Chapter 3
RDF Syntax 1
RDF Overview

- RDF data model
- RDF syntax?
- RDF serializations: XML, Turtle, N3, ntriples
- RDF Schema (RDFS)
- Semantics of RDF and RDFS
  - Axiomatic Semantics
  - Operational semantics based on rules
- Querying RDF via SPARQL
Problem: What does an XML document mean?
- XML is about data structures
- The meaning (semantics) not apparent to machines

RDF is more a data model than a language
- It is realized in many different formats

RDF defines very basic semantics
- RDFS and OWL define more RDF vocabulary for building rich data models

RDF remains domain independent
What does this mean?
- Are professors also academic staff members?
- If someone teaches a course, are they an academic staff member?

Can’t say in XML, but can say so in RDFS
Example 2

- Embedding of elements is just a syntactic constraint
- No meaning is defined
- Meaning is in documentation or viewer’s minds
- Does the machine have a mind?
Key RDF documents: standards

http://w3.org/standards/techs/rdf

## RDF CURRENT STATUS

Completed Work

W3C Recommendations have been reviewed by W3C Members, by software developers, and by other W3C groups and interested parties, and are endorsed by the Director as Web Standards. Learn more about the W3C Recommendation Track.

Group Notes are not standards and do not have the same level of W3C endorsement.

### Standards

<table>
<thead>
<tr>
<th>Date</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>2014-01-16</td>
<td><strong>JSON-LD 1.0 Processing Algorithms and API</strong>&lt;br&gt;An Application Programming Interface and a set of algorithms for programatically transforming JSON-LD documents in order to make them easier to work with in programming environments like JavaScript, Python, and Ruby.</td>
</tr>
<tr>
<td>2014-01-16</td>
<td><strong>JSON-LD 1.0</strong>&lt;br&gt;A common JSON representation format for expressing directed graphs; mixing both Linked Data and non-Linked Data in a single JSON document.</td>
</tr>
<tr>
<td>2013-10-29</td>
<td><strong>Internationalization Tag Set (ITS) Version 2.0</strong>&lt;br&gt;This document defines data categories and their implementation as a set of elements and attributes called the Internationalization Tag Set (ITS). ITS 2.0 is the successor of ITS 1.0; it is designed to foster the creation of multilingual documents.</td>
</tr>
</tbody>
</table>
Topics

• Basic concepts of RDF
  • Resources, properties, values, statements, triples
  • URIs and URIrefs
  • RDF graphs
  • Literals, qnames
• Vocabularies and modeling
  • Vocabularies
  • Blank nodes, data modeling, types, reification
  • Lists, bags, collections
• Serialization of RDF graphs
  • XML, Turtle, Ntriples
• Critique of RDF
What is RDF?

- A data model for representing information (esp. metadata) about resources in the Web
- Can represent information about things that can be identified on the Web, even when not retrievable (e.g., a book)
- Use cases: provide data for applications rather than directly to people
RDF Basics

• Core idea: identify resources using **Web identifiers** and describing resources in terms of simple **properties** and property **values**.
• RDF is a RDF data model is as a “pure” graph model.
• To identify resources, RDF uses **Uniform Resource Identifiers (URIs)** and URI references (URIrefs).
• **Definition:** A **resource** is anything that is identifiable by a URIref.
Example

Consider the following information:

“there is a Person identified by http://www.w3.org/People/EM/contact#me, whose name is Eric Miller, whose email address is em@w3.org, and whose title is Dr.”
Example (cont’d)
The resources being described have properties which have values, and that resources can be described by making statements that specify those properties and values

- The part that identifies the thing the statement is about is the **subject**
- The part that identifies the property of the subject the statement specifies is the **predicate**
- The part that identifies the property’s value is the **object**
Example

http://www.example.org/index.html has a creator whose value is “John Smith”

- The **subject** is the URL http://www.example.org/index.html
- The **predicate** is the word "creator"
- The **object** is the phrase “John Smith”
RDF Triples

- RDF statements can be written as **triples**
- The simple **ntriples** notation has a set of triples terminated by a period, where URI’s are given inside angle brackets


Graphs: pure and impure
A pure graph model consists only of edges between pairs of nodes.

Can be directed or undirected; can be labeled or not.

A graph can be represented as an unordered collection of (subject, predicate, object) triples.

If directed, predicate goes from subject to object.

Nodes not the subject or object of a triple are not even allowed.

(John, likes, Mary),
(Mary, likes, Bill),
(John, hates, Bill)
RDF graph model

- RDF is like this with a few caveats
  - A subject or predicate is identified by a URI
  - An object can also be a URI but can also be a literal, i.e., a string or a number
- RDF has defines some special URIs and gives them a specific meaning
  - E.g., a type predicate
- RDF has simple conventions for representing both ordered and unordered sequences and a few other data structures
Graph databases have become popular in the past ten years.

