RDF(S)

Resource Description Framework (Schema)
Where are we?

- DL Extensions
- Scalability
- OWL Reasoning
- OWL in practice
- RDF(S)
- Practical Topics
- XML
- PL/FOL
- OWL
- INTRODUCTION :: FUNDAMENTALS :: SYNTAX :: CAPABILITIES :: RDFSUMMARY :: RDFS :: RDFS SUMMARY
Where are we?

- PL, FOL, XML
- Today: RDF
Where are we?
Where are we?

- **XML:**
  - interchange syntax, no semantics

- **RDF:**
  - data model, some semantics & inference (recent!)

- **RDF Schema:**
  - concept modeling, more semantics & inference

- **OWL:**
  - more expressive ontology language
  - quite expressive
  - expensive inference
Task

Find five of the following:

“All soccer players, who played as goalkeeper for a club that has a stadium with more than 40,000 seats and who are born in a country with more than 10 million inhabitants.“
Roadmap for Today

- RDF
  - Introduction
  - Syntax
  - Collections
  - Reification
- RDF Schema
Introduction: What is RDF

- RDF is a data model
  - used in representing information about resources in the World Wide Web (WWW)
  - can be seen as directed graph with labeled nodes and arcs or as an object-oriented model (object/attribute/value)

![Graph Diagram]
http://semanticmatching.org/semantic-matching.html

author

Fausto Giunchiglia
Introduction: What is RDF

- domain, and application independent

- goal is to make information available for applications to process, rather than only display to the human beings

- is based on the idea of identifying things using Web identifiers (i.e., Uniform Resource Identifiers, or URIs)

- RDF data model is an abstract, conceptual layer
Fundamentals of RDF

- Three fundamental concepts in RDF are:
  - Resources
  - Properties
  - Statements
Fundamentals of RDF

Resources

- Resource can be considered as an object, a “thing”, we want to talk about
  - For example, web page, books, authors, publishers, people, organizations, places, etc.

- Each resource has a URI (i.e., Universal Resource Identifier)

- A URI can be
  - a URL (Web address) or
  - some other kind of unique identifier
Fundamentals of RDF

Properties

- Properties are a special kind of resources
- They describe relations between resources
  - For example, “author”, “publisher”, “hasStudent”, “teach”, “age”, “title”, “name”, “locatedIn”, etc.
- Properties are also identified by URIs
  - Advantages of using URIs:
    - A global, worldwide, unique naming scheme
    - Reduces the homonym (e.g., title) problem of distributed data representation
Fundamentals of RDF

**Resource**

http://semanticmatching.org/semantic-matching.html

**Property**

http://www.disi.unitn.it/terms/author

**Value**

Fausto Giunchiglia

Statement

"http://semanticmatching.org/semantic-matching.html" has "http://www.disi.unitn.it/terms/author" Fausto Giunchiglia

**Important:** value can be another resource or literals (e.g., character strings such as "Fausto Giunchiglia", and values from other data types such as integers and dates, as the values of properties)
Fundamentals of RDF

Statements

- RDF statements consist of resources (= nodes) which have properties which have values (= nodes, strings) = subject = predicate = object

- Statements assert the properties of resources
  - consisting of a resource, a property, and a value
Fundamentals of RDF

Three views of a RDF statement

- A triple
- A piece of a graph
- A piece of XML code

Hence, a RDF document can be seen as,

- A set of triples
- A graph (semantic net)
- An XML document
Fundamentals of RDF

Statements as Triples


- **Triple** \((x, P, y)\) can be considered as a logical formula \(P(x, y)\)
  - **Binary predicate** \(P\) relates object \(x\) to object \(y\)
Fundamentals of RDF

A Set of Triples as a Semantic Net

http://semanticmatching.org/semantic-matching.html

Fausto Giunchiglia

F. D. Natale

DISI
Fundamentals of RDF

Statement in XML

```xml
<?xml version="1.0"?>
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:disi-voc="http://www.disi.unitn.it/terms/">
    <rdf:Description
        rdf:about="http://semanticmatching.org/semantic-matching.html">
        <disi-voc:author>Fausto Giunchiglia</disi-voc:author>
    </rdf:Description>
</rdf:RDF>
```
RDF Syntax

- The RDF graphs are useful tool for human understanding while
- The Semantic Web (SW) vision requires “machine accessible” and “machine processable” representations
- RDF uses eXtensible Markup Language (XML) where XML is used as a transfer syntax for RDF
  - Important: XML is not a part of the RDF data model!
- RDF provides only binary predicates (properties)
  - E.g., P(x,y), here, binary predicate P relates object x to object y
RDF/XML

- An RDF document is represented by an XML element with the tag `rdf:RDF`
  - The content of this `rdf:RDF` element is a number of descriptions, which use `rdf:Description` tags.
  
