Ontologies, Description Logics and Semantic Web

A Short Introduction

Marek Obitko
obitko@labe.felk.cvut.cz

Department Of Cybernetics
Faculty of Electrical Engineering
Czech Technical University in Prague

October 23, 2008
Outline

Introduction

Ontologies

Description Logics

Semantic Web

Conclusion
Outline

Introduction

Ontologies

Description Logics

Semantic Web

Conclusion
Importance of Knowledge

- Critical for intelligent systems
- In many cases, better knowledge can be more important for solving a task than better algorithms
- Truly intelligent system — knowledge capture, processing, reuse, communication
- How to express knowledge or information?
- Need to agree! $\Rightarrow$ Ontologies
Knowledge Sharing — Motivation

- Real-life example (US Army) of transportation chain involving different companies around the world
- Transport of containers weighing tons - works well
- Transport of information saying what is in containers - doesn’t work well
- Containers need to be unpacked to see what is in them
- Problem: different computer systems that were not designed to work together
- Systems designed to be efficient, but the interface between the systems require significant human intervention
Outline

Introduction

Ontologies

Description Logics

Semantic Web

Conclusion
Philosophical Roots

- Ontology = branch of philosophy
  - what is existence, what properties can explain existence
  - Aristotle: science of being as such
  - all the species qua being and the attributes that belong to it qua being
    - reformulation: what is the being?
  - what are the features common to all beings?
- general ontology versus ontologies for a particular domain
- entities of the world (physical objects, events, regions, ...), meta-level categories (concept, property, quality, state, role, part...)
Explicit Specification of Conceptualization

- conceptualization
  - semantic structure for encoding implicit knowledge
  - language independent
  - way of seeing domain, usually also depends on application

- explicit specification ...
  - description
  - explicit, using a (formal) language (like a formal specification of a program)

- engineering artifact
  - concepts, relations, described in some logical language
  - constraints — what is possible, background knowledge
Example — Transportation System

- **Ontology (description of domain)**
  - Conveyor-belt
  - Diverter
  - is-connected-to
  - is-connected-to *is symmetric*

- **Knowledge base (particular state of affairs)**
  - ConveyorBelt123 is-connected-to DiverterXYZ
  - DiverterXYZ is-connected-to ConveyorBelt123
Explicit Specification of Conceptualization

Intended models (description of the domain, what is possible in it)

Ontology (restriction of the possible models, expressing conceptualization)

Possible models expressible in the ontology language
Ontological Commitment

- Agreement to use a vocabulary...
- (to ask and make assertions)
- ... in a way consistent with ontology
- Shared ontology
- agents can share knowledge, i.e. communicate it and reason in consistent way
Formality

**Formal** explicit specification of shared conceptualization
Outline

Introduction

Ontologies

Description Logics

Semantic Web

Conclusion
Description Logics

- Formal description of concepts and roles
- Formalization of semantic networks and frame-based systems
- Intents
  - enough for practical modeling purposes
  - good computational properties (e.g., decidability)
- Usually subset of first order predicate logic
- DL system = TBox (terminology) + ABox (assertions)
Attributive Language $\mathcal{AL}$

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Semantics</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A$</td>
<td>$A^I \subseteq \Delta^I$</td>
<td>atomic concept</td>
</tr>
<tr>
<td>$R$</td>
<td>$R^I \subseteq \Delta^I \times \Delta^I$</td>
<td>atomic role</td>
</tr>
<tr>
<td>$\top$</td>
<td>$\Delta^I$</td>
<td>top (most general) concept</td>
</tr>
<tr>
<td>$\bot$</td>
<td>$\emptyset$</td>
<td>bottom (most specific) concept</td>
</tr>
<tr>
<td>$\neg A$</td>
<td>$\Delta^I \setminus A^I$</td>
<td>atomic negation</td>
</tr>
<tr>
<td>$C \sqcap D$</td>
<td>$C^I \cap D^I$</td>
<td>intersection</td>
</tr>
<tr>
<td>$\forall R.C$</td>
<td>${a \in \Delta^I</td>
<td>\forall b. (a, b) \in R^I \Rightarrow b \in C^I }$</td>
</tr>
<tr>
<td>$\exists R.\top$</td>
<td>${a \in \Delta^I</td>
<td>\exists b. (a, b) \in R^I }$</td>
</tr>
</tbody>
</table>

