ĐẠI HỌC BÁCH KHOA HÀ NỘI
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

IT3160E
Introduction to Artificial Intelligence

Chapter 4 – Knowledge and inference

Part 1: Knowledge representation

Lecturer:
Muriel VISANI

Department of Information Systems
School of Information and Communication Technology - HUST

Content of the course

- Chapter 1: Introduction
- Chapter 2: Intelligent agents
- Chapter 3: Problem Solving
  - Search algorithms, adversarial search
  - Constraint Satisfaction Problems
- Chapter 4: Knowledge and Inference
  - Knowledge representation
  - Propositional and first-order logic
- Chapter 5: Uncertain knowledge and reasoning
- Chapter 6: Advanced topics
  - Machine learning
  - Computer Vision

Outline

- Chapter 4 – part 1: Knowledge representation
  - Definitions and goals
  - Different kinds of logic
  - Diversity of knowledge representations
  - Taxonomies and ontologies
    - Categories and objects
  - Building ontologies
  - Using ontologies
  - Summary
  - Homework

Goal of this Lecture

Goal	Description of the goal or output requirement	Output division/ Level (I/T/U)
M1	Understand basic concepts and techniques of AI	1.2

Knowledge representation

Definitions and goals

Knowledge representation: definition

- Knowledge representation (KR) is the study of
  - how knowledge and facts about the world can be represented, and
  - what kinds of reasoning can be done with that knowledge
- Important KR questions one has to consider:
  - Representational adequacy,
  - Representational quality,
  - Computational cost of related inferences,
  - Representation of default, commonsense, or uncertain information

Knowledge representation: goals

- We want a representation that is:
  - rich enough to express the knowledge needed to solve the problem
  - as close to the problem as possible: compact, natural and maintainable
  - able to trade off accuracy and computation time

Recall: Knowledge bases for agents

- Knowledge Base (KB) is a set of sentences in a formal language (**knowledge representation** language), telling an agent what it needs to know
  - Initially containing some background knowledge
  - Can be enriched with new sentences

- An agent can **TELL** the KB what it perceives, then **ASK** the KB which action it should take, then **TELLs** the KB which action it took

Example of KB: Cycl

- **Objective of Cycl:** providing computers with « **common sense** », everyday knowledge
  - Project led by Douglas Lenat, started in 1984. Two talks by Douglas Lenat
    - https://www.youtube.com/watch?v=2w_ekB08ohU
    - https://www.youtube.com/watch?v=3wMKoSRbGVs
  - Language based on predicates and ontologies
  - Cycl contains hundreds of thousands of **concepts**, and millions of **assertions** (facts and rules)

- **Facts:**
  - (#$capitalCity #$France #$Paris): "Paris is the capital of France. »
  - Specialization and generalization: predicates #$isa and #$genls
    - (#$isa #$BarakObama #$UnitedStatesPresident): « BarakObama belongs to the collection of U.S. presidents »
    - (#$genls #$Tree-ThePlant #$Plant): "All trees are plants".
  - Definition of a frightened person in Cycl, with variable x (the person):

    (#$and

    (#$isa ?x #$Person)

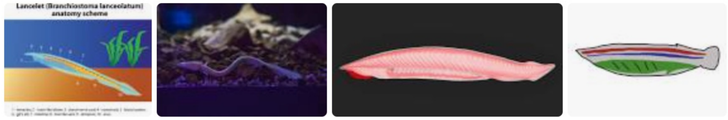
    (#$feelsEmotion ?x #$Fear #$High)

- **Rules**
  - (#$relationAllExists #$biologicalMother #$ChordataPhylum #$FemaleAnimal)
    - for every instance of the collection #$ChordataPhylum (i.e. for every chordate), there exists a female animal (instance of #$FemaleAnimal) which is its mother (described by the predicate #$biologicalMother).

ZOOLOGY

noun

an animal of the large phylum *Chordata*, comprising the vertebrates together with the sea squirts and lancelets.



Recall: Knowledge-based agents

- The KB-based agent must be able to:
  - Incorporate new percepts
  - Update internal representations of the world
  - Deduce hidden properties of the world
  - Deduce appropriate actions

Inference

Knowledge-based agents

Knowledge-based agent

Knowledge base
Inference engine

- **Knowledge base (KB):**
  - A set of sentences that describe the world and its behavior in some formal (representational) language
  - Typically **domain-specific** (*e.g.* Systematized Nomenclature of Medicine: SNOMED CT, for medical science)
  - Some KBs are more **general** (*e.g.* Cyc)
- **Inference engine**:
  - A set of procedures that:
    - use the representational language to infer new facts from known facts
    - Deduce the most appropriate actions from the KB
  - Typically, domain independent

Knowledge-based agents

Knowledge-based agent

Knowledge base
Inference engine

- **Simple example with an autonomous car agent**
  - **Extract of the Knowledge base (KB):**
    - a plastic bag is soft and cannot puncture the tires
  - **Example of a new percept**
    - I just drove over a stone, and it punctured the tires
  - **Deduced hidden property of the world that can be integrated in the KB**
    - Stones can puncture tires
  - **Outputs of the Inference engine**:
    - I can drive on plastic bags
    - I'd better avoid to drive on stones

Example of KB agent: MYCIN

- MYCIN: an expert system for diagnosis of bacterial infections
- Knowledge base represents
  - Facts about a specific patient case
  - Rules describing relations between entities in the bacterial infection domain

If	1. The stain of the organism is gram-positive, and
	2. The morphology of the organism is coccus, and
	3. The growth conformation of the organism is chains
Then	the identity of the organism is streptococcus

- Inference engine:
  - manipulates the facts and known relations to answer diagnostic queries (consistent with findings and rules)

