Knowledge Representation

Pınar Yolum
Email: p.yolum@uu.nl

Department of Information and Computing Sciences
Utrecht University

Knowledge representation

- Information about the world so that an agent can use to do its tasks
- What content do we want to put into an agent's knowledge base?
- What are the different properties of the content?
  - Example: Physical Objects
  - Example: Events
- What are the languages for representation?
  - First order logic
  - Description logics
  - Semantic networks
  - Frames

Ontology

- A specification of a conceptualization or a set of knowledge terms for a particular domain, including
  - The vocabulary
  - The semantic interconnections
  - Some rules of inference
- An agent can use the ontology to reason about its environment
- As new information becomes available, the agent can add them to the knowledge base and make new inferences
- The ontology can be individual or common

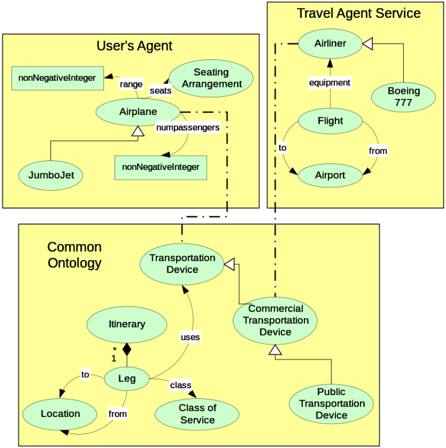
Ontology components

- *Concepts* organized as a hierarchy
- Inheritance (isA) relation
  - Build a taxonomy
  - Example: A human is a mammal
- Part-whole (isPartOf) relation
  - Build a mereology
  - Example: A steering wheel is part of a car
- *Concept properties* with values (e.g., age of a human)
- *Concept relations* among concepts (e.g., a human uses a steering wheel)
- *Concept instances* that correspond to individuals (e.g., Pınar is an instance of a human)

Common ontologies

- A shared representation is essential to successful communication and coordination
  - For humans: physical, biological, and social world
  - For computational agents: common ontology (terms used in communication)
- Example effort: Dbpedia (https://wiki.dbpedia.org/services-resources/ontology)

Example common ontology



Example from: Service Oriented Computing: Semantics, Processes, Agents. Singh and Huhns, 2005

Semantic Web

Can we do these on the Web?

- Meaningful queries to search engines: Find me movies that were shot in Utrecht but are not about Utrecht.
- Meaningful filtering of recommendation: Get a recommendation of a restaurant where the food is served on time.
- Meaningful composition of reviews: Buy a reliable tablet PC that can run Ubuntu.

Semantic Web

- Giving meaning to the Web
- Agents can understand and process the Web on their own
- Make inferences (rather than smart queries as in DB)
- E-commerce: Offer services with descriptive content
- Search: Describe word or documents to allow meaningful search (not just keyword)
- BBC Ontologies: https://www.bbc.co.uk/ontologies

Description Logics (DL)

- **Decidable** fragment of first-order logic
- Contains only unary and binary predicates
- Different syntax than first-order logic and does not use variables
- Family of logics that differ in their expressivity

Operators

Represent all concepts as sets

- Intersection: $X \sqcap Y$ (set of things that are both in X and Y)
  - father $\equiv$ parent $\sqcap$ malePerson
  - FOL?
- Union: person $\equiv$ malePerson $\sqcup$ femalePerson (set of things that are in either X or Y)
- Complementary: $\neg X$ (set of things that are not in X)

Restrictions

A constraint set on a relationship value:

- Universal restriction: $\forall R.C$, where all the relation values should be in C.
  - $hasOnlySons \equiv \forall$hasChild.malePerson
  - FOL?
- Existential restriction: $\exists R.C$, where at least one relation value is in C.
  - $hasASon \equiv \exists$hasChild.malePerson
  - FOL?
  - How many sons? How many daughters?