Example:
- Person $\sqcap$ Female
- Person $\sqcap \neg$Female
- Person $\sqcap \exists$hasChild.$\top$
- Person $\sqcap \forall$hasChild.Female
### AL Extensions

<table>
<thead>
<tr>
<th>Name</th>
<th>Syntax</th>
<th>Semantics</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{U}$</td>
<td>$C \sqcup D$</td>
<td>$C^I \cup D^I$</td>
<td>union of two concepts</td>
</tr>
<tr>
<td>$\mathcal{E}$</td>
<td>$\exists R.C$</td>
<td>${a \in \Delta^I</td>
<td>\exists b.(a, b) \in R^I \land b \in C^I}$</td>
</tr>
<tr>
<td>$\mathcal{N}$</td>
<td>$\geq nR$</td>
<td>${a \in \Delta^I</td>
<td></td>
</tr>
<tr>
<td>$\leq nR$</td>
<td>${a \in \Delta^I</td>
<td></td>
<td>{b</td>
</tr>
<tr>
<td>$\mathcal{C}$</td>
<td>$\neg C$</td>
<td>$\Delta^I \setminus C^I$</td>
<td>negation of arbitrary concept</td>
</tr>
</tbody>
</table>

- $\mathcal{S}$ — role transitivity $\text{Trans}(R)$ (asserting that role is transitive)
- $\mathcal{H}$ — role hierarchy $R \subseteq S$ (asserting hierarchy of roles)
- $\mathcal{I}$ — role inverse $R^-$ (creating inverse role)
- $\mathcal{F}$ — functionality $\leq 1R$ (functional role in concept creation)
- $\mathcal{O}$ — nominals $\{a_1, ..., a_n\}$ (concept declared by enumeration)
Reasoning

- Satisfiability of a concept — whether an individual can exist
- Subsumption of concepts — $C$ subsumes $D$?
- Consistency of ABox with respect to TBox
- Check an individual — is individual an instance of a concept?
- Retrieval of individuals — find instances of a concept
- Realization of an individual — find concepts for individual

- All can be reduced to satisfiability or subsumption
- Algorithms: structural, logical
- Tableau algorithm: proving satisfiability, using expansion rules
Example — OWL Pizzas in Protégé

- In Web Ontology Language OWL
- Corresponds to description logic $SHOIN$
- Web Ontology Language terminology
  - Class = concept
  - Property = role

- Let’s start Protégé...

- [http://protege.stanford.edu/download/download.html](http://protege.stanford.edu/download/download.html)
Example — OWL Pizzas in Protégé

- Open pizzas.owl
- Check consistency — everything OK?
- Classify taxonomy — see classification
- Define CheesePizza
- Classify taxonomy — see classification
- Full tutorial, if you are interested: http://www.co-ode.org/resources/tutorials/ProtegeOWLTutorial.pdf
Outline

Introduction

Ontologies

Description Logics

Semantic Web

Conclusion
Current WWW = hypertext web, library of interlinked documents transferred by computers, presented to people

Semantic web = effort to enhance current web so that computers can process information presented in WWW

Share data instead of documents (not necessarily AI!)

...provide a common framework that allows data to be shared and reused across application, enterprise, and community boundaries. It is a collaborative effort led by World Wide Web Consortium (W3C)...