- The `rdf:Description` element makes a statement about the resource http://www.semanticmatching.org/semantic-matching.html
  
- Within the description
  
  - the property “disi-voc:author” is used as a tag
    
    - the content “http://www.disi.unitn.it/teachers/FaustoGiunchiglia” is the value of the property “disi-voc:author”
RDF/XML

- Every description makes a statement about a resource, identified in 3 ways:
  - an `about` attribute, referencing an existing resource
  - an `ID` attribute, creating a new resource
  - without a name, creating an anonymous resource
An element `rdf:Description` has

- an `rdf:about` attribute indicates that the resource has been “defined” elsewhere
  - Assigns an absolute identifier in general

- An `rdf:ID` attribute indicates that the resource is defined
  - Assigns a fragment identifier (relative URIref)

Sometimes it is good (for **better organization** and **human readability**) to have things defined in one location, while other location state “additional” properties
Task

Model our Simpsons family in RDF:

1) Use a graph-like notation
2) Use one possible XML serialization
Data Types

- Unlike typical programming languages and database systems, RDF has **no** built-in set of data types of its own (e.g., integers, strings, dates)

- Basic XML Schema datatypes such as `xsd:string`, `xsd:boolean`, `xsd:time`, `xsd:date`, etc. are **suitable** for use in RDF
  - **Important**: some of the built-in XML Schema datatypes are **not** suitable for use in RDF (e.g., `xsd:duration`)

- RDF provides **no mechanism** for defining new datatypes

- But the use of **any** externally defined data typing scheme is allowed in RDF documents
Data Types

<?xml version="1.0"?>

<!DOCTYPE rdf:RDF [<!ENTITY xsd "http://www.w3.org/2001/XMLSchema#">]>

<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:disi-voc="http://www.disi.unitn.it/terms/">

    <rdf:Description rdf:about="http://www.semanticmatching.org/semantic-matching.html">
        <disi-voc:author>Fausto Giunchiglia</disi-voc:author>
        <disi-voc:title>Professor</disi-voc:title>
        <disi-voc:age rdf:datatype="&xsd:integer">55</disi-voc:age>
    </rdf:Description>

</rdf:RDF>

attribute rdf:datatype="&xsd:integer", a typed literal is used to indicate the datatype of the value of the property “age”
Similar to the programming languages, concept of objects having different types or classes, RDF also supports this concept by providing a predefined property, rdf:type.

When an RDF resource is described with an rdf:type property, the value of that property is considered to be a resource that represents a category or class of things, and the subject of that property is considered to be an instance of that category or class.
<!DOCTYPE rdf:RDF [<!ENTITY xsd "http://www.w3.org/2001/XMLSchema#">]>
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:disi-voc="http://www.disi.unitn.it/terms/>

<rdf:Description rdf:ID="ICT001">
    <rdf:type rdf:resource="http://www.disi.unitn.it/course"/>
    <disi-voc:courseName>LDKR</disi-voc:courseName>
    <disi-voc:isTaughtBy rdf:resource="DISI111"/>
</rdf:Description>

<rdf:Description rdf:ID="DISI111">
    <rdf:type rdf:resource="http://www.disi.unitn.it/lecurer"/>
    <disi-voc:name>Fausto Giunchiglia</disi-voc:name>
    <disi-voc:title>Professor</disi-voc:title>
    <disi-voc:age rdf:datatype="&xsd:integer">55</disi-voc:age>
</rdf:Description>
</rdf:RDF>
Task

Model the following in RDF:

The lecture „Text Analytics“ takes place twice a week (Thursday and Friday) in two different rooms (1‘306 and 2‘303).
Blank Node

RDF/XML allows representation of graphs that include nodes without any URIrefs, i.e., the blank nodes
Blank Node: RDF/XML

```xml
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:dc="http://purl.org/dc/elements/1.1/"
    xmlns:disi-voc="http://www.disi.unitn.it/terms/">

  <rdf:Description rdf:about="http://www.semanticmatching.org/semantic-matching.html">
    <dc:title>S-Match – semantic matching</dc:title>
    <dc:creator rdf:nodelD="abc"/>
  </rdf:Description>

  <rdf:Description rdf:nodelD="abc">
    <disi-voc:name>Fausto Giunchiglia</disi-voc:name>
    <disi-voc:homePage rdf:resource="http://disi.unitn.it/~fausto"/>
  </rdf:Description>

</rdf:RDF>
```
Blank nodes are truly evil!