Knowledge representation languages

- Goal: express the knowledge about the world in a computer-tractable form
- Key aspects of knowledge representation languages:
  - Syntax: describes how sentences are formed in the language
  - Semantics: describes the meaning of sentences, what is it the sentence refers to in the real world
  - Computational aspect: describes how sentences and objects are manipulated in concordance with semantical conventions
    - Programming languages: LISP; Prolog, SmallTalk, Python…
  - Example:
    - (#$capitalCity #$France #$Paris): sentence, following the syntax of Cycl
    - "Paris is the capital of France. »: semantics (meaning) of this sentence
- Many KB systems rely on some variant of **logic**

Knowledge representation

Different kinds of logic

Logic

- Many knowledge representation systems rely on some variant of logic, e.g.:
  - Propositional logic (see Chapter 4 – part 2)
  - First order logic (see Chapter 4 – part 3)
  - Temporal logic
  - … and many possible extensions of the above-mentioned
- Logic defines:
  - Syntax: describes how sentences are formed in the language
  - Semantics: describes the meaning of sentences:
    - Answers the question: what does the sentence refer to in the real world?

1- Propositional logic

- Propositional logic is the simplest type of logic
  - A **proposition** is a statement that is either true or false
  - Examples of simple sentences:
    - Hanoi is located in Vietnam
    - It is raining today
  - Examples of more complex sentences:
    - It is raining outside and the traffic in Hanoi is heavy.

It is raining outside $\wedge$ the traffic in Hanoi is heavy

1- Propositional logic

- Limitations of propositional logic
  - In propositional logic, we can only represent the facts, which are either true or false
  - The real-life is often more complicated than that!!!
  - PL is not sufficient to represent the complex sentences or natural language statements
  - The propositional logic has **very limited expressive power**
  - More about propositional logic in Chapter 4-part2

2- First order logic

- First order logic is more complex than propositional logic:
  - Objects, relations, properties are explicit
- Examples of simple sentences:
  - Red(car12)
  - Brother(Peter, John)
- Examples of more complex sentences:
  - $\forall x,y$ parent(x,y) $\Rightarrow$ child(y,x)
- More about first-order logic in Chapter 4 - part 2

Knowledge representation

Diversity of knowledge representations

Diversity of Knowledge Representations

- Many different ways of representing the same knowledge.
- Representation may make inference easier or more difficult
- Example: How to represent: "Car #12 is red"?
  - **Solution 1** (propositional logic): car12 is red
    - It's easy to ask « how is car12 »?
    - … But we don't necessarily know that « red » is one of the possible car colors
  - **Solution 2** (first-order logic): Red(car12)
    - It's easy to ask "What's red?"..
    - … But we can't ask "what is the color of car12?"
  - **Solution 3**: Color (car12, red).
      - *Turning a proposition into an object is called **reification** (thingification)*
    - *I*t's easy to ask "What's red?"
    - It's easy to ask "What is the color of car12?"…
    - …But we can't ask "What property of car12 has value red?"
  - **Solution 4**: Prop(car12, color, red).
    - It's easy to ask all these questions…
    - … But it might take more time than solution 2 to find all the red objects in the KB

Diversity of Knowledge Representations

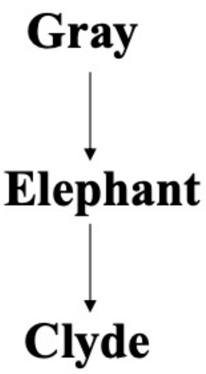
- Many different ways of representing the same knowledge
- One needs to choose the type of Knowledge Representation that best fits its needs
    - In terms of inference

Object-Property-Value Representation

- Object-property-value representation
  - Prop(Object, Property, Value)
- If we merge many properties of the same object we get the frame-based (object-centered) representation:
  - Prop(Object, Property1, Value1)
  - Prop(Object, Property2, Value2)
  - ...
  - Prop(Object, Property-n, Value-n)

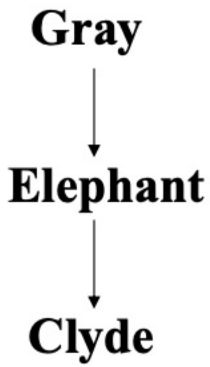
Property Inheritance

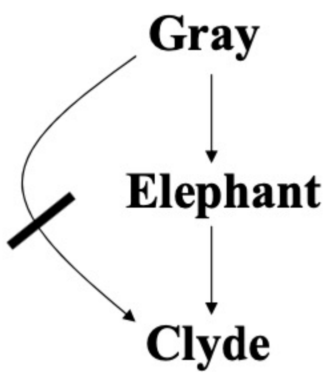
- **Properties** can be **inherited** from more general concepts (notion of category)
- Example:
  - Clyde is an Elephant & Elephant is Gray



Property Inheritance and exceptions

- **Properties** can be **inherited** from more general concepts
  - But, there might be exceptions!!!
- Example: Albino elephant
  - Clyde is an Elephant & Elephant is Gray & Clyde is not Gray





- Uncertainty will be studied in Chapter 5 of this course
  - For this lecture on knowledge representation, we ignore exceptions

