CS 2740 Knowledge representation
Lecture 21

Semantic web 2.

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

Semantic web

The **Semantic Web** provides a common framework that allows data and knowledge to be shared and reused across application, enterprise, and community boundaries.

The development of the Sematic web is a collaborative effort led by **W3C** with participation from a large number of researchers and industrial partners.

It defines standards for exchanging knowledge and for sharing conceptualizations.

Basic standards:

- **RDF** - Resource Description Framework, representation of information/data for the purpose of sharing

  OWL – a language for sharing vocabularies, sets of terms supporting web searches and other applications (a part of RDF)

Semantic web

In terms of the knowledge representation and reasoning:

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

Differences from other KR systems:

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

Semantic web

Benefits:

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

Semantic web

Benefits:

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

Semantic web: knowledge integration

Benefit of large amounts of information and knowledge on the web stands and falls on the data/knowledge integration

Technical challenges:

- ***Location*:** where the data/knowledge resides. The *location* of a Semantic Web resource is defined by the **Uniform Resource Identifier (URI).** A URI is simply a formatted string that identifies - via name, location, or any other characteristic - a resource. A standard web link is a form of a URI. URI allows us to label a Semantic Web source with a *findable, unique* location.
- ***Query Protocol*:** We need to interact with web resources. We need an communication language. The protocol for the Semantic Web uses standards such as ***http*** to form a flexible, easily understood, request/response exchange.
- ***Format*:** The data must be in a comprehensive and translatable format. The Semantic Web uses a standard format - the **OWL Web Ontology Language.** It is based on the **Resource Description Framework (RDF)** standard and **XML.**

Technical challenges are resolved by standards

Semantic web: knowledge integration

Other Challenges

- ***Timely, Authoritative*****:** The data must be trusted and be up-to-date. It is possible to have multiple answers to the same question. In addition, information may get outdated. The Semantic Web lets you to deal directly with the actual source to avoid the problem. You need not maintain complex synchronization unless it is absolutely necessary due to performance or other requirements.

The key challenge:

- ***Purpose*****:** We have to align the data with our purpose. This may require translation and modifications. It needs to fit your world view be it english, medical, financial to name but a few. This is about getting right the semantic. For example, we can tie *person* in one data source with *individual* from another data source - they represent the same meaning or a related meaning.
- Semantic web standards do enable easier and more efficient data sharing and integration but really reach their full potential by the ability to align purpose across different data sources.

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: OWL

The Semantic Web is always written in the same language:

- **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).

Ontology

If more than one person is building a knowledge base, they must be able to share the **conceptualization**.

- A conceptualization is a mapping from the problem domain into the representation.
- A conceptualization specifies:
  - What types of objects are being modeled
  - The vocabulary for specifying objects, relations and properties
  - The meaning or intention of the relations or properties
- **An ontology is a specification of a conceptualization.**

Semantic web: OWL

```
- <owl:Ontology rdf:about="">
  <rdfs:comment>This is a weather forecast ontology.</rdfs:comment>
  <rdfs:label>Weather Site Ontology</rdfs:label>
  </owl:Ontology>
<!-- Weather Observation Class   -->
- <owl:Class rdf:ID="WeatherObservation">
  <rdfs:label>Weather Observation</rdfs:label>
- <rdfs:subClassOf>
- <owl:Restriction>
  <owl:onProperty rdf:resource="#hasLocation" />
  <owl:cardinality>1</owl:cardinality>
  </owl:Restriction>
  </rdfs:subClassOf>
- <rdfs:subClassOf>
- <owl:Restriction>
  <owl:onProperty rdf:resource="#hasTime" />
  <owl:cardinality>1</owl:cardinality>
  </owl:Restriction>
  </rdfs:subClassOf>
…
```

Ontology definition

Class definition

Class properties

Semantic web: OWL

```
…
<rdfs:subClassOf>
- <owl:Restriction>
  <owl:onProperty rdf:resource="#hasTemperature" />
  <owl:cardinality>1</owl:cardinality>
  </owl:Restriction>
  </rdfs:subClassOf>
- <rdfs:subClassOf>
- <owl:Restriction>
  <owl:onProperty rdf:resource="#hasHumidity" />
  <owl:cardinality>1</owl:cardinality>
  </owl:Restriction>
  </rdfs:subClassOf>
- <rdfs:subClassOf>
- <owl:Restriction>
  <owl:onProperty rdf:resource="#hasWindSpeed" />
  <owl:cardinality>1</owl:cardinality>
  </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
```

Class properties

Semantic web: OWL

```
<!-- Location Class   -->
- <owl:Class rdf:ID="Location">
  <rdfs:label>Location: City, State</rdfs:label>
- <rdfs:subClassOf>
- <owl:Restriction>
  <owl:onProperty rdf:resource="#hasState" />
  <owl:cardinality>1</owl:cardinality>
  </owl:Restriction>
  </rdfs:subClassOf>
- <rdfs:subClassOf>
- <owl:Restriction>
  <owl:onProperty rdf:resource="#hasCity" />
  <owl:cardinality>1</owl:cardinality>
  </owl:Restriction>
  </rdfs:subClassOf>
  </owl:Class>
- <owl:DatatypeProperty rdf:ID="hasState">
  <rdfs:label>The State that this location is in. Abbreviated.</rdfs:label>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string" />
  </owl:DatatypeProperty>
```