Why?
Task

Extend our Simpsons family example (in RDF/XML) to use:
- Blank nodes
- RDF type
Container

- A container is a resource that contains things
- Allows grouping of resources (including blank nodes) or literals values
  - about which we want to make statements as a whole
- The contained things are called members
- A typical use of a container is to indicate that the value of a property is a group of things
  - For example, we may wish to talk about a list of students taking a particular course, or, we may wish to talk about a list of courses offered by a particular lecturer, and so on.
Container

- The content of container elements (i.e., members) are named `rdf:_1, rdf:_2, etc.
- Alternatively `rdf:li`
- **Important**: RDF/XML provides `rdf:li` as a convenience element to avoid having to explicitly number each membership property
Container

RDF defines three types of containers:

- **rdf:Bag** an unordered container
  - E.g. members of the university library board, documents in a folder

- **rdf:Seq** an ordered container
  - E.g. modules of a course, items on an agenda, an alphabetized list of staff members (order is imposed)

- **rdf:Alt** a set of alternatives
  - E.g., alternative (language) translations for the title of a book, or describing a list of alternative Internet sites at which a resource might be found,
  - **Important**: an application using a property whose value is an Alt container should be aware that it can choose any one of the members of the group as appropriate

- **Important**: describing a resource as being one of these types of containers, the resource is given an rdf:type property whose value is one of the predefined resources rdf:Bag, rdf:Seq, or rdf:Alt (whichever is appropriate)
Container

Triples
{http://disi.unitn.it/members/library/board_members,mem:libraryBoardMembers, x}
{x, rdf:_1, “J. Kaiser”}
{x, rdf:_2, “V. Verma”}
{x, rdf:_3, “J. Sarkhel”}
{x, rdf:_4, “S. Sukla”}
{x, rdf:_5, “D. Madalli”}
{x, rdf:type, rdf:bag}
Task

Create a RDF/XML serialization of the previous RDF graph!
Container (Bag): RDF/XML

```xml
<rdf:RDF
xmlns:mem="http://www.disi.unitn.it/members/vocabularies/">

  <rdf:Description rdf:about="http://disi.unitn.it/members/library/board_members">
    <mem:libraryBoardMembers>
      <rdf:Bag>
        <rdf:li> J. Kaiser </rdf:li>
        <rdf:li> V. Verma </rdf:li>
        <rdf:li> J. Sarkhel </rdf:li>
        <rdf:li> S. Sukla </rdf:li>
        <rdf:li> D. Madalli </rdf:li>
      </rdf:Bag>
    </mem:libraryBoardMembers>
  </rdf:Description>

</rdf:RDF>
```

**Important:** RDF/XML provides syntactic shorthand, similar like HTML lists
RDF Collections

- Limitation of those containers is that there is no way to close them
- E.g., “these are all the members of the container”
- There is no mechanism enforcing the unique value constraints
- RDF provides support for describing groups containing only the specified members, in the form of RDF collections
- List structure in the RDF graph constructed using a predefined collection vocabulary: rdf:List, rdf:first, rdf:rest and rdf:nil
RDF Collections

http://disi.unitn.it/members/library/board_members

- D. Madalli
- J. Sarkhel
- J. Kaiser
Reification

- In RDF it is possible to make statements about statements
- Such statement can be used in building trust
- Can be referred as provenance information (like, who made, where, when made)
- Important: solution is to assign a unique identifier to each statement
Reification: RDF/XML

```xml
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:dc="http://purl.org/dc/elements/1.1/"
    xmlns:mem="http://www.disi.unitn.it/members/vocabulary/">
    <rdf:description rdf:about="http://www.disi.unitn.it/members/library/board_members">
        <mem:libraryBoardMembers>D. Madalli</mem:libraryBoardMembers>
    </rdf:description>
    
    <rdf:Statement rdf:about="http://disi.unitn.it/members/library#triple12345">
        <rdf:predicate rdf:resource="mem:libraryBoardMembers"/>
        <rdf:object>D. Madalli</rdf:object>
        <dc:creator>Biswa Nath Dutta</dc:creator>
    </rdf:Statement>
</rdf:RDF>
```
RDF: Summary Pro

