CS 2740 Knowledge representation
Lecture 17

Semantic web II

Milos Hauskrecht
milos@cs.pitt.edu
5329 Sennott Square

Semantic web

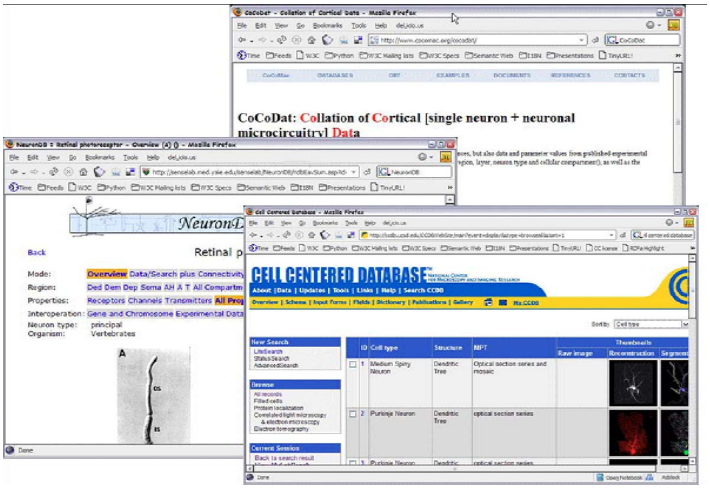
- Provides a common framework that allows data and knowledge to be shared and reused across application, enterprise, and community boundaries.
- The development is a collaborative effort led by **W3C**
- W3C defines: standards for exchanging knowledge and for sharing conceptualizations.

Basic standards:

- **RDF** - Resource Description Framework, representation of information/data for the purpose of sharing
  - **Based on XML - Extensible Markup Language format -** a general-purpose *specification* for building custom markup languages
- **OWL** – a language for sharing vocabularies, sets of terms supporting web searches and other applications (a part of RDF)

Semantic web

Problem: Many different knowledge/information sources and applications specific for these sources



Semantic web

Semantic Web and languages let us:

- Represent the knowledge
- Support search queries on knowledge and data
- Support inference

Specifics of the Semantic Web approach:

- Multiple sources of information and knowledge built for potentially many different purposes
- Ambiguities may arise (the same term with two different meanings or two different terms with the same meaning) and must be resolved
- Dynamically changing environment – knowledge is added at fast pace so it should be robust to handle

RDF

Resource Description Framework (RDF)

- a data model that lets us make statements about Web resources in the form of subject-predicate-object sentences, called *triples:*
- The subject denotes the resource, the predicate expresses a subject-object relationship
- The triples can be easily represented by a graph

Example: "The sky has the color blue"

- "a sky" is a subject
- "has the color" is a predicate
- "blue" is an object

RDF is an abstract model with several serialization formats (i.e., file formats): **typically XML**

Semantic web: OWL

The OWL Web Ontology Language (http://www.w3.org/TR/owl-ref/)

The Web Ontology Language OWL is:

- **a semantic markup language** for publishing and sharing **ontologies** on the World Wide Web.
- a **vocabulary extension of RDF** (the Resource Description Framework)

OWL contains all the reference information to **define any term contained within**

- it maintains its own definition of each and every term (it is self-referential).

Semantic web

Benefits:

- **knowledge integration,**
- **knowledge storage,**
- **knowledge searching,**
- **and knowledge inference.**

Semantic web

Benefits:

- **knowledge integration,**
- **knowledge storage,**
- **knowledge searching,**
- **and knowledge inference.**

CS 2740 Knowledge representation M. Hauskrecht

Semantic web: knowledge integration

Three steps of integration:

- ***Aggregation*:**
  - Combines the Semantic Web data sources into one unified, virtual data source.
- ***Mapping*/Binding:**
  - Associates similar references with each other and builds upon data in existing references. For example synonyms are identified.
- ***Rules*:**
  - Enables more sophisticated alignment and enrichment such as conditional logic that adds information based on the condition of other data