Semantic Web Architecture

- User interface and applications
- Trust
- Proof
- Unifying logic
  - Querying: SPARQL
  - Ontologies: OWL
  - Rules: RIF/SWRL
  - Taxonomies: RDFS
  - Data interchange: RDF
  - Syntax: XML
  - Identifiers: URI
  - Character set: UNICODE
- Cryptography
Resource Description Framework (RDF)

- representing information about resources (IRI) in a graph form
- triples *subject-predicate-object*
Resource Description Framework (RDF)

- subject and predicate can be resource
- object can be resource or literal (may be typed using XML Datatypes)
RDF Graph

http://xmlns.com/foaf/0.1/Person

http://www.w3.org/1999/02/22-rdf-syntax-ns#type

http://xmlns.com/foaf/0.1/homepage

http://www.example.org/~joe/contact.rdf#joesmith

http://xmlns.com/foaf/0.1/mbox

http://xmlns.com/foaf/0.1/givenname

mailto:joe.smith@example.org

Joe

Smith
@prefix : <http://www.example.org/~joe/contact.rdf#> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .

:joesmith a foaf:Person ;
    foaf:givenname "Joe" ;
    foaf:family_name "Smith" ;
    foaf:homepage <http://www.example.org/~joe/> ;
    foaf:mbox <mailto:joe.smith@example.org> .
RDF Serialization — XML

```xml
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:foaf="http://xmlns.com/foaf/0.1/
    xmlns="http://www.example.org/~joe/contact.rdf#">
  <foaf:Person rdf:about="http://www.example.org/~joe/contact.rdf#joesmith">
    <foaf:mbox rdf:resource="mailto:joe.smith@example.org"/>
    <foaf:homepage rdf:resource="http://www.example.org/~joe/"/>
    <foaf:family_name>Smith</foaf:family_name>
    <foaf:givenname>Joe</foaf:givenname>
  </foaf:Person>
</rdf:RDF>
```
Selected RDF vocabulary

- **rdf:type** — it is a predicate used to state that a resource is an instance of a class
- **rdf:Property** — the class of properties (i.e. binary relations that are used as predicates in triples)
- **rdf:Alt**, **rdf:Bag**, **rdf:Seq** — containers of alternatives, unordered containers, and ordered containers
- **rdf:List** — the class of RDF Lists
- **rdf:nil** — an instance of **rdf:List** representing the empty list
- **rdf:Statement**, **rdf:subject**, **rdf:predicate**, **rdf:object** — reification support
RDF Schema RDFS

- extending RDF vocabulary
- describing taxonomies of classes and properties
- range and domain of properties

@prefix : <http://www.example.org/sample.rdfs#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>.

:Dog rdfs:subClassOf :Animal.
:Person rdfs:subClassOf :Animal.
:hasSon rdfs:subPropertyOf :hasChild.
:Max a :Dog.
:Abel a :Person.
:Adam a :Person; :hasSon :Abel.
Web Ontology Language OWL

- Syntactically builds RDF and RDFS
- Its aim is to bring description logic to semantic web
- For employing logic, the freedom of RDF has to be limited
- Three species
  - OWL Lite — taxonomy and simple constraints, such as 0 and 1 cardinality (SHIF)
  - **OWL DL** — full power of DL (SHOIN)
  - OWL Full — freedom of RDF, no computational properties guaranteed
- In short, OWL = \(SHOIN(D)\) written in RDF
  - classes, object properties, datatype properties, individuals
  - taxonomies, restrictions — cardinality, values
  - open world assumption
OWL Pizza Example — DL notation

\[ \text{Pizza} \sqsubseteq \exists \text{hasBase}. \text{PizzaBase} \]
\[ \text{Pizza} \sqcap \text{PizzaBase} \equiv \bot \]
\[ \text{NonVegetarianPizza} \equiv \text{Pizza} \sqcap \neg \text{VegetarianPizza} \]
\[ \text{Tr(isIngredientOf)} \]
\[ \text{isIngredientOf} \equiv \text{hasIngredient}^- \]
OWL Pizza Example — OWL Abstract Syntax Notation

Namespace(p = <http://example.com/pizzas.owl#>)
Ontology( <http://example.com/pizzas.owl#>
    Class(p:Pizza partial
        restriction(p:hasBase someValuesFrom(p:PizzaBase)))
    DisjointClasses(p:Pizza p:PizzaBase)
    Class(p:NonVegetarianPizza complete
        intersectionOf(p:Pizza complementOf(p:VegetarianPizza)))
    ObjectProperty(p:isIngredientOf Transitive
        inverseOf(p:hasIngredient))
)
OWL Pizza Example — in RDF, N3 notation

