324: AI (Real-world Knowledge Representation and Reasoning) **Spring 2015**

Lecture 9: Feb. 05, 2015

Lecturer: K.R. Chowdhary *: Professor of CS (VF)*

9.1 Introduction

Ontology is a theory about nature of being or the kinds of existent. Task of intelligent systems in using computers is to formally represent these existents. A body of formally represented knowledge is based on conceptualization. A concept is some thing we can form a mental image of. Conceptualization consists of a set of objects, concepts, and other entities about which knowledge is being expressed and of relationships that hold among them. Every knowledge model is committed to some conceptualization, implicitly or explicitly.

An ontology is a systematic arrangement of all of the important categories of objects or concepts which exist in some field of discourse, showing the relations between them. When complete, an ontology is a categorization of all of the concepts in some field of knowledge, including the objects and all of the properties, relations, and functions needed to define the objects and specify their actions. A simplified ontology may contain only a hierarchical classification (called taxonomy) showing the type *subsumption* relations between concepts in the field of discourse. An ontology may be visualized as an abstract graph with nodes and labeled arcs representing the objects and relations, as shown in figure 9.1, which is the upper level of the CYC project hierarchy.

The CYC Project is an example of ontology, it contains more than 10,000 concept types used in the rules and facts encoded in the knowledge base. A concept is some thing we can form a mental image of. At the top of the hierarchy is the *Thing* “concept”, which does not have any properties of its own. The hierarchy under *Thing* is quite tangled. Not all the subcategories are exclusive. In general, Thing is partitioned in three ways: First is *Represented Thing* versus *Internal Machine Thing*. Every CYC category must be an instance of one and only one of these sets. Internal-Machine-Thing is anything that is local to the platform CYC is running on (strings, numbers, and so on). *Represented-Thing* is everything else.

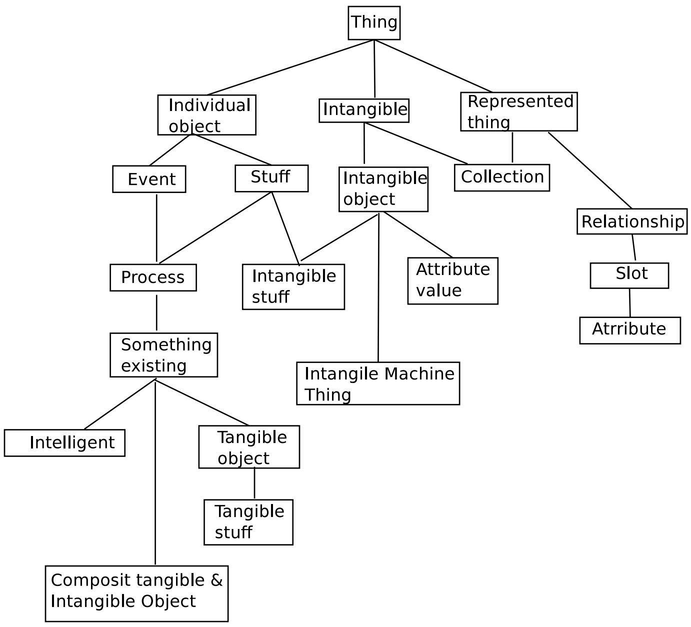

Figure 9.1: World Ontology.

Different degrees of aggregation of concepts may be used, and distinctions of importance for one purpose may be of no concern for a different purpose. For example, *class* is collection of students, and when all of them are playing in ground we call it a *team*; when both parents are teaching they are called *teachers*, and when traveling together they are *passengers*.

9.2 Ontology Structures

When compared with FOPL for knowledge representation and reasoning, in the ontologies we are particular for knowledge *organization* as well as *contents*. This approach has generalization, combined with exceptions. For example, “all birds fly” can be a rule. But, there are some birds which do not fly, for example, *penguin*. This is an exception. We should be able to add this concepts of knowledge, not as exception, but as extension. This requires categorization of objects.

9.2.1 Language and Knowledge

The figure 9.2 shows the interrelationship of language and logic with ontology, and how each of these are related to commonsense, reasoning, and type checking. The language, ontology, and logic are representations, where as the corresponding blocks, i.e., common-sense, type-checking, reasoning, are respectively, the applications. Consider the object “ball”, we identify it as a member in the collection of balls, by type checking; we say plants have life, they belong to the category of all the living beings (type-check). At lowest level in figure 9.2 is *logic*, making use of binary operators. This logic helps in reasoning process. The common-sense is context level knowledge, which is essential for language understanding. For example, the phrase - “a good play”, while in a cricket ground is understood as good shot, while in a theater, it is taken as performance by an artist. We resolve the phrase “a good play” by context.

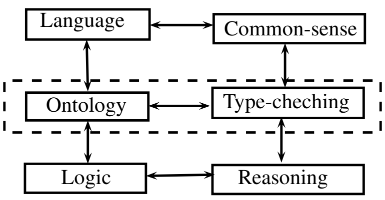

Figure 9.2: Language, Logic, and Ontology.