Another class definition

Semantic web: OWL

```
<!-- Location Class   -->
- <owl:Class rdf:ID="Location">
  <rdfs:label>Location: City, State</rdfs:label>
- <rdfs:subClassOf>
- <owl:Restriction>
  <owl:onProperty rdf:resource="#hasState" />
  <owl:cardinality>1</owl:cardinality>
  </owl:Restriction>
  </rdfs:subClassOf>
- <rdfs:subClassOf>
- <owl:Restriction>
  <owl:onProperty rdf:resource="#hasCity" />
  <owl:cardinality>1</owl:cardinality>
  </owl:Restriction>
  </rdfs:subClassOf>
  </owl:Class>
- <owl:DatatypeProperty rdf:ID="hasState">
  <rdfs:label>The State that this location is in. Abbreviated.</rdfs:label>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string" />
  </owl:DatatypeProperty>
```

Another class definition

hasState property datatype

Semantic web: OWL

```
<!-- Precipitation Class   -->
- <owl:Class rdf:ID="Precipitation">
  <rdfs:label>Precipitation Condition</rdfs:label>
- <owl:oneOf rdf:parseType="Collection">
  <Precipitation rdf:about="#Snow" />
  <Precipitation rdf:about="#Rain" />
  <Precipitation rdf:about="#Thunderstorm" />
  <Precipitation rdf:about="#None" />
  </owl:oneOf>
  </owl:Class>
```

Semantic web: OWL

Properties for Weather Observation class

```
<!-- Properties   -->
- <owl:ObjectProperty rdf:ID="hasLocation">
  <rdfs:label>Location of observation.</rdfs:label>
  <rdfs:range rdf:resource="#Location" />
  </owl:ObjectProperty>
- <owl:DatatypeProperty rdf:ID="hasTime">
  <rdfs:label>Date and time of observation.</rdfs:label>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string" />
  </owl:DatatypeProperty>
- <owl:DatatypeProperty rdf:ID="hasTemperature">
  <rdfs:label>Temperature, farenheit</rdfs:label>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#float" />
  </owl:DatatypeProperty>
- <owl:DatatypeProperty rdf:ID="hasHumidity">
  <rdfs:label>Relative humidity, percent.</rdfs:label>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#float" />
  </owl:DatatypeProperty>
```

Semantic web: OWL

Example: Aggregating knowledge from multiple ontologies

```
<owl:Ontology rdf:about="">
  <rdfs:comment>This is the project assignment client ontology</rdfs:comment>
  <rdfs:label>Project Assignment Client Ontology</rdfs:label>
  <owl:imports rdf:resource="http://localhost/contractors/ont/contractor-ont.owl"
     />
  <owl:imports rdf:resource="http://localhost/weather/ont/weather-ont.owl" />
  <owl:imports rdf:resource="http://localhost/projectsite/ont/project-ont.owl" />
  </owl:Ontology>
```

New 'Project assignment' ontology
Uses 3 ontologies: weather, project, contractor

Semantic web: OWL

```
<!-- Weather   -->
- <owl:Class rdf:ID="CurrentWeather">
  <rdfs:subClassOf rdf:resource="http://localhost/weather/ont/weather-
     ont.owl#WeatherObservation" />
  <owl:equivalentClass rdf:resource="http://localhost/weather/ont/weather-
     ont.owl#WeatherObservation" />
- <rdfs:subClassOf>
- <owl:Restriction>
  <owl:onProperty rdf:resource="#hasTemperature" />
  <owl:cardinality>1</owl:cardinality>
  </owl:Restriction>
  </rdfs:subClassOf>
- <rdfs:subClassOf>
- <owl:Restriction>
  <owl:onProperty rdf:resource="#forCity" />
  <owl:cardinality>1</owl:cardinality>
  </owl:Restriction>
  </rdfs:subClassOf>
…
</owl:Class>
```

Class Current Weather in the new Ontology: equivalent class Weather Observation

Semantic web: OWL

```
<!-- Weather   -->
- <owl:Class rdf:ID="CurrentWeather">
  <rdfs:subClassOf rdf:resource="http://localhost/weather/ont/weather-
     ont.owl#WeatherObservation" />
  <owl:equivalentClass rdf:resource="http://localhost/weather/ont/weather-
     ont.owl#WeatherObservation" />
- <rdfs:subClassOf>
- <owl:Restriction>
  <owl:onProperty rdf:resource="#hasTemperature" />
  <owl:cardinality>1</owl:cardinality>
  </owl:Restriction>
  </rdfs:subClassOf>
- <rdfs:subClassOf>
- <owl:Restriction>
  <owl:onProperty rdf:resource="#forCity" />
  <owl:cardinality>1</owl:cardinality>
  </owl:Restriction>
  </rdfs:subClassOf>
…
</owl:Class>
```

Different properties as used for the weather observation class before. Temperature, for state, for city, is hot, is dry

Semantic web: OWL

```
<owl:DatatypeProperty rdf:ID="hasTemperature">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string" />
  <owl:equivalentProperty rdf:resource="http://localhost/weather/ont/weather-
     ont.owl#hasTemperature" />
  </owl:DatatypeProperty>
- <owl:DatatypeProperty rdf:ID="forCity">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string" />
  </owl:DatatypeProperty>
- <owl:DatatypeProperty rdf:ID="forState">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string" />
  </owl:DatatypeProperty>
- <owl:DatatypeProperty rdf:ID="isWarm">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#boolean" />
  </owl:DatatypeProperty>
- <owl:DatatypeProperty rdf:ID="isDry">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#boolean" />
  </owl:DatatypeProperty>
```

Semantic web: aggregation of sources

- **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).
- **Consequence:** ***Aggregation of multiple sources is easy***.
  - *We can 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 just pour relational database data into another database without significant work behind behind the scenese with the databases schemas to clean up conflicts and the like.

  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

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 is:

- **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*.
- **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 by 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. "equivalentClass" keyword in OWL establishes a connection between two unique URIs declaring them equal or synonyms.
- **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. Inheritance is implemented using subClassOf keyword. .
- **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.
- **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".
- **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**
  - It can give 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 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