Ontologies in First Order Logic

The Power of First Order Logic

The syntax, semantics and proof-theory of pure first order logic (FOL) offer us a **general, flexible** and **powerful** framework for knowledge representation.

Chapter 10 of AIMA shows the power of FOL for representing:

- ontologies and taxonomic information
- physical composition
- measurements
- events, actions, processes, plans, time, space, causality

We will only cover **ontologies and taxonomic information** and **physical composition** in this lecture.

Subsets of FOL

Because FOL is very general and it is based on very primitive concepts (constants, variables, function symbols, predicates and quantifiers), some people have chosen to study **subsets of FOL** that are most appropriate for higher-level abstractions such as **taxonomies, time, space** etc.

In addition, by going to subsets of FOL, we try to **avoid undecidability**, and even better, have a subset for which **efficient implementations of reasoning algorithms exist** (as e.g., in **Datalog**).

Ontologies

An **ontology** is a formal, explicit, shared specification of a conceptualization of a domain (Gruber, 1993).

Conceptualization: the objects, concepts, and other entities that are assumed to exist in some area of interest and the relationships that hold among them. A conceptualization is an abstract, simplified view of the world that we wish to represent for some purpose.

The term ontology is borrowed from Philosophy, where ontology is a systematic account of existence (what things exist, how they can be differentiated from each other etc.).

Today the word **ontology is a synonym for (shared!) knowledge base**.

Formal Languages for Ontologies

Ontologies are typically expressed in some **formal, logic-based language** e.g., FOL.

The literature also offers us special formalisms for defining ontologies that contain mainly taxonomic knowledge:

- Semantic networks
- Frames
- Description logics
- RDF, RDF(S) and OWL 2 for ontologies in the **Semantic Web**.

Formal Languages for Ontologies (cont'd)

You can think about these formalisms as being **object-oriented logics**:

- They have special constructs for representing knowledge about **individuals** (or **objects**), **categories** (or **classes**) and **relationships** (or **roles**).

  Categories are organized into **taxonomies**.
- They have special reasoning methods to deal with these constructs.

How to Learn More

If you attend my postgraduate course “Knowledge Technologies”, you will learn about:

- Resource Description Framework (RDF)
- RDF Schema
- Description logics
- OWL 2

See `http://cgi.di.uoa.gr/~pms509/`.

Taxonomies

- Taxonomies have been used profitably for centuries in various **scientific and technical fields** (biology, medicine, library science, engineering etc.).
- Taxonomic information plays a **central role in other Computer Science areas** e.g., programming languages, databases and software engineering.
- Taxonomies are very important in **modern applications**: information retrieval in the Web, information integration, knowledge management, e-commerce, e-science, e-government etc.

Examples of Very Large Ontologies

Recently, there has been big progress in the construction of **very large ontologies** and their use in building intelligent applications.

We will summarize some of these efforts in the next lecture.

Upper vs. Domain Ontologies

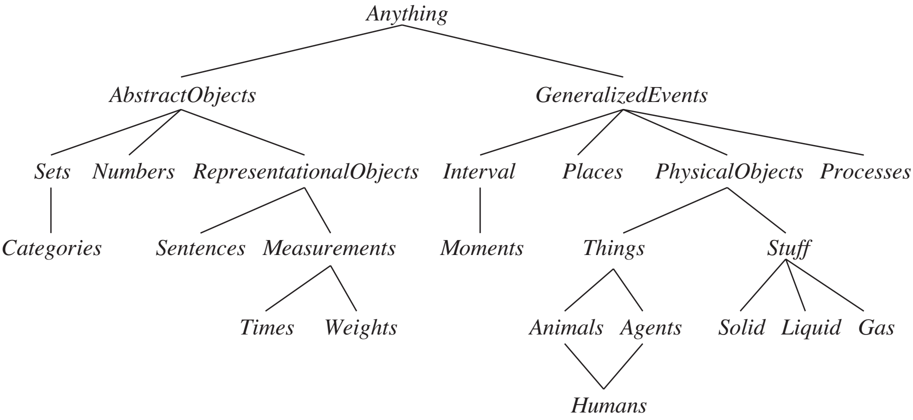
- **Upper ontologies** are ontologies limited to high-level concepts that are general enough to address a broad range of domain areas.
- **Domain ontologies** are ontologies that formalize a specific domain (e.g., the administrative regions of Greece as defined in the recent Kallikratis law). Domain ontologies are built by specializing and extending upper ontologies.