Semantic web: knowledge integration

Example: from Ivan Herman

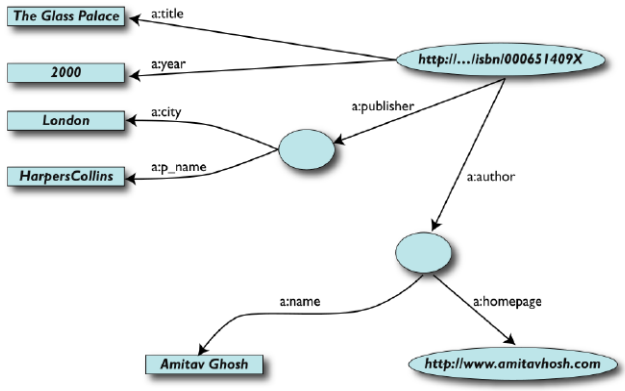
ID	Author	Title	Publisher	Year
ISBN 0-00-651409-X	id_xyz	The Glass Palace	id_qpr	2000

ID	Name	Home page
id_xyz	Amitav Ghosh	http://www.amitavghosh.com/

ID	Publisher Name	City
id_qpr	Harper Collins	London

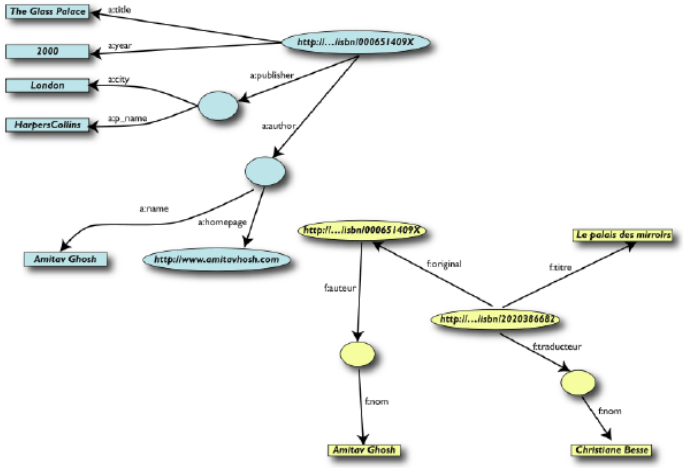
Semantic web: knowledge integration

Example: Date from one data source



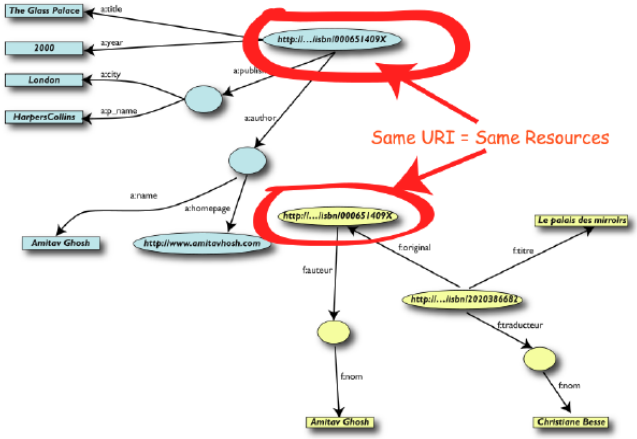
Semantic web: knowledge integration

Example: add a new data source



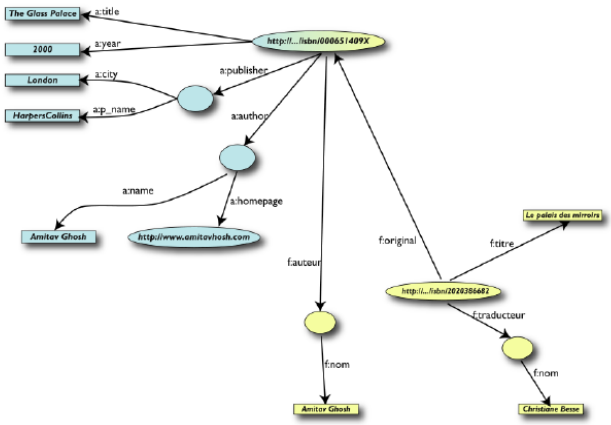
Semantic web: knowledge integration

Example: merge URIs



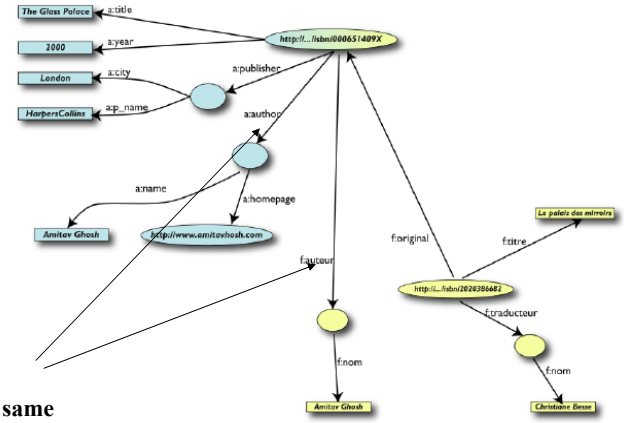
Semantic web: knowledge integration

Example: integration with respect to URIs



Semantic web: knowledge integration

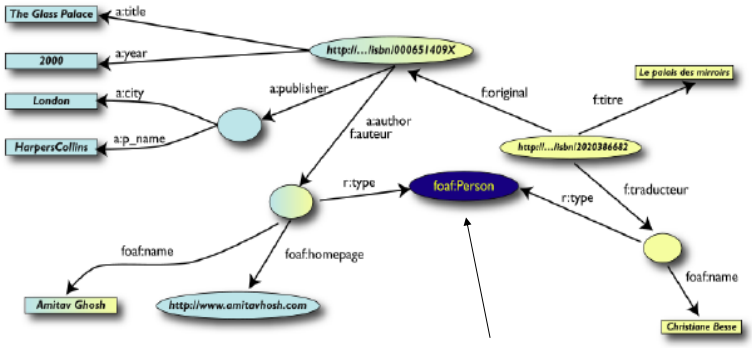
Example: Merging the relations



The same

