Toward Knowledge Formalization

• Acquisition Process
  – Elicit tacit knowledge
  – A set of terms/concepts

• More explicit information
  – Hierarchy and other relations
  – Categorizing (modifiers)
  – Constraints and definitions

• Hierarchical Relations
  – Nodes/Arcs representing a relationship (default IS-A)
  – What IS-A Is and Isn’t: An Analysis of Taxonomic Links in Semantic Networks (Ron Brachman)

leading to some form of knowledge base or ontology…
Ontology

- In Philosophy: the study of the nature of being, becoming, existence, or reality.
- In CS: a knowledge base, i.e., an engineering artefact.

A representation of the shared knowledge for a community

- Used to provide the intended meaning of the vocabulary to describe a certain conceptualisation in a domain of interest
- Usually a vocabulary (i.e., terms) plus explicit characterisations of the assumptions made in interpreting those terms
- Nearly always includes some notion of hierarchical classification (is-a)
- Richer languages allow the definition of classes through description of their characteristics

- Often based on some logic
  ➔ we may use reasoning to help in management & deployment of the knowledge captured in an ontology!
Ontology, taxonomies, terminologies…?

An attempt at clarifying these terms:

**Controlled Vocabulary** = \{terms for concepts\}

**Taxonomy** = CV + hierarchy

**Classification system** = Taxonomy + principles

**Thesaurus** = Taxonomy + more labels

**Terminology** = … + glossary/explanations

**Ontology** = … + logical axioms
 + well-defined semantics
 + reasoning
What is a Taxonomy?

• An organisation of entities
  – typically hierarchical
  – subclass/is-a relationships

• Organisationally Rigid
  – Terms are usually put in their proper place
  – Multiple places for terms?

• Impoverished descriptions
  – Cats are carnivores
    • Why?
    • What is it to be a Carnivore?
    • What if we say something is a Carnivore and a Herbivore?
"The W3C OWL 2 Web Ontology Language (OWL) is a Semantic Web language designed to represent rich and complex knowledge about things, groups of things, and relations between things. OWL is a computational logic-based language such that knowledge expressed in OWL can be reasoned with by computer programs either to verify the consistency of that knowledge or to make implicit knowledge explicit. OWL documents, known as ontologies, can be published in the World Wide Web and may refer to or be referred from other OWL ontologies.

OWL is part of the W3C's Semantic Web technology stack, which includes RDF [RDF Concepts] and SPARQL [SPARQL]."
Requirements from this (1)

“The W3C OWL 2 Web Ontology Language (OWL) is a Semantic Web language designed to represent rich and complex knowledge about things, groups of things, and relations between things. OWL is a computational logic-based language such that knowledge expressed in OWL can be reasoned with by computer programs either to verify the consistency of that knowledge or to make implicit knowledge explicit. OWL documents, known as ontologies, can be published in the World Wide Web and may refer to or be referred from other OWL ontologies.

OWL is part of the W3C's Semantic Web technology stack, which includes RDF [RDF Concepts] and SPARQL [SPARQL].”

From http://www.w3.org/TR/owl-primer/
“The W3C OWL 2 Web Ontology Language (OWL) is a Semantic Web language designed to represent rich and complex knowledge about things, groups of things, and relations between things. OWL is a computational logic-based language such that knowledge expressed in OWL can be reasoned with by computer programs either to verify the consistency of that knowledge or to make implicit knowledge explicit.

OWL documents, known as ontologies, can be published in the World Wide Web and may refer to or be referred from other OWL ontologies.

OWL is part of the W3C's Semantic Web technology stack, which includes RDF [RDF Concepts] and SPARQL [SPARQL].”

From http://www.w3.org/TR/owl-primer/
"The W3C OWL 2 Web Ontology Language (OWL) is a Semantic Web language designed to represent rich and complex knowledge about things, groups of things, and relations between things. OWL is a computational logic-based language such that knowledge expressed in OWL can be reasoned with by computer programs either to verify the consistency of that knowledge or to make implicit knowledge explicit. OWL documents, known as ontologies, can be published in the World Wide Web and may refer to or be referred from other OWL ontologies.