Upper Ontologies

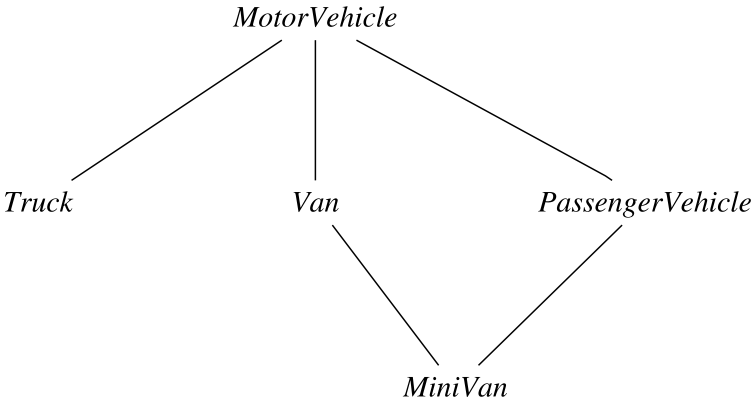
There have been various upper ontologies defined:

- SUMO (`http://www.ontologyportal.org/`)
- DOLCE (`http://www.loa-cnr.it/DOLCE.html`)
- The Cyc upper ontology (`http://www.cyc.com/cycdoc/upperont-diagram.html`)
- ...

The AIMA Upper Ontology



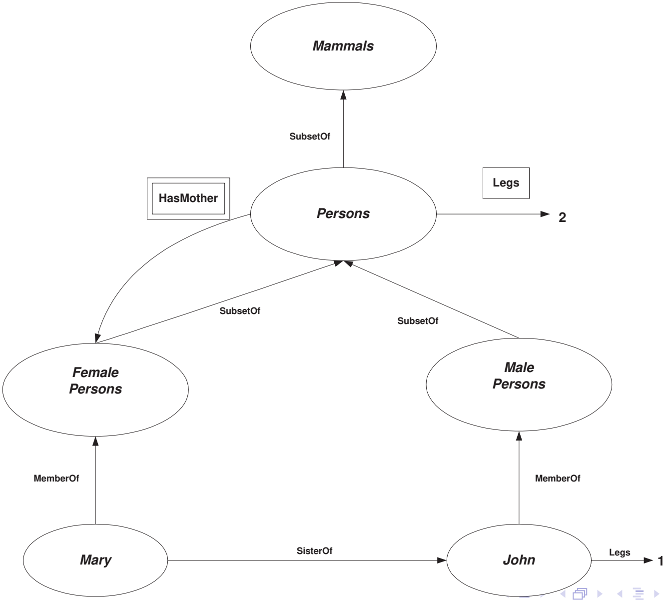
A Domain Ontology for Motor Vehicles



Taxonomic Information in FOL

Now let us see in detail how to represent taxonomic information in FOL!

Example



Mammals
SubsetOf
HasMother
Persons
Legs
2
SubsetOf
SubsetOf
Female Persons
Male Persons
MemberOf
MemberOf
Mary
SisterOf
John
Legs
1

Categories in FOL

FOL offers us two ways to talk about **categories (or classes) of objects**:

- **Unary predicate symbols.** For example:
  $Person(x)$, $FemalePerson(x)$
- **Constant symbols** (through **reification**). For example:
  $Persons$, $FemalePersons$.
  In this case, we also need predicates for **membership** and **subclass**: $MemberOf$ and $SubsetOf$.

Both of the above ways are needed! We will see why below.