Semantic web: knowledge integration

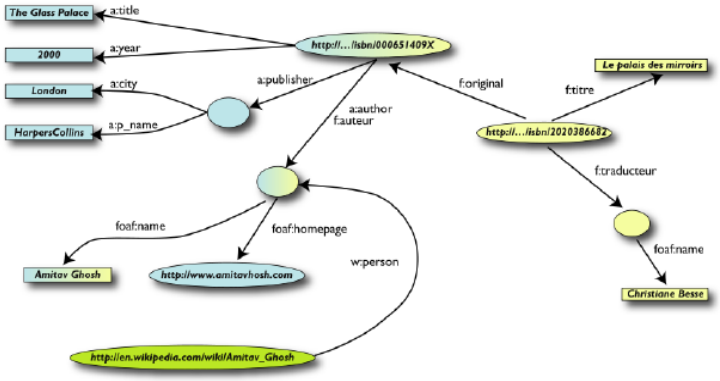
Example: uniformization with the help of ontology



Add ontology

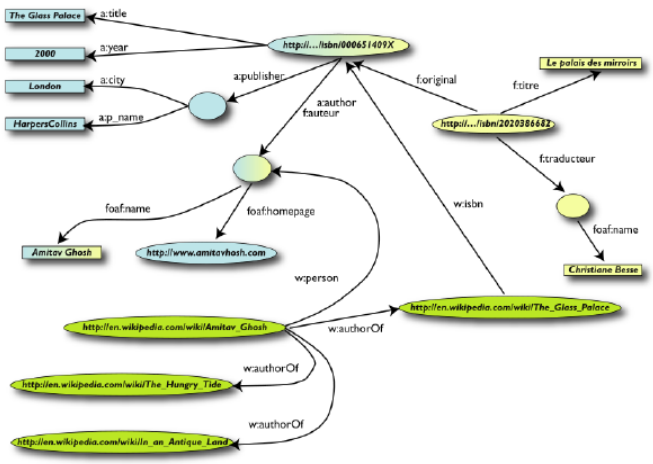
Semantic web: knowledge integration

Example: adding more knowledge



Semantic web: knowledge integration

Example: and go on



Semantic web: aggregation of sources

OWL is self-referential **:**

- **it** contains all the reference information to **define any term contained within** - it maintains its own definition of each and every term

OWL allows easy aggregation of multiple sources:

- simply add any OWL data to each other - in any combination or order . Unlike relational databases, the structure (i.e. schema) or ontology is just another set of statements within a Semantic Web data source. You can simply combine multiple OWL sources together. You cannot do this in the databases without significant work.