OWL is part of the W3C's Semantic Web technology stack, which includes RDF [RDF Concepts] and SPARQL [SPARQL]."

From http://www.w3.org/TR/owl-primer/
Expressive: Ontologies versus Taxonomies

- Taxonomy: hierarchy of is-a/subsumption relationships
- Ontology can represent rich and complex knowledge about things, groups of things, and relations between things:
  - Knowledge about things:
    - Bob is a Calf
    - Mary is Bob’s Mother
  - Knowledge about groups of things and relations between things:
    - Definitions:
      - A Herbivore is an an Animal that eats only Plants.
      - A Calf is a Young Cow
      - Cows are Herbivores
    - Constraints:
      - Carnivores are not Herbivores (and vice versa)
      - Calfs are Playful and drink some Milk
      - being-a-daughter-of implies being-a-child-of
  - Implicit knowledge in the above:
    - Herbivores eat only Plants
    - Bob is Playful, Young, and eats only Plants
    - ...

OWL: Syntax and Semantics

• OWL is a (formal) language, so we consider its
  – syntax:
    • what is/isn’t a legal OWL (class/property) expression/axiom/ontology/…?
    • what can an OWL parser accept?
    • should be web compatible!
    • see COMP60332 for syntax of logics!
  
  – semantics:
    • what does an OWL (class/property) expression/axiom/ontology… stand for/mean?
    • what can we conclude from an OWL ontology?
    • should be based on logic - but which?
From the OWL 2 Primer

An Overview

We concentrate on this

Ontologies

Documents

parse

serialise

Semantics

interpret

Direct Semantics

correspondence theorem (for DL subset)

RDF-Based Semantics
OWL Syntax: entities

Entities

• are basic building blocks of an OWL ontology
• fall into 3 main categories:
  – **Class Names**:  
    • e.g., Animal, Person, Idea, Table, Grass, Water  
    • stand for sets of things
  – **Property Names**:  
    • e.g., eats, likes, hasPart, hasChild, hasParent, isMarriedTo  
    • stand for relations between things
  – **Individual Names**:  
    • e.g., Peter, Paul, Mary  
    • stand of things
Descriptions (aka class expressions) stand for sets of elements.

Examples:
- Animal that eats only Animal
- eats some (not Animal)
- not (eats only Animal and some Animal)

```
description ::= conjunction 'or' conjunction { 'or' conjunction }
   | conjunction
conjunction ::= classIRI 'that' [ 'not' ] restriction
   { 'and' [ 'not' ] restriction }
   | primary 'and' primary { 'and' primary }
   | primary
primary ::= [ 'not' ] ( restriction | atomicClass )
restriction ::= Property 'some' primary
   | Property 'only' primary
atomicClass ::= [A-Z][a-zA-Z]* (in camel case)
Property ::= [a-z][a-zA-Z]* (in camel case)
```

Grammar is a slightly modified subset of the one given in: http://www.w3.org/TR/owl2-manchester-syntax/
OWL Syntax: axioms

- **Axioms** (aka propositions, statements)
  - can be true or false
  - are often formulated in a **frame**
- **Examples**
  - Class: CarnivorousAnimal EquivalentTo: Animal that eats only Animal
  - Class: Cow SubClassOf: eats some (not Animal)
  - Class: ConfusedCow SubClassOf: not (eats only Animal and some Animal)

```plaintext
classFrame ::= 'Class:' atomicClass
              { 'Annotations:'    annotation { ',' annotation } 
                | 'SubClassOf:'    description { ',' annotation } 
                | 'EquivalentTo:'  description { ',' annotation } }
```
An **OWL ontology** is a collection of axioms, which is the **imports closure** of an OWL document, which is in one of the OWL syntaxes [https://www.w3.org/TR/owl2-syntax/](https://www.w3.org/TR/owl2-syntax/).