There is a close relationship between natural language understanding (NLU) vs. knowledge representation and reasoning. In other words, the relationship between *language* and *knowledge*. In fact, ‘understanding is reasoning’paradigm, as it has become quite clear by now that understanding natural language is, for the most part, a commonsense reasoning process at the pragmatic level. As an example, to illustrate this strong interplay between language understanding and commonsense reasoning, consider the following:

(a) Rajan defended a jailed activist in every state.

(b) Rajan knows a jailed activist in every state.

Through commonsense, we have no difficulty in inferring for (a) that Rajan supports for the same activist in every state. Perhaps Rajan might have traveled to different states for campaign. Where as in (b), we hardly conceive of a same. It may mean, “there is a jailed activist in every state”, or an activist being jailed in every state. Such inferences lie beyond syntactic and semantic explanations, and are in fact depend on our commonsense knowledge of the world (where we actually live).

As another example, consider resolving of the noun He in the following:

(c) Rajan shot a policeman. He instantly

(i) ran away.

(ii) collapsed.

Clearly, such references must be resolved by commonsense knowledge. For example, typically, when $shot(x, y)$ holds between some x and some y, x is the more likely subject to run away (c(i)), and y is the more likely to collapse (c(ii)). Note, however, that such inferences must always be considered defeasible, since quite often additional information might result in the retraction of previously made inferences. For example, (c(ii)) might be describing a situation in which Rajan, a ten-year old who was shooting for practice, fell down. Similarly, and (c(i)) might actually be describing a situation in which the policeman, upon being slightly injured, tried to run away, may be to escape further injuries!

9.2.2 Levels of Ontologies

In comparing various ontologies, they can be viewed at three different levels: (1) *is-a* taxonomy of concepts, (2) the *internal concept* structure and relation between concepts, and (3) the presence or absence of *explicit axioms*. The figure 9.3 shows that various concepts are related by *is-a* hierarchy relation (a member of relation). Taxonomy is central part of most ontologies, and its organization can vary greatly, for example, all concepts can be in one large taxonomy, or there can be number of smaller hierarchies, or there can be no explicit taxonomy at all.

Although all general-purpose ontologies try to categorize the same world, they are very different at the top level. They also differ in their treatment of basic parts: *things*, *processes*, and *relations*.

The next level of comparison is internal concept structure, which can be realized by *properties* and *roles*. The internal concept relations are the relations, for example, between *bird* vs. *canfly, wings, and feathers*. Concepts in some ontologies are atomic and might not have any properties or roles or any other internal structure associated with them.

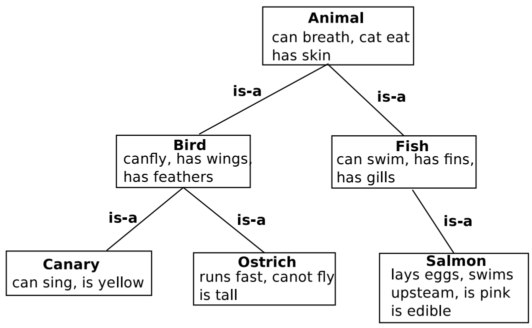

Figure 9.3: Top level Ontology for Animal.

9.2.3 Sowa's Ontology

John Sowa (1997, 1995a) states fundamental principles for ontology design as "distinctions, combinations, and constraints". There are three top-level distinctions: First is *Physical* versus *Information*, or *Concrete* versus *Abstract*. This is a disjoint partition of all the categories in the ontology (see figure 9.4).

The second principle for ontology design is based on combinations. The combinations classify the objects into *firstness* versus *secondness* versus *thirdness*, or *Form* versus *Role* versus *Mediation*. These categories are not mutually exclusive. For example, *Woman* is considered to be a form (firstness) because it can be defined without considering anything outside a person. As a mother, a teacher, or an employee, the same individual would be an example of a role (secondness). These roles represent an individual in relation to another type (a child, a student, an employer). Marriage is a mediation (thirdness) category because it relates several (in this case, two) types together.

The third principle is based on constraints. It is, *continuant* vs. *occurrent*, or *object vs. Process*. Continuants are objects that retain their identity over some period of time; occurrents are processes "whose form is in the state of flux". For example, *Avalanche* is a process, and *Glacier* is an object. Note that this distinction depends on the time scale. On a grand time scale of centuries, Glacier is also a process. These distinctions are combined to generate new categories (figure 9.4).

At a lower level, for example, Script (a computer program, a baking recipe) is a form that represents sequences and is thus defined as *Abstract*, *Form*, *Process*. Also, *History* (an execution of a computer program), which is a proposition that describes a sequence of processes, is then Abstract, Form, or object.