Queries:

- With OWL, you can simply query the knowledge structure the same way you query any instance data. An OWL query doesn't differentiate between the structure and the instance data.

Semantic web: mapping/binding

- More than one ontology may exist
- The same term may have multiple entries but it really can mean the same thing at the end
- Mapping allows us to accomplish two unifying actions;
  - declaring synonyms and
  - establishing relationships.

Synonyms: we can declare the two terms used in two different resources to be the same.

Relations: inheritance relations among terms can be defined

Semantic web: integration with rules

Rules enables more complex knowledge aggregation methods.

We can use if/then constructs to establish relationships and groupings.

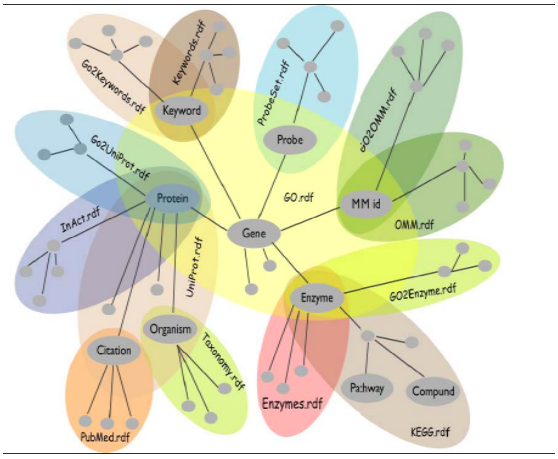
Rules can also add flexibility to your integration by handling special cases.

- **Example:** the *weather ontology* contains temperatures whereas the *project ontology* contains broader classifications such as hot and cold. We can establish a rule to convert certain temperatures to the correct hot or cold classification.

Rules can exist alongside with OWL as part of a the ***knowledge base***, or reside in the programs that manipulate the Semantic Web.

Semantic web: ontology buildings

- Many possible knowledge/data sources



Semantic web

Benefits:

- **knowledge integration,**
- **knowledge construction and storage,**
- **knowledge inference,**
- **and knowledge searching.**

Knowledge construction and storage

Knowledge is stored in databases:

- Originally one database was used by one or many applications
- Multiple databases can be used by multiple applications

Problem:

- Applications take advantage of pieces of knowledge stored in databases
- Each application has its own view so that the knowledge in the database is fragmented and relations are lost

Semantic web approach:

- Do not fragment data. Knowledge is built upon all data. A central term is enriched with relations, attributes, constraints defining its context. Relations and attributes are terms and define semantic links in between objects (entities) instead of just links as in the html.

Knowledge construction and storage

Knowledge construction and enrichment:

- **Horizontal:**
  - add new attributes and peer relationships. Examples include adding a *birthday* to *person* (new attribute) and adding a *boss relationship* between two *workers* (new peer relationships)
- **Vertical:**
  - via inheritance. Inheritance provides all the context of the base term plus whatever else we want to add. E.g. a *person* has a *name*, *birthday*, and *sex*. A *worker* is a type of *person* that also adds a *workplace* and *boss*. An update to *person*, such as adding a *birthplace*, automatically adds *birthplace* to all *workers*.

Knowledge construction and storage

Knowledge construction and enrichment:

- **Constraints:**
  - new constraints can be introduced to further refine the context. For example *parent* is defined as a *person* with a *daughter* and/or a *son*. Constraints can get quite rich with logic such as *tall person* is a *person* with *height* greater than six feet.
- **Distributed:**
  - build on knowledge anywhere in your network. Knowledge and data that resides elsewhere are referenced uniquely with the help of its URI.

Knowledge construction and storage

What knowledge can be expressed:

Commonality:

- Declaring two data items equivalent simplifies data. This could occur in the structural ontology level in declaring person and contractor as the same. This also extends to specific instances. You can declare Joe Smith at a given URI equal to J Smith at another URI. So simply declaring two items equal adds knowledge.
- OWL uses:
  - "equivalentClass" keyword to establish a connection between two unique URIs declaring them equal or synonyms.