Properties or Relationships in FOL

In FOL we can use the following methods to talk about **properties of an object or relationships among two objects:**

- **Binary predicate symbols.** For example: $SisterOf(x, y), HasMother(x, y), Legs(x, n)$
- **Function symbols** (if a property is functional). For example: $BiologicalFatherOf(x) = y, Legs(x) = n$

Membership of an Object in a Category

We have two ways to express that an object is a **member** of a category.

Example:

$$FemalePerson(Mary)$$

$$MemberOf(Mary, FemalePersons)$$

In the second option, predicate $MemberOf$ has to be axiomatized appropriately. See Chapter 8 for an axiomatization of the concept of sets.

Properties of Objects - Relationships

Example of asserting that two objects are related by a **relationship**:

$$SisterOf(Mary, John)$$

Examples of asserting that a **property** of an object has a **value**:

$$Legs(John, 2), \text{ or by using a function symbol } Legs(John) = 2$$

$$HasTelNo(John, 6975532334)$$

$$HasTelNo(John, 2107943525)$$

Other Examples of Properties of Objects

Consider the domain of basketball:

$$Basketball(BB_{123})$$

$$Shape(BB_{123}, Round) \text{ or } Shape(BB_{123}) = Round \text{ or } Round(BB_{123})$$

$$Diameter(BB_{123}, 9.5'')$$

Subcategory/Subclass Relations

We have two ways to express that a category is a **subcategory (subclass)** of another category:

$$(\forall x)(BasketBall(x) \Rightarrow Ball(x))$$

$$SubsetOf(BasketBalls, Balls)$$

For the second option, predicate *SubsetOf* has to be axiomatized appropriately.

Necessary Conditions for Membership in a Category

Sometimes we want to express that all members of a category have some properties (i.e., these are **necessary conditions** for being a member of the category).

Examples:

$$(\forall x)(BasketBall(x) \Rightarrow Shape(x, Round))$$

$$(\forall x)(MemberOf(x, BasketBalls) \Rightarrow Shape(x, Round))$$

Sufficient Conditions for Membership in a Category

We can also express that all members of a category can be recognized by some properties (i.e., these are **sufficient conditions** for being a member of the category).

Example:

$$(\forall x)(Orange(x) \land Round(x) \land Diameter(x) = 9.5'' \land$$

$$MemberOf(x, Balls) \Rightarrow MemberOf(x, BasketBalls))$$

Categories and Definitions

Some categories can be given both necessary and sufficient conditions or **"if and only if" definitions**.

Example: An object is a **triangle** if and only if it is a polygon with three sides.

Natural kind categories cannot be defined in this way.

Example: Try to define **tomatoes** with an "if and only if" definition.

For natural kind categories, we can write down "if and only if" definitions that hold for **typical instances.**

Examples

Definition. An object is a **triangle** if and only if it is a polygon with three sides.

$$(\forall x)(MemberOf(x, Triangle) \Leftrightarrow (MemberOf(x, Polygon) \wedge NoOfSides(x, 3))$$

Any objections to the formula $NoOfSides(x, 3)$?

Categories are Objects Too

A category can be a member of a **category of categories.**

Example:

$$MemberOf(Dogs, DomesticatedSpecies)$$

When we use the term class instead of category, we talk about a class being a member of a **meta-class.**

This cannot be done without having a constant symbol for each category.

Categories are Objects Too (cont'd)

We might want to assert that a category as a whole has some **property**.

Examples:

$$ReproductiveCycle(FemaleDogs, 6months)$$

$$HasCardinality(MyDogs, 3)$$

Categories are Objects Too (cont'd)

Can we have **categories of categories of categories**? Are they useful?

In various OO modeling frameworks (e.g., Telos) we have 4+ levels of data modeling:

- Instances (e.g., *John*)
- Classes (e.g., *Person*)
- Meta-classes (e.g., the class of all classes with no instances).
- Meta-meta-classes (e.g., the class of all meta-classes we have defined).