Knowledge representation

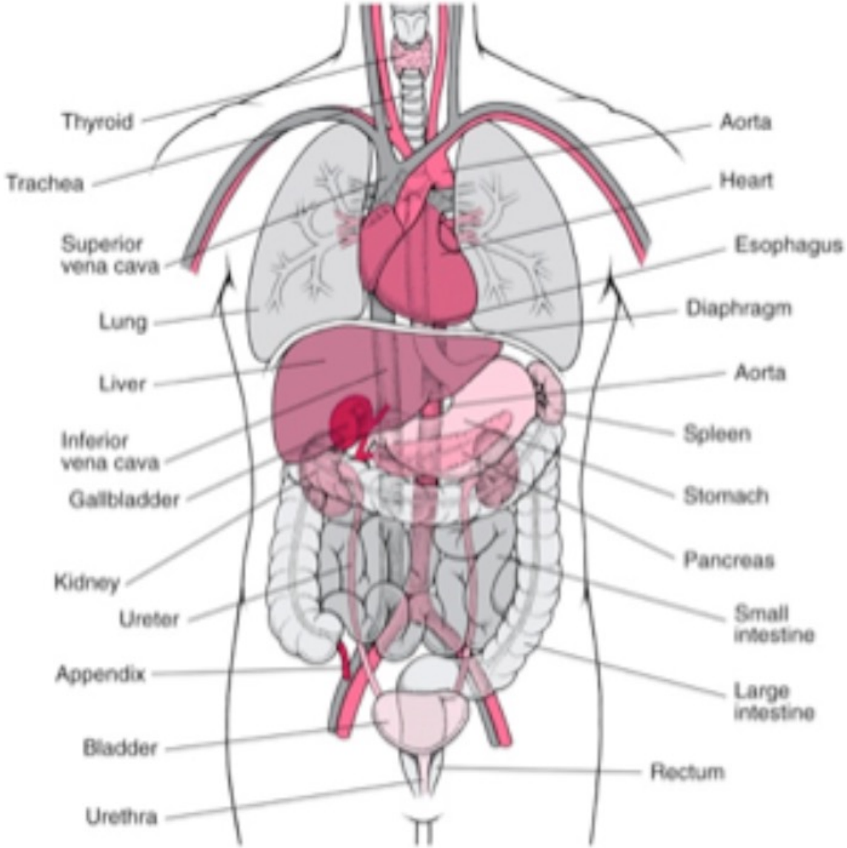
Taxonomies and ontologies

Conceptualization and ontologies

- 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 domain
- A conceptualization specifies:
  - What types of **objects** are being modeled
  - The **vocabulary** for specifying **objects**, **relations** and **properties**
  - The meaning (*a.k.a* **intention**) of the relations or properties
- An ontology is a **specification** of a conceptualization
  - Ontologies are not specific to any technology…
  - … But, many technologies are built upon ontology concepts

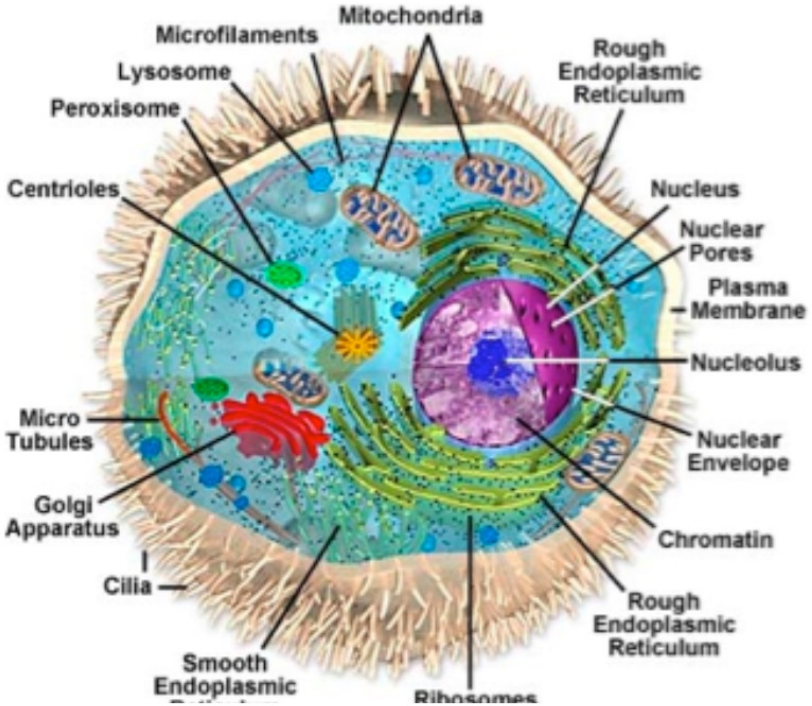
What is an ontology?

- An ontology is a model of some aspect of the world
  - Introduces **vocabulary** relevant to the domain, *e.g.*
    - Anatomy



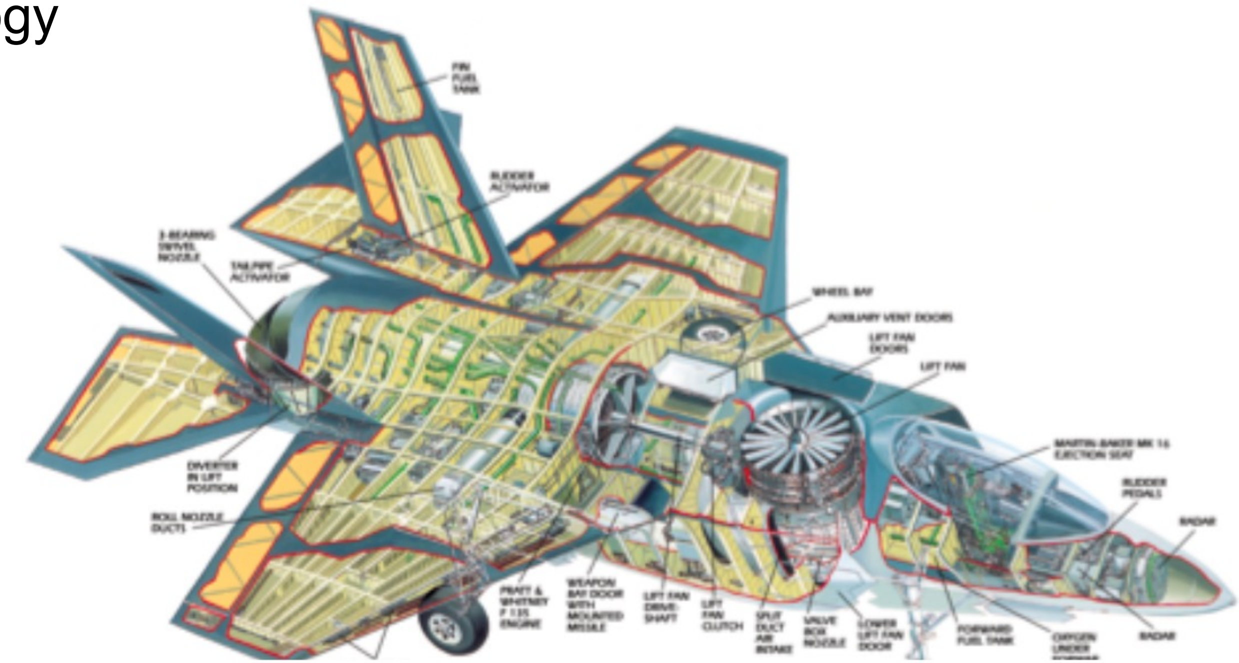
What is an ontology?