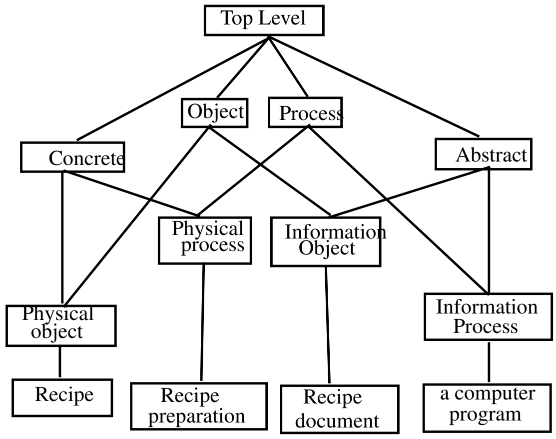

Figure 9.4: John Sowa's Ontology.

9.3 Ontological Engineering

The *knowledge engineering* is process of knowledge representation - identification of task, assemble relevant knowledge, decide vocabulary of predicates, functions and constructs, encode knowledge about domain, encode description of specific problem instance, pose queries to procedures and get answers and debug knowledge-base.

The ontological engineering can be defined on the same line of knowledge engineering. It is related to the creating representation of general concepts - actions, time, physical objects and beliefs.

Ontological engineering comprise a set of activities conducted during the time of: conceptualization, design, implementation and deployment of ontologies. It covers knowledge representation formalisms, development methodology, knowledge sharing and reuse, knowledge management, business process modeling, commonsense knowledge, systematization of domain knowledge, information retrieval from the Internet, standardization, and evaluation. It also gives us design rationale of a knowledge base, helps define the essential concepts of the world of interest, allows for a more disciplined design of a knowledge base, and enables knowledge accumulation. In practice, knowledge of these disciplines helps to:

- Organize the knowledge acquisition process;
- Specify the ontology's primary objective, purpose, granularity, and scope; and
- Build its initial vocabulary and organize taxonomy in an informal or semi-formal way, possibly using an intermediate representation.

Special-purpose languages/tools for implementing ontologies, such as *Ontolingua*, *CycL*, and *LOOM*, use a frame-based formalism, a logic-based formalism, or both. For example, 'Ontolingua'is a frame-based language that uses KIF (Knowledge Interchange Format), a language for publication and knowledge communication that has notation and semantics of an extended version of first-order predicate calculus. It enables writing knowledge-level specifications, independent of particular data or programming languages, and translating knowledge bases from one specialized representation language into another.

9.3.1 Categories and Objects

Classifying objects into categories is important for knowledge organization. Of course interactions take place at individuals levels but relations are created at the level of categories. For example, the sentence, "Basketball team has won", does not refer to a single ball and nor a single player, but a relationship among team members as a group.

The Category also helps in prediction of objects once they are classified. One refers the objects from perceptual inputs, infers category of membership from perceived properties of the objects, then uses category information to predict about object. For example, from its yellow colour, size, shape, we identify that an object is a 'Mango'.

In FOPL there are two choices for categories: *predicates* and *objects*. Following are the examples:

$basketball(b)$;

$member(b, basketballs)$;

$b \in basketballs$; (the object b is a member of category basketball)

$subset(basketballs, balls)$; (category is subclass of another category)

$basketballs \subset balls$.

The category is defined by *members* and *subset* relation. We also note that categories organize the knowledge as well as they help to inherit. Thus, 'mango'∈ 'mangoes', and 'mangoes'⊂ 'fruits'. Thus, mango inherits the properties of 'taste'from fruits (fruits have 'tastes'). Similarly, we have categories: animals, birds, foods, institutions. A "set of institutions" ⊂ "institutions", and MBM ∈ "institutions". The subclass organize the categories into a *taxonomic* hierarchy. The taxonomical hierarchies are common in government, military, and other organizations.

Following are representations of taxonomies:

$(x \in basketballs) \Rightarrow round(x)$; (all members of of a category have same property)

$color(x, brown) \wedge round(x) \wedge dia(x) = 9\text{”} \wedge x \in balls \Rightarrow x \in basketballs$; (the member category can be recognized by some property)

$dogs \in domesticatedanimals$. (categories are members)

Having represented in taxonomy has benefit that, when top element is identified having some property, those its subordinates can be easily identified, as they possess similar properties as the top class, unless there is an exception.

9.3.2 Physical Decomposition of Categories

An object can be part of another, for example, nose, eyes, hands, are part of body; steering is part of car, wheels are part of wheel-assembly, and wheel-assembly is part of car. This can be represented by relations of physical decompositions as follows.

$partof(wheel, wheelassembly)$.

$partof(wheelassembly, car)$.

Thus taxonomies are *transitive relations.*

$partof(x, y) \wedge partof(y, z) \Rightarrow partof(x, z)$.

Also, the relation of reflexivity holds on individual objects, e.g., $partof(x, x)$, as x has one part, and that part is itself. The categories of composite objects is defined by structural relations between parts and assemblies.

9.3.3 Measurements

The physical objects have height, weights, mass, length, and other physical measurements. To characterize a physical object, values need to be assigned to the objects in the form of their properties. Following are the examples:

$length(ipad) = inches(5)$ or $length(ipad) = centimeters(12.5)$.

$listprice(basketball) = rupees(500)$.

$height(x) > height(y) \Rightarrow taller(x, y)$.