An **OWL axiom** takes one of the following forms:

- Class Frame (see above)
- C SubClassOf: D (**subclass**)
- C EquivalentTo: D (**class equivalence**)
- R SubPropertyOf: S (**subproperty**)
- R EquivalentTo: S (**property equivalence**)
- ...
- x Type: C (**class instantiation**)
- x R y (**property instantiation**)

where

- C, D are **class expressions**
- R is a **property expression**

**Note:** OWL doesn’t make this TBox/ABox distinction, but Protégé & DL do and I like it.

**TBox**

**ABox**

built using OWL’s constructors (see above)
Exploring Benefits of Axioms

- E.g., Omnivorous
  - Annotations: comment "Carnivorous and Herbivorous"
  - has no meaning
  - so let’s be explicit:
    • add definition in class description
    • run reasoner
    • check inferred class hierarchy

⇒ our definition was wrong!
Exploring Benefits of Axioms II

• E.g., Cows
  – Annotations:
    comment “Animal that eats only Plants”
  – has no meaning
  – so let’s be explicit:
    • add definition in class description
    • run reasoner
    • check inferred class hierarchy

➢ our class hierarchy is improved: Cows are indeed herbivores!
First Benefits of Axioms & Reasoner

• Links/Sub-Super-class relations/Taxonomy for “free”
  – Tools make implicit links explicit
  – We don’t have to encode every link ourselves
  – Different modality
    • Instead of is-a/subsumption relations...focus on meanings
    • ...we can think local rather than global

• Verification
  – Definitions have consequences
    • May change links:
      – wrong definitions may cause wrong links
      – links can be so wrong they are obviously wrong
Finally: OWL 2 Semantics

- …here we concentrate on “Direct Semantics”, “semantics” for short
- Is defined in terms of an interpretation
  - like in First Order Logic
- and comes in 3 stages:
  1. what do classes/properties/individuals stand for
     a. for names
     b. for expressions
  2. what does it mean for an interpretation to satisfy an
     a. axiom
     b. ontology
  3. what does it mean for an
     a. ontology to entail an axiom
     b. ontology to be consistent
     c. ontology to be coherent
     d. …or what is the inferred class hierarchy
Why Semantics? Isn’t meaning obvious?

- The **semantics** of a language can tell us **precisely** how to interpret a complex expression.
- Well defined semantics is **vital** to support machine interpretability
  - it removes ambiguities in the interpretation of the descriptions
  - i.e., all **tools** agree on their behaviour/give the same results & answers
  - …semantics acts as partial **specification** for tool developers

Is every Y and X (or only most/normally)?
Can a Y be a Z?
Can there be an X that’s neither a Y nor a Z?
…
An interpretation is a pair <\(\Delta, I\)>, where
- \(\Delta\) is the domain (a non-empty set)
- \(I\) is an interpretation function that maps each
  - class name \(A\) to a set \(A^I \subseteq \Delta\)
    ...we call \(A^I\) the extension of \(A\) in \(I\)
  - property name \(R\) to a binary relation \(R^I \subseteq \Delta \times \Delta\)
    ...if \((e, f) \in R^I\) we call \(f\) an R-filler of \(e\) in \(I\)
  - individual name \(i\) to an element \(i^I \in \Delta\)
    ...if \(i^I \in A^I\) we say that \(i\) is an instance of \(A\) in \(I\)
- ...and we can draw interpretations!
  - \(\Delta = \{v, w, x, y, z\}\)
  - \(A^I = \{v, w, x\}\)
  - \(B^I = \{x, y\}\)
  - \(C^I = \{w, y\}\)
  - \(R^I = \{(v, w), (v, x), (y, x), (x, z)\}\)

Like in FOL!
OWL 2 Semantics: an interpretation (1a)

- An **interpretation** is a pair $\langle \Delta, I \rangle$, where
  - $\Delta$ is the **domain** (a non-empty set)
  - $I$ is an **interpretation function** that maps each
    - **class name** $A$ to a set $A^I \subseteq \Delta$
      ...we call $A^I$ the **extension** of $A$ in $I$
    - **property name** $R$ to a binary relation $R^I \subseteq \Delta \times \Delta$
      ...if $(e,f) \in R^I$ we call $f$ an $R$-**filler** of $e$ in $I$
    - **individual name** $i$ to an element $i^I \in \Delta$
      ...if $i^I \in A^I$ we say that $i$ is an **instance of** $A$ in $I$
  - ...and we can draw interpretations!
    - $\Delta = \{v, w, x, y, z\}$
    - $A^I = \{v, w, x\}$
    - $B^I = \{x, y\}$
    - $C^I = \{w, y\}$
    - $R^I = \{(v, w), (v, x), (y, x), (x, z)\}$
Interlude: Drawing Interpretations