- An ontology is a model of some aspect of the world
  - Introduces **vocabulary** relevant to the domain, *e.g.*
    - Anatomy
    - Cellular biology



What is an ontology?

- An ontology is a model of some aspect of the world
  - Introduces **vocabulary** relevant to the domain, *e.g.*
    - Anatomy
    - Cellular biology
    - Aerospace



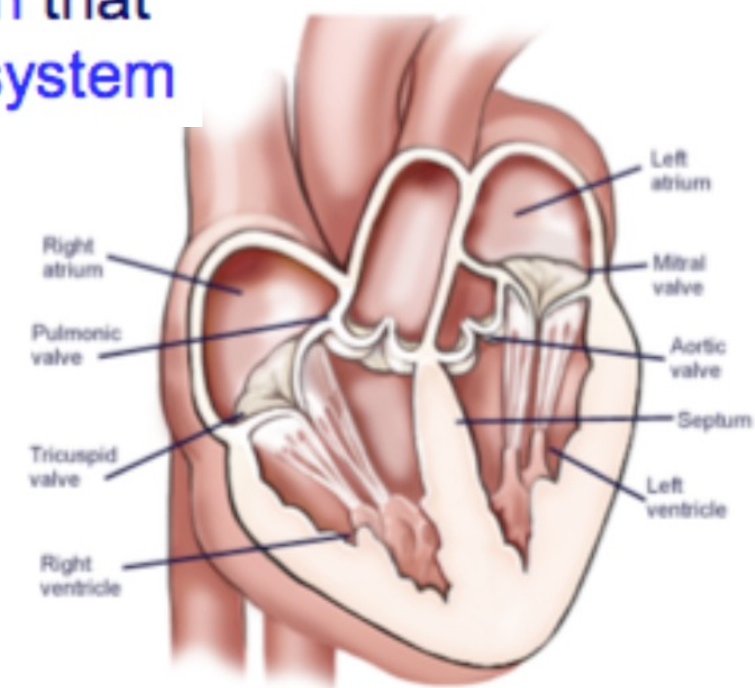
What is an ontology?

- An ontology is a model of some aspect of the world
  - Introduces **vocabulary** relevant to the domain, *e.g.*
    - Anatomy
    - Cellular biology
    - Aerospace
    - …
    - Hotdogs
    - …

What is an ontology?

- An ontology is a model of some aspect of the world
  - Introduces **vocabulary** relevant to the domain
  - Specifies the **meaning** (intention, semantics) of terms

Heart is a muscular organ that is part of the circulatory system



What is an ontology?

- An ontology is a model of some aspect of the world
  - Introduces **vocabulary** relevant to the domain
  - Specifies the **meaning** (intention, semantics) of terms…

    Heart is a muscular organ that
    is part of the circulatory system

  - … and **formalizes** it using suitable logic

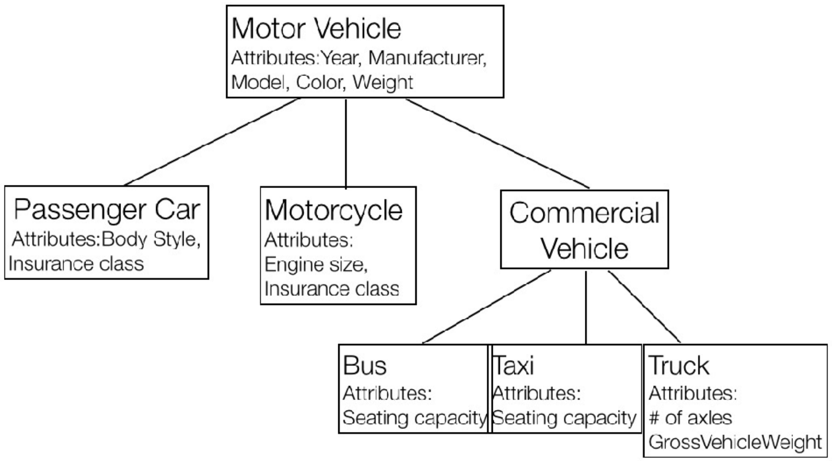
$$\forall x.[\mathsf{Heart}(x) \rightarrow \mathsf{MuscularOrgan}(x) \wedge \exists y.[\mathsf{isPartOf}(x,y) \wedge \mathsf{CirculatorySystem}(y)]]$$

    - Heart, MuscularOrgans, CirculatorySystem = **objects** linked by a **hierarchy**
    - IsPartOf = **relation** between objects
    - Heart thus has all **properties** of a muscular organ (inheritance)