Cardinality Restrictions

- Maximum cardinality: $\geq nR.C$
  - The class of things that have at least n relations to C
  - person $\sqcap \geq 5 hasChild.happyPerson$
- Minimum cardinality: $\leq nR.C$

Other Constructs

- Enumeration: List all the elements of the class
  - myFriends $\equiv \{$mary, susan, pinar$\}$
  - Possible to use it without a name
  - $\exists$hasFriend.$\{$mary, susan, pinar$\}$
- Top: Class of everything ($\top$)
  - Everybody has only sons
  - $\top \sqsubseteq \forall$ hasChild.malePerson
- Bottom: The empty class ($\bot$)
  - Nobody has only sons.
  - $\forall$hasChild.malePerson $\sqsubseteq \bot$

Concept inclusion and equivalence

- Concept Inclusion: Every X is a Y; thus X is a subset of Y.
  - $X \sqsubseteq Y$: student $\sqsubseteq$ person
  - Y can be a class itself or a class defined through other classes
  - student $\sqsubseteq$ person $\sqcap$ young
  - student $\sqsubseteq$ person $\sqcap$ $\forall$hasFriend.young
- Equivalence: Every X is a Y; every Y is a X.
  - $X \equiv Y$: student $\equiv$ person
  - Every student is a person and every person is a student

Assertions

Talk about individuals that are in classes or relations

- Role assertions: R(x,y)
  - *R* relates *x* and *y*
  - hasFriend(mary, susan)
- Class assertions: C(x)
  - *x* belongs to class *C*
  - student(mary)

Exercise

Given person, hasChild, hasFriend, malePerson, happyPerson, femalePerson, define the following:

- parents have at least one child
- parents with exactly five children
- married men are happy people
- no parent is happy
- everybody has only happy friends
- people who have the same friends are happy
- people with an unhappy child are not happy

Reasoning: Consistency

Given a knowledge base, is there a contradiction between axioms?

- femalePerson $\sqsubseteq$ happyPerson
- malePerson $\sqsubseteq \neg$ happyPerson
- malePerson(john)
- happyPerson(john)

It is allowed to say this but it will create an inconsistency. How can an agent deal with this?

Reasoning: Inference

Given a knowledge base, is one class a subclass of another one?

- Query: $\exists hasChild.\{mary\} \sqsubseteq$ happyPerson
- Are all parents of mary included in happyPerson?
- Check for all parents

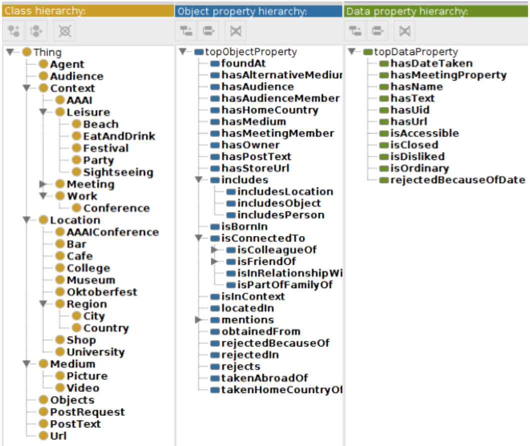
Reasoning: Inference

- Given a knowledge base, is one individual instance of a class?
  happyPerson(mary)?
- Given a knowledge base, list all individuals with the given property
  malePerson $\sqcap \exists hasChild.\{mary\} \sqsubseteq$ happyPerson
  Find ...

Standards and Tools

- Web Ontology Language (OWL): W3C Standard, Description Logic equivalent
- Ontology editors: Protégé
- Ontology APIs: OWL API
- Query language: OWL DL
- Ontology Reasoners: Pellet, Fact++, Hermit

Example Ontology



- *Concepts* represent a class of individuals (e.g., wig `:wig` is an instance of *Object*).
- Object properties relate different individuals with a specific relation (e.g., includesObject relates a `:Medium` to a `:wig`).
- Data properties relate data values to individuals (e.g., isOrdinary relates `:wig` to either *true* or *false*).

Various Syntax

OWL Constructor	DL Syntax	Manchester	Example
intersectionOf	$C \sqcap D$	C AND D	Fruit AND Wine
unionOf	$C \sqcup D$	C OR D	Wine OR Fruit
complementOf	$\neg C$	NOT C	NOT Wine
oneOf	$\{a\} \sqcup \{b\}$	$\{a\ b\ ..\}$	$\{White\ Red\ Rose\}$
someValuesFrom	$\exists R.C$	R SOME C	hasColor SOME WineColor
allValuesFrom	$\forall R.C$	R ONLY C	hasColor ONLY $\{Red, Rose\}$
minCardinality	$\leq NR$	R MIN N	hasColor MIN 3
maxCardinality	$\geq NR$	R MAX N	hasColor MAX 3
cardinality	$= NR$	R EXACTLY N	hasColor EXACTLY 1
hasValue	$\exists R\{a\}$	R VALUE a	hasColor VALUE White