF statements (Triples)

Simple Facts	John (S) - loves (P) - Mary (O) . Car (S) - hasColor (P) - red (O) .
Database statements	x1 - isSmallerThan - x2 . x1 - hasValue - 0.5 . x2 - isSmallerThan - x3 .
Facts from knowledge bases	Cat - isA - Carnivore . Carnivore - isA - Mammal .
Distributed information	BlueEye - isUser - Facebook . FreakEye - isUser - Google . FreakEye - isEquivalent - BlueEye .

Each entry used for S, P or O should be represented as URI, but literals are allowed as well (e.g., "Name"^^xs:string).

Triples: S - P - O

The format: S(ubject) P(redicate) O(bject) .
In short: S - P - O is the data standard for the Resource Description Framework. — It complies with other standards such as Extended Markup Language (XML), XML schema, Web Ontology Language (OWL) and others.

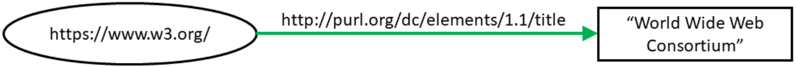
- All data can be transformed into a graph but not showing anymore an explicit root element (as in XML)
- The order of triple statements is not relevant and can be changed to any order
- In principle, the triple statements can be contained in a single file, or in several files, or even in several files on several servers, and data redundancy (duplication) is resolved automatically
- Due to the use of URIs, all data at all sites can be combined through the World Wide Web.

First example in RDF

This example uses RDF/XML syntax to represent the triple statement and gives the title of the Word Wide Web Consortium to the Web page. Below is the triple statement and the graph representation.

```
<?xml version="1.0"?>
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:dc="http://purl.org/dc/elements/1.1/">
  <rdf:Description rdf:about="http://www.w3.org/">
    <dc:title>World Wide Web Consortium</dc:title>
  </rdf:Description>
</rdf:RDF>
```

Number	Subject	Predicate	Object
1	https://www.w3.org/	http://purl.org/dc/elements/1.1/title	“World Wide Web Consortium”



Different notations: RDF/XML

- Standard XML notation
- all URIs used within the elements
- `rdf:type` and `rdfs:label` denote the two triples relating to the definition of Toy.

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">

  <rdf:Description rdf:about="http://www.w3.org/2002/07/owl#Toy">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Thing"/>
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
      Toy
    </rdfs:label>
  </rdf:Description>

</rdf:RDF>
```

Different notations: RDF/XML ABBREV

- Shortened XML notation
- `rdfs:label` denoted explicitly and `rdf:type` implicitly

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xmlns:owl="http://www.w3.org/2002/07/owl">

  <owl:Thing rdf:about="http://www.w3.org/2002/07/owl#Toy">
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
      Toy
    </rdfs:label>
  </owl:Thing>

</rdf:RDF>
```

Different notations: N-Triples

- Representation as two separated triples (one per statement: type and label)
- No use of namespace identifiers

```
<http://www.w3.org/2002/07/owl#Toy>
  <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
    <http://www.w3.org/2002/07/owl#Thing> .

<http://www.w3.org/2002/07/owl#Toy>
  <http://www.w3.org/2000/01/rdf-schema#label>
    "Toy"^^<http://www.w3.org/2001/XMLSchema#string> .
```

Different notations: Turtle

- Namespaces use a different syntax `@prefix namespace: <URI>`
- Some bits similar to RDF/XML ABBREV

```
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

owl:Toy
  a owl:Thing ;
  rdfs:label "Toy"^^xsd:string .
```

Different notations: JSON-LD

- JSON syntax supporting Linked Data
- Various flavors (compact, extended, flattened)
- Namespaces can be added via a `@context`

```
{
  "@id": "http://www.w3.org/2002/07/owl#Toy",
  "@type": "http://www.w3.org/2002/07/owl#Thing",
  "http://www.w3.org/2000/01/rdf-schema#label": "Toy"
}
```

```
{
  "@context" : [
    {"owl" : "http://www.w3.org/2002/07/owl#"},
    {"rdfs": "http://www.w3.org/2000/01/rdf-schema#"}
  ],
  "@id":"owl:Toy",
  "@type":"owl:Thing",
  "rdfs:label": "Toy"
}
```

RDF Syntax Validators

RDF validators are, for example, provided by the W3C:

`http://www.w3.org/RDF/Validator/`

For turtle (a very efficient, i.e. less verbose language representation), the following one should do:

`http://www.rdfabout.com/demo/validator/`

For JSON-LD (a JSON-based RDF representation), the following can be used:

`https://json-ld.org/playground/`

Different notations: From one to another

There are some converter tools out there:

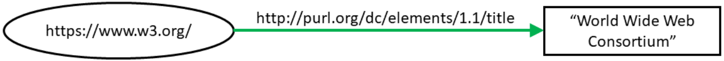
- `https://www.easyrdf.org/converter`
- `https://rdf-translator.appspot.com/`