Ontologies *vs.* taxonomies

- Ontologies are often confused with taxonomies

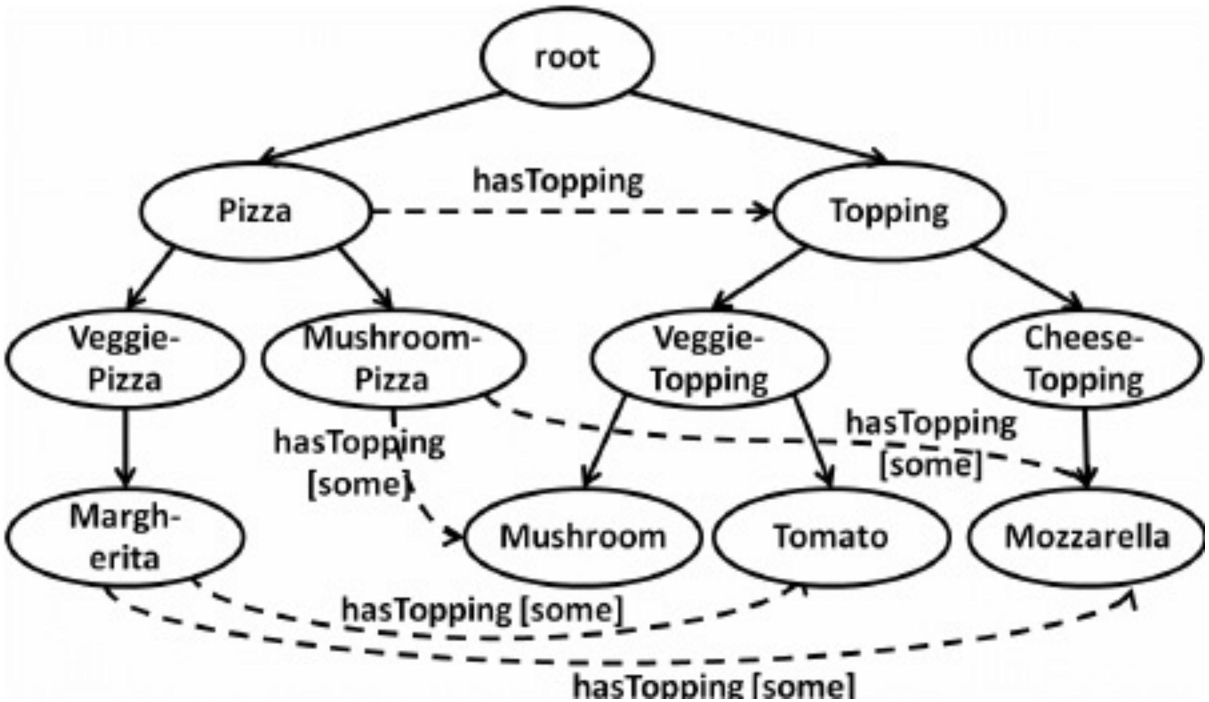
- Taxonomy
  - Represents hierarchical relationships within a category
  - Categorizes entities within only one dimension



- Ontology
  - More sophisticated and informative:
    - Include information about the relationship among entities
    - The relationships depend on the context
  - Can be seen as a connection of context-dependent taxonomies
    - *E.g.* the motor vehicle taxonomy can be connected with "client needs" into a larger structure of an ontology

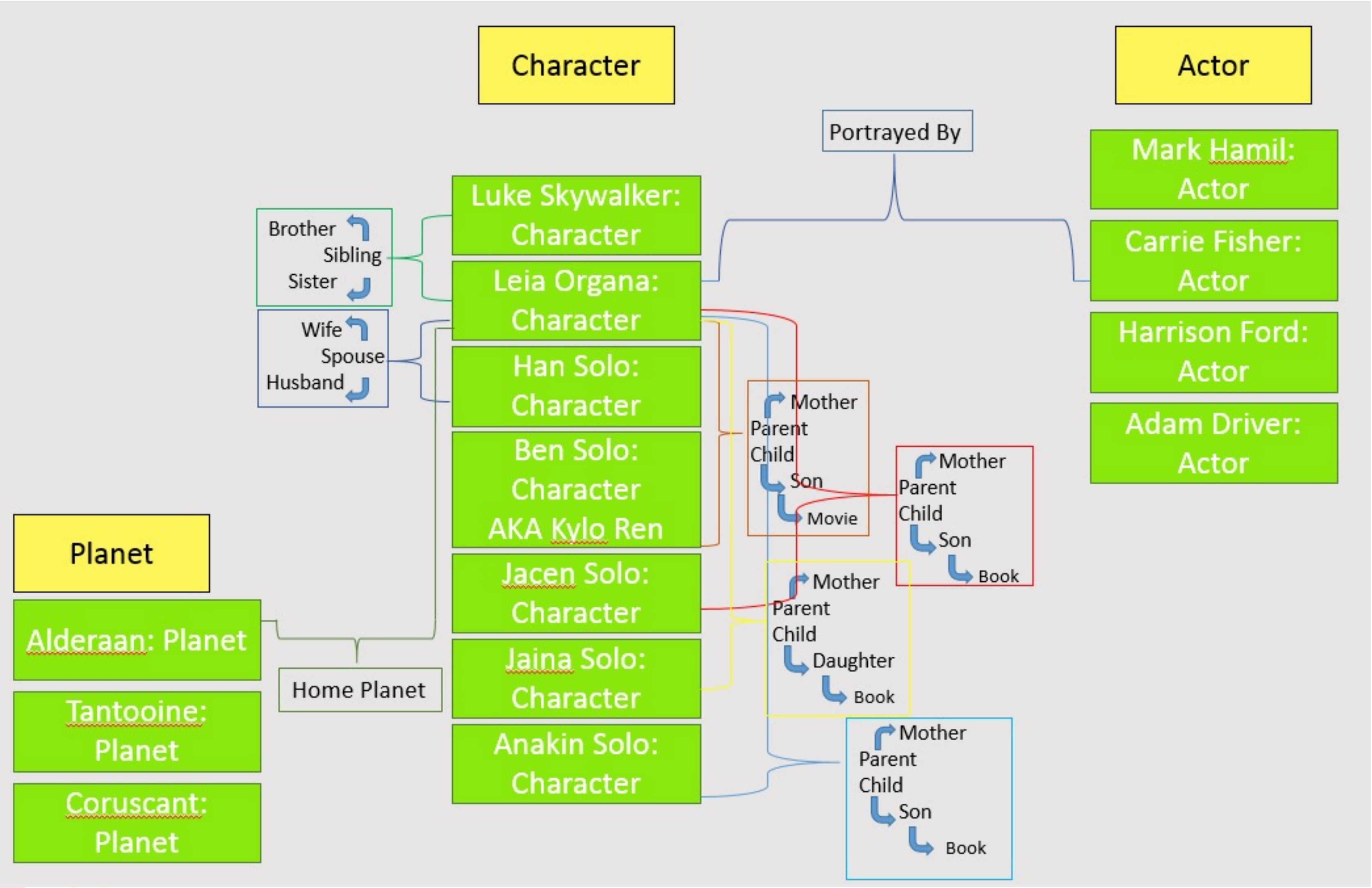
- More information on: https://www.dataversity.net/smart-data-webinar-organizing-data-knowledge-role-taxonomies-ontologies/