- is really important for understanding interpretations and hence semantics of OWL
- make sure you understand that you need **arrows** (not just lines)
- possibly with **labels** for property names
- what nodes and their labels mean
- check/re-read the definition:
  - what size can the domain have?
  - what size are extensions?
  - which restrictions are on them?
  - what’s a really small interpretation?
  - what’s a really big interpretation?

---

An interpretation is a pair \(<\Delta, I>\), where
- \(\Delta\) is the **domain** (a non-empty set)
- \(I\) is an **interpretation function** that maps each
  - **class** name \(A\) to a set \(A^I \subseteq \Delta\)
  - **property** name \(R\) to a binary relation \(R^I \subseteq \Delta \times \Delta\)
  - **individual** name \(i\) to an element \(i^I \in \Delta\)
### OWL 2 Semantics: an interpretation (1b)

**Interpretation of class expressions:**

<table>
<thead>
<tr>
<th>Constructor</th>
<th>Example</th>
<th>Interpretation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class name</td>
<td><em>Human</em></td>
<td><em>Human</em> $^\downarrow \subseteq \Delta$</td>
</tr>
<tr>
<td>Thing</td>
<td>n/a</td>
<td>$\Delta$</td>
</tr>
<tr>
<td>Nothing</td>
<td>n/a</td>
<td>$\emptyset$</td>
</tr>
<tr>
<td>and</td>
<td><em>Human and Male</em></td>
<td><em>Human</em> $^\downarrow \cap \text{Male}^\downarrow$</td>
</tr>
<tr>
<td>or</td>
<td><em>Doctor or Lawyer</em></td>
<td><em>Doctor</em> $^\downarrow \cup \text{Lawyer}^\downarrow$</td>
</tr>
<tr>
<td>not</td>
<td>not <em>Male</em></td>
<td>$\Delta \setminus \text{Male}^\downarrow$</td>
</tr>
</tbody>
</table>
OWL 2 Semantics: an interpretation (1b)

Interpretation of more class expressions:

<table>
<thead>
<tr>
<th>Constructor</th>
<th>Example</th>
<th>Interpretation</th>
</tr>
</thead>
<tbody>
<tr>
<td>some</td>
<td>hasChild some Lawyer</td>
<td>{e \in \Delta \mid \text{there is some } f: (e,f) \in hasChild^I \text{ and } f \in Lawyer^I}</td>
</tr>
<tr>
<td>only</td>
<td>hasChild only Doctor</td>
<td>{e \in \Delta \mid \text{for all } f \in \Delta: if } (e,f) \in hasChild^I \text{ then } f \in Doctor^I}</td>
</tr>
<tr>
<td>min</td>
<td>hasChild min 2 Tall</td>
<td>{e \in \Delta \mid \text{there are at least 2 } f \in \Delta \text{ with } (e,f) \in hasChild^I \text{ and } f \in Tall^I}</td>
</tr>
<tr>
<td>max</td>
<td>hasChild max 2 Tall</td>
<td>{e \in \Delta \mid \text{there are at most 2 } f \in \Delta \text{ with } (e,f) \in hasChild^I \text{ and } f \in Tall^I}</td>
</tr>
</tbody>
</table>
Interpretation of Classes - Examples

- $\Delta = \{v, w, x, y, z\}$
- $A^I = \{v, w, x\}$
- $B^I = \{x, y\}$
- $R^I = \{(v, w), (v, x), (y, x), (x, z)\}$

- $(\text{not } B)^I = $
- $(A \text{ and } B)^I = $
- $((\text{not } A) \text{ or } B)^I = $
- $(R \text{ some } B)^I = $
- $(R \text{ only } B)^I = $
- $(R \text{ some } (R \text{ some } A))^I = $
- $(R \text{ some } \neg(A \text{ or } B))^I = $
- $(R \text{ min } 1.\text{Thing})^I = $
- $(R \text{ max } 1.\text{Thing})^I = $
OWL 2 Semantics: an interpretation satisfying ... (2)