Sometimes they mess up with namespaces and they will not always give you what you would expect for as there are different ways to go (e.g. "Goethe - wrote - Prometheus" and "Prometheus - was written by - Goethe" convey the same information)

RDF triples

S - P - O in RDF are rdfs:Resource

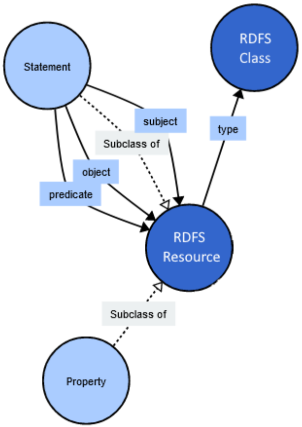
Number	Subject	Predicate	Object
1	https://www.w3.org/	http://purl.org/dc/elements/1.1/title	“World Wide Web Consortium”



but they play different roles and have different characteristics

- Subjects are commonly Classes (or individuals/instances)
- Predicates are Properties, all predicate possible subjects are its domain and all predicate possible objects are its range
- Objects are commonly Classes (or individuals) or Literals or Datatypes
- The triple S - P - O is a Statement

rdf:Statement



Statements convey some truth which can be validated or tested, i.e. the statement is true or false but nothing in between

- Statements are resources which are represented as a triple
- Statements have 3 parts: `rdf:subject`, `rdf:predicate`, `rdf:object`
- For instance "my car - is - red"
- Statements are specified in RDF

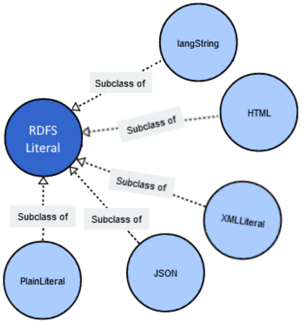
rdf:Property

Properties are important to determine the relationship between concepts and/or give them specific characteristics

- Properties are resources identified by a URI
- Properties are an rdfs:Class
- Properties are specified in RDF

```
<rdfs:Class rdf:about="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property">
  <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
    The class of RDF properties.
  </rdfs:comment>
  <rdfs:isDefinedBy rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#"/>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Property</rdfs:label>
  <rdfs:subClassOf rdf:resource="http://www.w3.org/2000/01/rdf-schema#Resource"/>
</rdfs:Class>
```

rdfs:Literal



RDF foresees that we use literals, we do not have to conceptualize them

- Any type of text is a literal as well as dates "3rd of October of 2020" and numbers "3.1415"
- A literal is a construct to specify data without specifying a resource
- Literal are combined with datatypes, commonly specified on the XSD vocabulary
- Literals just like constants in a computer program often have a very restricted importance

```
<rdfs:Class rdf:about="http://www.w3.org/2000/01/rdf-schema#Literal">
  <rdfs:isDefinedBy rdf:resource="http://www.w3.org/2000/01/rdf-schema#"/>
  <rdfs:label>Literal</rdfs:label>
  <rdfs:comment>The class of literal values, eg. textual strings and integers.</rdfs:comment>
  <rdfs:subClassOf rdf:resource="http://www.w3.org/2000/01/rdf-schema#Resource"/>
</rdfs:Class>
```

Summary

RDBS Schema vs. Vocabularies

This table shows the distinction between the use of a schema and a vocabulary for representing semantics, i.e. the schema denotes the way how data is kept in the XML/RDF file and the alternative, a 'Vocabulary', provide its content from a single file and would require metadata information to use it correctly

Type	(RDB) Schema	Vocabulary
Function	specified columns	specified properties
Data type	mandatory	not necessary
Use as resource	rather not	mandatory
Standard	wanted	anticipated
Metadata	not required	mandatory

Summary

- RDF triples offer a simple and basic syntax to represent facts
- The whole framework of RDF, XML and OWL contribute to the standardization of the data across the internet
- Facts from Wikipedia can be represented "easily" in RDF syntax and verified, if the represented facts follow the predefined standards
- Any concept can be produced as a URI. Once it has been generated, it has to be well specified either in relation to other concepts (Classes, Properties), or the lacking information has to be added
- We can use this approach to (1) represent our own data, and (2) generate and profit from large-scale external and public resources
- Caveat: producing world knowledge in electronic resources is hard work and time-consuming.

So where are we now?

- With regards to biomedical semantics, we should now be able to "read" simple vocabularies expressed in RDF, identify triples and depict them as a graph
- With regards to data integration we now understand how to use RDF triples to model interoperable and connected data
- RDF offers a vocabulary related to syntactic elements `rdf` together with an extension enhancing the semantics support `rdfs` (more of it in the next lesson)
- There are different flavors but all of them offering similar support for RDF
- With regards to knowledge discovery we are moving towards stronger semantics that will allow us to infer some facts from our data (modelled using RDF principles)