Simple example of ontology #1



From: Kim *et al.*. (2012). Efficient Regression Testing of Ontology-Driven Systems. Proceedings of the 2012 International Symposium on Software Testing and Analysis

Simple example of ontology #2



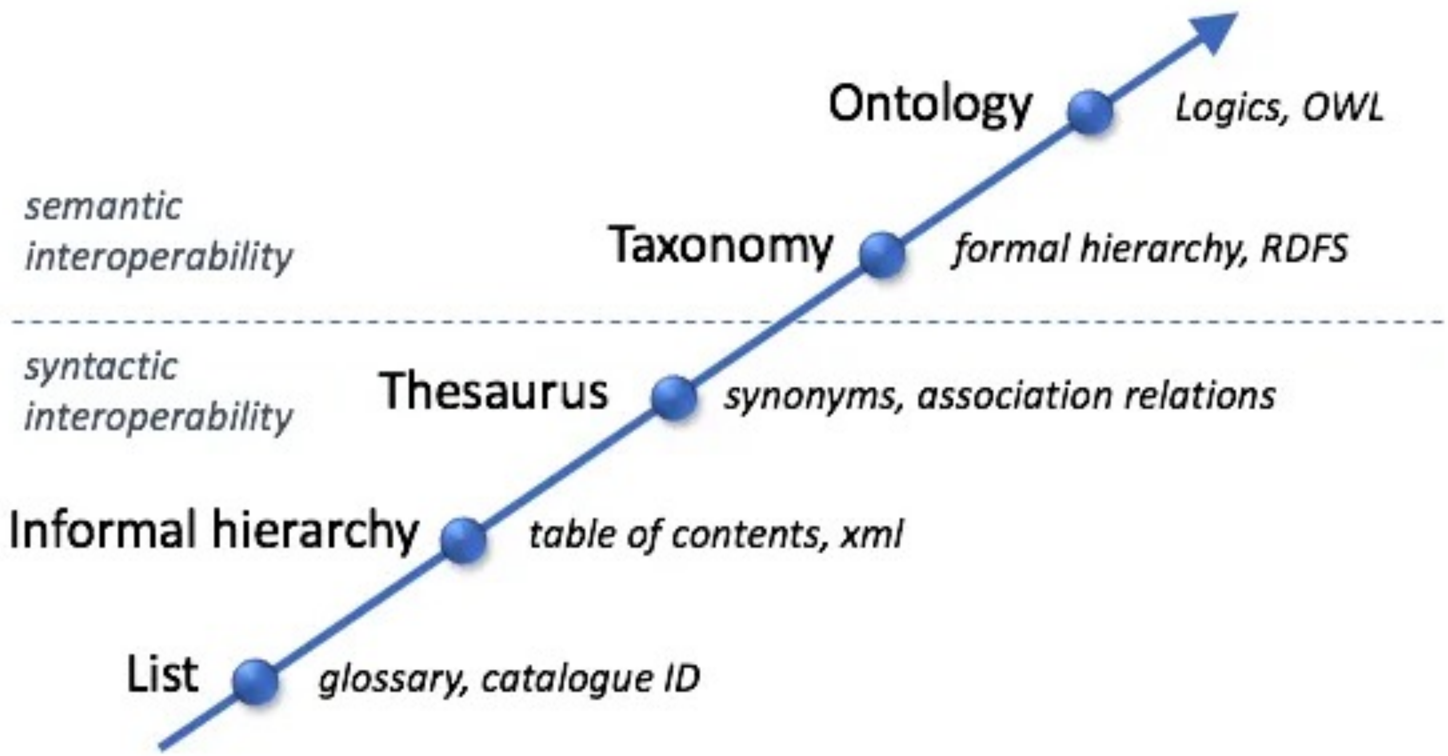
From: https://www.earley.com/

Ontologies *vs.* taxonomies

- Taxonomies
  - Useful for organizing information
    - Both internally and externally
  - Example of standard: RDF
    - https://en.wikipedia.org/wiki/Resource_Description_Framework

- Ontologies
  - Based on taxonomies
  - Useful to harmonize Knowledge Bases
    - Often employed in standards
    - Can be used for logic inference
    - Can be used to enhance machine learning algorithms
  - Example of standard: OWL
    - https://www.scss.tcd.ie/Owen.Conlan/CS7063/06%20Introduction%20to%20OWL%20(1%20Lecture).ppt.pdf