Categories are Objects Too (cont'd)

In the Information Resource Dictionary Standard (IRDS) we have 4 levels of data description:

- Level 1: Application data (e.g., code for a function).
- Level 2: Data dictionary for application data (e.g., the signature of the function: name, names of arguments, types of arguments, type of result).
- Level 3: Schema of the data dictionary (e.g., a schema for functions).
- Level 4: Different types of IRDS schemas (e.g., a schema for development purposes vs. a schema for requirement-modelling purposes).

You can model this hierarchy using meta-classes.

Other Set-Theoretic Relations Among Categories

Often we want to say that two categories are **disjoint** or that they form an **exhaustive decomposition** of some other category or that they form a **partition** of some other category.

Examples:

$$Disjoint(\{Animals, Vegetables\})$$

$$ExhaustiveDecomposition(\{Americans, Canadians, Mexicans\},$$

$$NorthAmericans)$$

$$Partition(\{Males, Females\}, Animals)$$

Relations Among Categories (cont'd)

The three predicates used above can be defined as follows:

$$(\forall s)(Disjoint(s) \Leftrightarrow$$

$$(\forall c_1, c_2)(c_1 \in s \land c_2 \in s \land c_1 \neq c_2 \Rightarrow Intersection(c_1, c_2) = \{\}))$$

$$(\forall s, c)(ExhaustiveDecomposition(s, c) \Leftrightarrow$$

$$(\forall i)(i \in c \Rightarrow (\exists c_2)(c_2 \in s \land i \in c_2)))$$

$$(\forall s, c)(Partition(s, c) \Leftrightarrow Disjoint(s) \land ExhaustiveDecomposition(s, c))$$

In the above formulas we use $\in$ instead of *MemberOf*. Predicate *Intersection* needs to be axiomatized appropriately.

Discussion

The method of using unary predicates for representing knowledge about categories is **weaker** than the method of using constants.

In other words, the latter method can represent everything that the former one can and more (as we saw in the example with categories of categories).

The two methods can co-exist.

Physical Composition

The idea that one object is **part of** another is an important one in many domains (e.g., engineering design, e-commerce catalogs, human anatomy). We use the general predicate *PartOf* to represent such information.

Example:

$$PartOf(Athens, Greece),\ \ PartOf(Greece, WesternEurope)$$

$$PartOf(WesternEurope, Europe),\ \ PartOf(Europe, Earth)$$

Physical Composition (cont'd)

The relation *PartOf* is **irreflexive** and **transitive:**

$$(\forall x)(\neg PartOf(x, x))$$

$$(\forall x, y, z)(PartOf(x, y) \wedge PartOf(y, z) \Rightarrow PartOf(x, z))$$

Thus we can conclude: $PartOf(Athens, Earth)$.

Physical Composition (cont'd)

Categories of **composite** objects are often characterized by the **structure** of those objects i.e., the parts and how the parts relate to the whole.

Example: How can we define a biped?

Defining a Biped

$$(\forall a)(Biped(a) \Leftrightarrow$$
$$(\exists l_1, l_2, b)(Leg(l_1) \wedge Leg(l_2) \wedge Body(b) \wedge$$
$$PartOf(l_1, a) \wedge PartOf(l_2, a) \wedge PartOf(b, a) \wedge$$
$$Attached(l_1, b) \wedge Attached(l_2, b) \wedge$$
$$l_1 \neq l_2 \wedge (\forall l_3)(Leg(l_3) \Rightarrow (l_3 = l_1 \vee l_3 = l_2))))$$

Discussion

Question: Now that we know how to represent taxonomies in FOL, shall we use it in each application we encounter?

For example, **shall we use it to represent human anatomical knowledge** as capture by the foundational model of anatomy (see `http://sig.biostr.washington.edu/projects/fm/AboutFM.html`)?

Discussion (cont'd)

Answer: Certainly! But it is not such a good idea.

It is a better to use **modern logics and ontology languages** that have been especially designed for this.

Description logics and the **Web ontology language OWL 2** enable us to represent taxonomical knowledge much more easily.

Readings

- AIMA book, Chapter 10.