- An interpretation $I$ satisfies an axiom $\alpha$ if:
  - $\alpha = C \text{ SubClassOf: } D$ and $C^I \subseteq D^I$
  - $\alpha = C \text{ EquivalentTo: } D$ and $C^I = D^I$
  - $\alpha = P \text{ SubPropertyOf: } S$ and $P^I \subseteq S^I$
  - $\alpha = P \text{ EquivalentTo: } S$ and $P^I = S^I$
  - ...
  - $\alpha = x \text{ Type: } C$ and $x^I \in C^I$
  - $\alpha = x \text{ R y }$ and $(x^I, y^I) \in R^I$

- $I$ satisfies an ontology $O$ if $I$ satisfies every axiom $A$ in $O$
  - If $I$ satisfies $O$, we call $I$ a **model of** $O$

- See how the axioms in $O$ constrain interpretations:
  - ✓ the more axioms you add to $O$, the fewer models $O$ has
  - ...they do/don’t hold/are(n’t) satisfied in an ontology
    - in contrast, a class expression $C$ **describes a set** $C^I$ in $I$

Check OWL 2 Direct Semantics for more!!!
Draw & Match Models to Ontologies!

O1 = {}
O2 = {a:C, b:D, c:C, d:C}
O3 = {a:C, b:D, c:C, b:C, d:E}
O4 = {a:C, b:D, c:C, b:C, d:E, D SubClassOf C}
O5 = {a:C, b:D, c:C, b:C, d:E, a R d, D SubClassOf C, D SubClassOf S some C}
O6 = {a:C, b:D, c:C, b:C, d:E, a R d, D SubClassOf C, D SubClassOf S some C, C SubClassOf R only C}

I₁:
Δ = {v, w, x, y, z}
CI = {v, w, y}
DI = {x, y} EI = {}
RI = {(v, w), (v, y)} SI = {}
a₁ = v  b₁ = x
c₁ = w  d₁ = y

I₂:
Δ = {v, w, x, y, z}
CI = {v, w, y}
DI = {x, y} EI = {y}
RI = {(v, w), (v, y)} SI = {}
a₁ = v  b₁ = x
c₁ = w  d₁ = y

I₃:
Δ = {v, w, x, y, z}
CI = {x, v, w, y}
DI = {x, y} EI = {y}
RI = {(v, w), (v, y)} SI = {}
a₁ = v  b₁ = x
c₁ = w  d₁ = y

I₄:
Δ = {v, w, x, y, z}
CI = {x, v, w, y}
DI = {x, y} EI = {y}
RI = {(v, w), (v, y)} SI = {(x,x), (y,x)}
a₁ = v  b₁ = x
c₁ = w  d₁ = y
The world in an ontology: ontology as surrogate

Our view of our domain

World

Ontology O

Daisy: Cow
Cow SubClassOf Animal

Mary: Person
Person SubClassOf Animal

Z123ABC: Car
Mary drives Z123ABC

Model of O

Should agree with our view

Δ
Let $O$ be an ontology, $\alpha$ an axiom, and $A, B$ classes, $b$ an individual name:

- $O$ is **consistent** if there exists some model $I$ of $O$
  - i.e., there is an interpretation that satisfies all axioms in $O$
  - i.e., $O$ isn’t self contradictory
- $O$ **entails** $\alpha$ (written $O \models \alpha$) if $\alpha$ is satisfied in all models of $O$
  - i.e., $\alpha$ is a consequence of the axioms in $O$
- $A$ is **satisfiable** w.r.t. $O$ if $O \not\models A \text{ SubClassOf Nothing}$
  - i.e., there is a model $I$ of $O$ with $A^I \neq \{\}$
- $b$ is an **instance of** $A$ w.r.t. $O$ (written $O \models b : A$) if $b^I \subseteq A^I$ in every model $I$ of $O$