- It has "sufficient" expressive power
- Scalable in theory
  - Everything is modelled by “simple” triple
- Standardize the syntax and abstract semantics
- Exchangeability
  - Inherited from XML
- Providing a **standard** way of defining standard vocabularies (but **without** defining any)
  - RDF Schema
RDF: Summary Contra

- Is not an optimal modeling language
  - Properties can be objects in statements
  - Reification seems misplaced in a low-level formalism
- Not immediately scalable in practice -> graph problems
- Does not support named graphs
- RDF/XML
  - Some technical limitations
    - e.g. http://example.org/foo(bar) no property name
  - No canonical serialization
  - Verbose format and expensive to write/parse
  - Not easy to learn for beginners
RDF Schema (RDFS)

- RDF provides a way to express simple statements about resources, using named properties and values, but

- We also need the ability to define the vocabularies (terms) they intend to use in those statements, specifically, to indicate that they are describing specific types or classes of resources

- Users can specify in RDF Schema
  - Classes and properties
  - **Class hierarchies**
    - Creating subclasses of classes
  - a new class can be created by extending an existing class
  - Class instances
  - **Property hierarchies**

- A class can have multiple super-classes

- **Towards** a logical foundation
RDF schema: type facilities

- RDF Schema definitions consist of classes (= “types”) and properties
- Individual object(s) belonging to a class is referred as instances of that class
- The relationship between instances and classes is expressed by rdf:type
- Schema definitions allow constraints on properties (which express validation conditions)
  - domain constraints link properties with classes
  - range constraints limit property values

- Schema definitions are expressed in RDF itself
  - Important: Vocabulary descriptions (i.e., schemas) written in the RDF Schema language are legal RDF graphs
RDF Layer vs. RDFS Layer
Core Classes

- **Important:** RDF Schema itself does not provide a vocabulary of application-specific classes
  - Provides a framework to do so

- `rdfs:Resource` the class of everything (i.e., class of all resources)

- `rdfs:Class` the class of all classes

- `rdfs:Literal` the class of all literals (strings)

- `rdfs:Datatype` is both an instance of and a subclass of `rdfs:Class`

- `rdf:Property` the class of all RDF properties; and is an instance of `rdfs:Class`. 
Core Properties

- **rdf:type** which relates a resource to its class
  - The resource is declared to be an instance of that class
- **rdfs:subClassOf** relates a class to one of its superclasses
  - All instances of a class are instances of its superclass
- **rdfs:subPropertyOf** relates a property to one of its superproperties
- **rdfs:domain** specifies the domain of a property P
  - The class of those resources that may appear as subjects in a triple with predicate P
  - If the domain is not specified, then any resource can be the subject
- **rdfs:range** which specifies the range of a property P
  - The class of those resources that may appear as values in a triple with predicate P
Task

Model some background knowledge about our Simpsons family in RDF Schema.
Modifiy our original RDF document (basically assign types) to use information from the schema.
Reification

- **rdf:Statement** the class of all reified statements
- **rdf:subject** relates a reified statement to its subject
- **rdf:predicate** relates a reified statement to its predicate
- **rdf:object** relates a reified statement to its object
Metaclasses

- A metaclass is a class of classes. Example:
  - The instances of an Aircraft class are:
    - F16, B1, F117, etc.
  - The properties of Aircraft are:
    - wingspan, range, weight, etc.
  - Each of the instances of Aircraft are themselves classes!
    - The instances of a F16 class are specific F16 planes.
    - The properties of F16 are:
      - tailNum, deployedAt, etc.
  - The Aircraft class is a metaclass!
Metaclasses

Aircraft

- F16
- F117
- B1
- XEJ-10
- Tango-1
Metaclasses

<?xml version="1.0"?>
<AirCraft rdf:about="http://www.nato.org/Aircraft.rdfs#F16"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns="http://www.nato.org/aircraft#">
  <wingspan>10 meters</wingspan>
  ...
</AirCraft>

F16 is being used as an instance!

<?xml version="1.0"?>
<F16 rdf:ID="XEJ-10"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns="http://www.military.org/aircraft#">
  <deployedAt>Persian Gulf</deployedAt>
  ...
</F16>

F16 is being used as a class!
Enough syntax and semantics for today ....
How about the soccer players?