More generally…



From: https://emmc.info/emmo-info/

Knowledge representation

Ontologies: categories and objects

Object categories and sub-categories

- Subclass relations organize categories into a **taxonomy**
- Definitions and examples, using first-order logic:

- An object is a member of a category.
  $BB_9 \in Basketballs$
- A category is a subclass of another category.
  $Basketballs \subset Balls$
- All members of a category have some properties.
  $(x \in Basketballs) \Rightarrow Spherical(x)$
- Members of a category can be recognized by some properties.
  $Orange(x) \land Round(x) \land Diameter(x) = 9.5'' \land x \in Balls \Rightarrow x \in Basketballs$
- A category as a whole has some properties.
  $Dogs \in DomesticatedSpecies$

Object categories and sub-categories

- 2+ categories are **disjoint** if they have no members in common
    - *E.g.* males and females
- An **exhaustive decomposition** covers all possible sub-categories
- A **partition** is a disjoint exhaustive decomposition

$$
\begin{aligned}
&Disjoint(\{Animals, Vegetables\}) \\
&ExhaustiveDecomposition(\{Americans, Canadians, Mexicans\}, \\
&\qquad\qquad\qquad NorthAmericans) \\
&Partition(\{Males, Females\}, Animals)\ .
\end{aligned}
$$

Category definition and relations between categories

- Categories can be **defined** by providing necessary and sufficient conditions for membership, *e.g.*

$$x \in Bachelors \Leftrightarrow Unmarried(x) \land x \in Adults \land x \in Males .$$

- Examples of **relations** between categories / objects
  - PartOf (to express hierarchies)
  - AttachedTo (to define composite objects using structural relations among their parts)
  - BunchOf (to define composite objects with definite parts but no particular structure)

Object / category properties

- Objects /catgeories have **properties**
  - Intrinsic or extrinsic
    - *E.g.* mass is an **intrinsic property** of any physical object, whereas weight is an **extrinsic property** (depending on the strength of the gravitational field)
- Property values for an object are called **measures**
- Measures are expressed using a **unit function**
  - *E.g.*, for a line segment L_1

$$Length(L_1) = Inches(1.5) = Centimeters(3.81)$$

Knowledge representation

Building ontologies

Building ontologies

- Knowledge **elicitation** plays a prominent role
  - Knowledge is often locked away in the heads of domain experts
  - The experts may not be aware of the implicit conceptual models that they use
  - We have to draw out and make explicit all the known & “unknown knowns”
- Uncovering the "obvious” is important
  - it must be expressed explicitly for the machine (if needed for inferencing)
    - For instance, using Cycl
- Huge differences across domains
  - Explicit models may or may not exist

Building ontologies

- Essential steps in modeling for building ontologies
  1. Establish the purpose
  2. **Informal / semiformal knowledge elicitation**
     (collect terms, organize them, paraphrase and clarify, diagram informally)
  3. Refine requirements and tests
  4. Implementation
     (Paraphrase and comment at each stage before implementing, Scale up a bit, check performance, populate)
  5. Evaluate
     (Against goals, include tests for evolution and change management)
  6. Monitor use

Knowledge representation

Using ontologies

Why using ontologies?

- They enable automated reasoning about data
- They provide coherent and interpretable representation of the data (for humans & for machine)
  - For any kind of data: unstructured, semi-structured or structured
    - Very useful for Natural Language Processing for instance
- They are easy to extend to new relationships / concepts
  - Without impacting dependent processes / systems

Where to use ontologies?

- Ontologies can be used in any domain
- They are especially useful in the **semantic web**
  - Semantic Web: extension of the WWW through standards set by the World Wide Web Consortium (W3C)
  - Its goal is to make the semantics (meaning) of internet data machine-readable
  - Resource Description Framework (RDF) is a standard set by W3C to describe web resources
  - RDF data is **queried** using **SPARQL**

Example of a RDF document

Title	Artist	Country	Company	Price	Year
Empire Burlesque	Bob Dylan	USA	Columbia	10.90	1985
Hide your heart	Bonnie Tyler	UK	CBS Records	9.90	1988

RDF is useful for describing **taxonomies**

RDF (XML) classes correspond to categories and objects

Example from www.w3schools.com

```
<?xml version="1.0"?>

<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:cd="http://www.recshop.fake/cd#">

<rdf:Description
rdf:about="http://www.recshop.fake/cd/Empire Burlesque">
  <cd:artist>Bob Dylan</cd:artist>
  <cd:country>USA</cd:country>
  <cd:company>Columbia</cd:company>
  <cd:price>10.90</cd:price>
  <cd:year>1985</cd:year>
</rdf:Description>

<rdf:Description
rdf:about="http://www.recshop.fake/cd/Hide your heart">
  <cd:artist>Bonnie Tyler</cd:artist>
  <cd:country>UK</cd:country>
  <cd:company>CBS Records</cd:company>
  <cd:price>9.90</cd:price>
  <cd:year>1988</cd:year>
</rdf:Description>
.
.
.
</rdf:RDF>
```

Web Ontology Language (OWL)

- **OWL is** a vocabulary built with **RDF** that provide new terms for creating more detailed descriptions of resources
  - OWL adds semantics to the schema
    - Allows to specify way more about the classes and properties
    - For example, it can indicate that
      - If "A isMarriedTo B" then this implies "B isMarriedTo A"
      - if "*C isAncestorOf D* " and *D isAncestorOf E* " then "*C isAncestorOf E* " (inference)

Web Ontology Language (OWL)