@prefix : <http://example.com/pizzas.owl#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
:Pizza rdfs:subClassOf
   [ a owl:Restriction ;
   owl:onProperty :hasBase ;
   owl:someValuesFrom :PizzaBase ] ;
owl:disjointWith :PizzaBase .
:NonVegetarianPizza owl:equivalentClass
   [ owl:intersectionOf
     ( [owl:complementOf :VegetarianPizza]
     :Pizza ) ] .
:isIngredientOf
   a owl:TransitiveProperty , owl:ObjectProperty ;
owl:inverseOf :hasIngredient .
OWL Pizza Example — in RDF, graph
OWL Reasoning Examples — Transport

- :targetNode rdfs:subPropertyOf :connectedTo.
  - “node TN is target node of a conveyor belt CB” entails “TN and CB are connected”
- :connectedTo a owl:SymmetricProperty.
  - “X is connected to Y” entails “Y is connected to X”
- :targetNode a owl:FunctionalProperty.
  - Commercial dept.: “Node Z1 is target node of conveyor belt BE”
  - Router supplier: “Router R5 is target node of conveyor belt BE”
  - ...entails “Node Z1 and Router R5 is the same thing” (can be explicitly stated using owl:sameIndividualAs)
- :contains a owl:TransitiveProperty.
  - “A contains B” and “B contains C” entails “A contains C”
RDF Query Language SPARQL

- SQL-like query language
- syntax based on N3/TURTLE

```sparql
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?name ?mbox
WHERE { ?x foaf:name ?name .
  ?x foaf:mbox ?mbox . }

FILTER (?price < 20)
ORDER BY, DISTINCT, ...
```
RDF Query Language SPARQL

- SELECT, CONSTRUCT, DESCRIBE, ASK

PREFIX foaf: <http://xmlns.com/foaf/0.1/>
CONSTRUCT { <http://example.org/person#Alice> foaf:knows ?x } FROM <http://example.org/foaf/people>
WHERE { ?x foaf:name ?name }
ORDER BY desc(?name)
LIMIT 2

- any RDF querying or manipulation
- ... including OWL, but RDF notation has to be used
OWL Pizza Example — SWOOP

- Look at pizza.owl in other than pure logical view...
- Download: http://code.google.com/p/swoop/
Semantic Web Realization

- Evolution from current web
- Annotations
  - Adding to data in existing documents
  - For some data easier than you may think — big part of web is generated from databases!
- Using annotated data
  - Connecting data together (mash-up from different sources) — possible today, very powerful
  - Even little semantics means a huge step for practical use
  - Reasoning with distributed data — still research
- “Enrichment of current web” versus “Web of data”
Outline

Introduction

Ontologies

Description Logics

Semantic Web

Conclusion
Summary

- Ontology — formal explicit specification of shared conceptualization
- Description logics — practical modeling, good computational properties for reasoning
- Semantic web — semantic extension of hypertext web
- RDF — organization of information in triples *subject-predicate-object* that form graph together
- RDFS — basic vocabulary for RDF, taxonomies
- OWL — $SHOIN(D)$ (description logic with datatypes) written in RDF
Some Opportunities and Challenges

- Semantic search — using semantic information, semantic disambiguation of terms (*return reviewers for a book*)
- Semantic integration, translation between different ontologies (heterogeneous systems)
- Semantic web services (matchmaking, choreography, orchestration), services in general
- Agents able to *reason* about information (their functionality is not hardcoded in program)
- Semantic desktop — organize information in files
- Development of ontologies, the process, management of ontologies and their versions
- Formal features of ontologies, description logics, probabilistic reasoning, proof and trust
- User interfaces, at least for search, query building, browsing
Thank you for attention, questions?

? (if time left, return to Protégé or SWOOP)