A common extension of the pure graph model is to allow nodes and edges to have properties.

A simple version of this is that a property has a string name (e.g., age, date) and a literal value (e.g., 25, “1990-09-21”).

So we might give John an age property with value 25.

We might give the likes edge from John to Mary two properties: from with value “2016-09--1” and to with value “2016-09--1”.
URIs and URIREFs
Uniform Resource Identifiers (URIs)

- URIs identify resources on the Web
- Unlike URLs, they aren’t limited to identifying things with network locations
- No organization controls who makes URIs or how they can be used
  - Some URI schemes (http: URL’s) depend on centralized systems such as DNS
  - Others are **completely decentralized**
A **URIref** is a URI with an optional fragment identifier at the end, e.g:

```
http://example.org/index.html#section2
```

**Fragment usecase:**
- In HTML a # fragment refers to a place in the page
- In RDF we can use fragments to refer to resources in a RDF graph that the URI denotes, e.g., subjects, predicates or objects
  - [http://www.w3.org/2004/02/skos/core](http://www.w3.org/2004/02/skos/core) : vocabulary for describing topics
  - [http://www.w3.org/2004/02/skos/core#broader](http://www.w3.org/2004/02/skos/core#broader) : the *broader* concept in SKOS Core vocabulary

**Like URLs, URIrefs may be either absolute or relative**
- Note: the empty URI refers to the resource it’s in
URIrefs in RDF (cont’d)

- RDF and Browsers use URIrefs to **identify things**, but interpret URIrefs slightly differently:
  - Browsers also use URIrefs to **retrieve** things
  - RDF uses URIrefs **only** to identify things and these might not even be retrievable

- **Linked Data** best practice is to use HTTP URIs that return RDF data for every URI
  - [http://dbpedia.org/resource/Alan_Turing](http://dbpedia.org/resource/Alan_Turing)
  - `curl --I http://dbpedia.org/page/Alan_Turing`
  - `curl --H "Accept:application/rdf+xml" http://dbpedia.org/page/Alan_Turing`
RDF Graphs
RDF Graphs

- RDF models statements by **nodes** and **arcs** in a **graph**
- A **statement** is represented by a node for the subject, a node for the object and an arc for the predicate (subject => object)
- A **node** may be identified by a **URIref** or it can be a **literal** or a **blank node**
- An **arc** is identified by a **URIref**

**Note:** We will draw RDF graphs as **directed graphs**
  - But an arc can be the subject of an RDF statement
  - :has_parent owl:inverseOf :has_child
Consider the following statements:

- http://www.example.org/index.html has a creation-date whose value is August 16, 1999.
- http://www.example.org/index.html has a language whose value is English.
The RDF Graph of the Example

http://www.example.org/index.html

http://www.example.org/terms/creation-date
  August 16, 1999

http://purl.org/dc/elements/1.1/creator
  http://www.example.org/staffid/85740

http://purl.org/dc/elements/1.1/language
  en
In terms of the **relational model**, an RDF statement is like a **tuple in a relation** `Graph` with columns `Subject, Predicate, Object`.

For **first-order logic**, an RDF statement is like an **atomic formula** `triple(subj, pred, obj)` where `triple` is a FOL predicate and `subj, pred` and `obj` are constants.

- Alternatively: `pred(subj, obj)`
Literals and QNames
What is 27? Number or string?
Plain and Typed Literals

- There are two kinds of literals: **plain** and **typed**
- Plain literals have a **lexical form** (their lexical value) and optionally a **language tag**, e.g.:
  - “27”, “Hello world”@en
- **RDF typed literals** are formed by pairing a string with a URIref for a particular **datatype**, e.g.:
  - “27”^^http://www.w3.org/2001/XMLSchema#integer
  - ”27”^^xsd:int
Data Types for Literals

- In practice, the most widely used data typing scheme will be the one by XML Schema
  - But the use of any externally defined data typing scheme is allowed in RDF documents
- XML Schema predefines a large range of data types
  - E.g. Booleans, integers, floating-point numbers, times, dates, etc.
XMLSchema Datatypes

http://www.w3.org/TR/xmlschema-2/
The ntriples notation results in very long lines

We can use an **XML qualified name (QName)** w/o brackets for a full URI reference

- `http://dbpedia.org/page/Alan_Turing`
- `dbp:Alan_Turing`

A **Qname** has a **prefix** that has been assigned to a **namespace URI**, followed by a **colon**, and then a **local name**.

The concepts of **names** and **namespaces** used in RDF originate in XML
Topics

- Basic concepts of RDF
  - Resources, properties, values, statements, triples
  - URIs and URIrefs
  - RDF graphs
  - Literals, qnames
- Vocabularies and modeling
  - Vocabularies
  - Blank nodes, data modeling, types, reification
  - Lists, bags, collections
- Serialization of RDF graphs
  - XML, Turtle, Ntriples
- Critique of RDF