- There are 3 versions of OWL
  - OWL Full
    - an extension of RDF
    - allows for classes as instances, modification of RDF and OWL vocabularies
  - OWL DL
    - Part of OWL Full that fits in the Description Logic framework
      - Description logics (DL) are a family of formal knowledge representation languages.
      - Many DLs are more expressive than propositional logic but less expressive than first-order logic.
    - Known to have decidable reasoning
      - In logic, a true/false decision problem is decidable if there exists an effective (inference) method for finding the correct answer
  - OWL Lite
    - a subset of OWL DL
    - Easier to use for reasoning

Examples

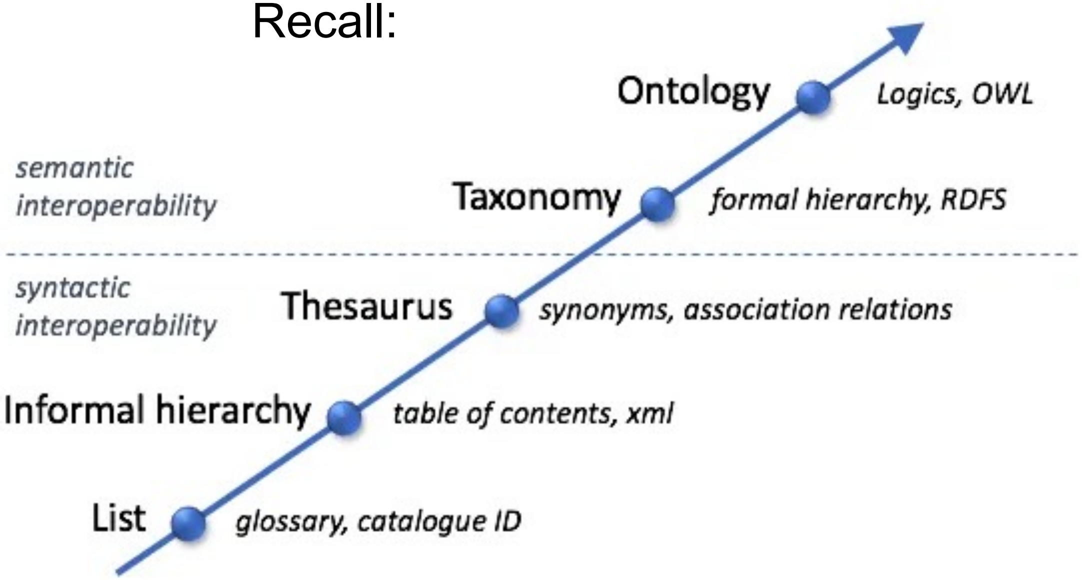
- **Transport:** https://www.ibm.com/support/knowledgecenter/SSWLGF_8.5.0/com.ibm.sr.doc/cwsr_configrn_classifications06.html
- **WordNet** is a large extension of dictionnay and **thesaurus** of English
  - Nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept
  - The WordNet schema

    has 3 main classes:

    Synset, WordSense

    and Word

Recall:



Wordnet class hierarchy

Property	Domain	Range
containsWordSense	Synset	WordSense
word	WordSense	Word
lexicalForm	Word	xsd:string
synsetId	Synset	xsd:string
tagCount	Synset	xsd:integer
gloss	Synset	xsd:string
frame	VerbWordSense	xsd:string
hyponymOf	Synset	Synset
entails	Synset	Synset
similarTo	Synset	Synset
memberMeronymOf	Synset	Synset
substanceMeronymOf	Synset	Synset
partMeronymOf	Synset	Synset
classifiedByTopic	Synset	Synset
classifiedByUsage	Synset	Synset
classifiedByRegion	Synset	Synset
causes	Synset	Synset
sameVerbGroupAs	Synset	Synset
attribute	Synset	Synset
adjectivePertainsTo	Synset	Synset
adverbPertainsTo	Synset	Synset

Property	Domain	Range
derivationallyRelated	WordSense	WordSense
antonymOf	WordSense	WordSense
seeAlso	WordSense	WordSense
participleOf	WordSense	WordSense
classifiedBy	Synset	Synset
meronymOf	Synset	Synset

For WordNet visualization:
http://wordvis.com

WordNet can be expressed using RDF and OWL:

https://www.w3.org/TR/wordnet-rdf/

Where to use ontologies? (cont'd)

- For instance, ontologies are very useful for intelligent virtual assistants and bots
  - Form the basis for inference engines: mechanisms to answer questions that have not been pre-programmed into the bot
  - Bots powered by ontologies are more effective and seem more natural because they have access to more sophisticated information
    - Can effectively detect user's intent by detecting keywords / key phrases along with synonyms and variations

Conclusion: Information Architecture matters…

- “There is no AI without IA” *Seth Earley, CEO of EIS*
- Taxonomies are often the first building block in an IA
- Then, ontologies can be conceived in stages
    - Not all at once
    - But, keeping in mind the future extensions, for each stage

Knowledge representation

Summary

Summary

- Knowledge representation is at the basis of any KB-agent, and any logic
- In the two next parts of this chapter, we will focus on Propositional Logic and First-Order Logic
- But, in this lecture, you've learned that there are many possible knowledge representations…
  - …And that the choice of the knowledge representation matters!
- In this lecture, you've also had an overview of taxonomies and ontologies
  - We will not really rely on them in the rest of this course, but…
  - … these will be very useful for the course on Semantic Web…
  - … and they are powerful Knowledge Representations, important in AI!
    - And possibly important for your future projects!

Knowledge representation

Homework

Homework

- Make the following tutorial:
  - http://www.linkeddatatools.com/semantic-web-basics
    - The Basics
    - 1: Introducing Graph Data
    - 2: Introducing RDF
    - 3: Semantic Modeling
    - 4: Introducing RDFS & OWL
    - 5: Querying Semantic Data

Chapter 4 – part 1

Questions

VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG
SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY

Thank you for your attention!