**Theorem:**

1. $O$ is consistent iff $O \not\models \text{ Thing SubClassOf Nothing}$
2. $A$ is satisfiable w.r.t. $O$ iff $O \cup \{n:A\}$ is consistent (where $n$ doesn’t occur in $O$)
3. $b$ is an instance of $A$ in $O$ iff $O \cup \{b: \text{ not}(A)\}$ is not consistent
4. $O$ entails $A \text{ SubClassOf } B$ iff $O \cup \{n:A\text{ and not}(B)\}$ is inconsistent
Let \( O \) be an ontology, \( \alpha \) an axiom, and \( A, B \) classes, \( b \) an individual name:

- \( O \) is **consistent** if there exists some model \( I \) of \( O \)
  - i.e., there is an interpretation that satisfies all axioms in \( O \)
  - i.e., \( O \) isn’t self contradictory
- \( O \) **entails** \( \alpha \) (written \( O \vDash \alpha \)) if \( \alpha \) is satisfied in all models of \( O \)
  - i.e., \( \alpha \) is a consequence of the axioms in \( O \)
- \( A \) is **satisfiable** w.r.t. \( O \) if \( O \vDash A \text{ SubClassOf } \) Nothing
  - i.e., there is a model \( I \) of \( O \) with \( A^I \neq \{\} \)
- \( b \) is an **instance of** \( A \) w.r.t. \( O \) if \( b^I \subseteq A^I \) in every model \( I \) of \( O \)

- \( O \) is **coherent** if every class name that occurs in \( O \) is satisfiable w.r.t \( O \)
- **Classifying** \( O \) is a reasoning service consisting of
  1. testing whether \( O \) is consistent; if yes, then
  2. checking, for each pair \( A, B \) of class names in \( O \) plus \( \text{Thing, Nothing} \) \( O \vDash A \text{ SubClassOf } B \)
  3. checking, for each individual name \( b \) and class name \( A \) in \( O \), whether \( O \vDash b: A \)

...and returning the result in a suitable form: \( O \)’s **inferred class hierarchy**
OWL Reasoners and Protégé

- **OWL reasoners**
  - implement **decision procedures** for consistency/entailments, and classify ontologies

- **Protégé**
  - interacts with reasoners via the OWL API
  - shows results as
    - inferred class hierarchy where
    - unsatisfiable classes are red and you get a
    - warning (red triangle) if O is inconsistent

- **OWL reasoners**
  - implement highly optimised algorithms which decide
    - complex logical decision problems:
      - between PTime for OWL 2 EL profile to
      - N2ExpTime-hard for OWL 2…
    - via (hyper)-tableau algorithm or other
    - …later more
Complete details about OWL

• here, we have concentrated on some **core** features of OWL, e.g., no
  • domain, range axioms
  • SubPropertyOf, InverseOf
  • datatype properties
  • …
• we expect you to look these up!

• OWL is defined via a **Structural Specification**
• [http://www.w3.org/TR/owl2-syntax/](http://www.w3.org/TR/owl2-syntax/)
• Defines language independently of concrete syntaxes
• Conceptual structure and abstract syntax
  • UML diagrams and functional-style syntax used to define the language
  • Mappings to concrete syntaxes then given.
• The structural specification provides the foundation for implementations (e.g. OWL API as discussed later)
**OWL Resources**

- The OWL Technical Documentation is all available online from the W3C site.
  
  [http://www.w3.org/TR/owl2-overview/](http://www.w3.org/TR/owl2-overview/)

All the OWL documents are relevant; we recommend in particular the
- Overview
- Primer
- Reference Guide and
- Manchester Syntax Guide

- An introduction to OWL for people who know logic at
  [http://owl.cs.manchester.ac.uk/about/orientation/a-logics-perspective/](http://owl.cs.manchester.ac.uk/about/orientation/a-logics-perspective/)

- Our Ontogenesis Blog at
Models of $O$

$Sushi \text{ SubClassOf } Food$ and
$contains \text{ some Rice}$

$ChSushi \text{ EquivalentTo } Sushi$ and
$contains \text{ some Chocolate}$

$Z123: Sushi$
$Z123 \text{ contains } Z234$
$Z243: Chocolate$

Assumption: you are knowledge engineers, but not domain experts!