Knowledge construction and storage

What knowledge can be expressed:

- **Inheritance:** This adds knowledge in declaring that a data element is a form of another data element but not *exactly* equivalent. One element is a subset of the other. All people have names and addresses but not all people are e.g. managers. Thus we could declare a manager a type of person but still distinct from people. This clarifies a term without duplicating similar information.

- **OWL:**
  - Inheritance is implemented using subClassOf keyword. .

Knowledge construction and storage

What knowledge can be expressed:

- **Restrictions:**
- Restrictions define a term relative to other terms or limits. For example, an *available* contractor must have the constraint of availability. This adds to the useful vocabulary by adding a new term that is a condition of an existing term.

Knowledge construction and storage

What knowledge can be expressed:

- **Properties:**
- data elements that describe another data element. A person has an property called "livesAt" populated with her home address. Similarly, the person may have an additional property called "worksAt" populated with her work address. A data type address is hence used to describe two data elements – work and home address.

Knowledge construction and storage

What knowledge can be expressed:

- **Collections:**
- abstractions built by combining terms together. The concept referred to by a particular term may be related to several other terms simultaneously. For instance, the term "contractor" could describe a collection of things called "skilled manual laborers" and also things called "construction workers".

Knowledge construction and storage

What knowledge can be expressed:

- **Rules:**
- Rules are used to wrest the knowledge out of particularly complex situations - ones that can't simply be addressed with the methods above. They allow us to consider one term for a given condition that then drives another term. It allows if/then constructs in defining the context of a given term.

Semantic web

Benefits:

- **knowledge integration,**
- **knowledge construction and storage,**
- **knowledge searching,**
- **and knowledge inference.**

Knowledge searching

Getting answers, is the end goal of any knowledge base or data formation.

Two ways to search the data/knowledge bases:

- **keyword matching**
  - Gives useful results if the data pattern is unusual and dissimilar to other data patterns. Many matches (e.g. 10,000 hits) on more common patterns are no good. Keyword matching is also weak at asking specific, detailed questions like what is the population of Utah in 1982**. Works even if we dynamically keep changing the knowledge base.**
- **relational database queries**
  - Relies on a standard language (SQL) that spans multiple database technologies and allows quite a bit of power. However, the queries are tightly taylored to the structure of each individual database, which when using keys, foreign keys, indexes etc. can quickly get quite complex. It is great for a well known structure but hardly useful for a dynamically changing and large knowledgebase.
- Semantic web is a compromise in between the two.

Knowledge searching

Semantic web:

- **supports several query languages** that enable powerful and flexible interrogation of resources. It provides structure to ask direct questions but also enough flexibility that you need not be an expert as to a specific Semantic Web formation.
- Queries have the same structure whether we are asking a question regarding the knowledge structure (i.e. *is a person the same as a contractor*?) and/or instance data (i.e. *is John a contractor?*).

A search query is nothing more than knowledge reflecting a certain interest or perspective. So queries may be directed into the ontology.

You can keep asking the same question, while the underlying data is dramatically changing through the integration of new data sources, and continually receive a better answer.

Semantic web

Benefits:

- **knowledge integration,**
- **knowledge construction and storage,**
- **knowledge searching,**
- **and knowledge inference.**

Knowledge inference

Search identifies and returns answers that are explicitly represented in available knowledge resources

Inference addresses the questions for which the answer is not directly available and encoded

- This is were **KR&R** experience helps

Problems:

- Synonyms (are these two things equivalent?)
- Ambiguities - different contexts for the same term may lead to different and contradictory answers

Knowledge perspectives

The Semantic Web enables you to have *knowledge your own way*. It does not force you to adopt someone else's view of data or knowledge.

This is accomplished using:

- Integration of knowledge and existing resources
- Construction of a new knowledge
- Support for inferences

Basically the options you have are:

- Start from scratch and build everything you need for your application
- Tap on resources available that you can tailor to your needs, you reuse not only the information, you also can reuse and integrate the semantics