The solution is DBPedia!

DBpedia is a community effort to extract structured information from Wikipedia and to make this information available on the Web. DBpedia allows you to ask sophisticated queries against Wikipedia and to link other datasets on the Web to Wikipedia data.
Any Wikipedia page

- Title
- Abstract
- Infoboxes
- Geo-coordinates
- Categories
- Images
- Links
  - Other languages
  - Other wiki pages
  - To the web
  - Redirects
  - Disambiguates
**Wikitext of Infoboxes**

{{infobox City Korea
  | full_name = Busan Metropolitan City
  | image = [[Image:Haeundaebeachbusan.jpg|250px|Haeundae Beach, Busan]]
  | rr = Busan Gwangyeoksi
  | mr = Pusan Kwangyŏksi
  | hangul = 부산 광역시
  | hanja = 釜山広域市
  | short_name = Busan (Pusan; 부산; 釜山)
  | population = 3,635,389 ...
  | area = 763.46 km²
  | government = [[Metropolitan cities of South Korea|Metropolitan City]]
  | divisions = 15 wards (Gu), 1 county (Gun)
  | region = [[Yeongnam]]
  | dialect = [[Gyeongsang Dialect|Gyeongsang]]
  | map = [[Image:Busan map.png|Map of South Korea highlighting the city]]
}}

<table>
<thead>
<tr>
<th>Korean name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Revised Romanization</td>
</tr>
<tr>
<td>McCune-Reischauer</td>
</tr>
<tr>
<td>Hangul</td>
</tr>
<tr>
<td>Hanja</td>
</tr>
<tr>
<td>Short name</td>
</tr>
</tbody>
</table>
Extracting Infobox data

http://en.wikipedia.org/wiki/Calgary

http://dbpedia.org/resource/Calgary

- dbpedia:native_name Calgary"
- dbpedia:altitude “1048”;
- dbpedia:population_city “988193”;
- dbpedia:populationMetro “1079310”;
- mayor_name
  - dbpedia:Dave_Bronconnier ;
- governing_body
  - dbpedia:Calgary_City_Council;


Calgary

- Mayor
  - Dave Bronconnier
    - (Past mayors)
    - Calgary City Council
- Governing body
  - Manager
    - Owen A. Tobert

Area
- City 726.50 km² (280.5 sq mi)
- Metro 5,107.43 km² (1,972 sq mi)
- Elevation 1,048 m (3,438.3 ft)

- City 988,193
- Density 1,360.2/km² (3,522.9/sq mi)
- Metro 1,079,310
- Population rank 3rd
- Metro rank 5th
The English version of the DBpedia knowledge base currently describes

3.77 million things, out of which

2.35 million are consistently classified, including

764,000 persons
573,000 places (including 387,000 populated places)
333,000 creative works (including 112,000 music albums, 72,000 films and 18,000 video games)
192,000 organizations (including 45,000 companies and 42,000 educational institutions)
202,000 species and
5,500 diseases.
Semantic Web technologies used:

- RDF as data model
- RDFS for background knowledge
- SPAQRL for queries RDF graphs
RDF Schema: Summary

- RDF Schema is a **primitive** ontology language

- The **key concepts** in RDF Schema are:
  - Class, and class relations, property, and property relations,
  - domain and range restrictions
  - “A simple ER-model”

- Is quite primitive as a modelling language for the Web
  - Offers **limited** modelling primitives with fixed meaning

- Many desirable modelling primitives are **missing**

- **So, we need an ontology layer** on top of RDF and RDF Schema
RDF Schema: missing!

- Equality of classes cannot be expressed directly
- No cardinality constraints
- No transitivity
- No inverse/symmetrical properties
- No localized range/domain constraints
- ...
- (RDFS/XML is ugly)
RDF Schema vs. XML Schema

- XML Schemas is all about **syntax**.
- RDF Schema is all about **semantics**.
- An XML Schema tool is intended to **validate** that an XML instance conforms to the syntax specified by the XML Schema.
- An RDF Schema tool is intended to **provide additional facts** to supplement the facts in RDF/XML instances.
- XML Schemas is **prescriptive** - an XML Schema prescribes what an element may contain, and the order the child elements may occur.
- RDF Schemas is **descriptive** - an RDF Schema simply describes classes and properties.