INF5390 – Kunstig intelligens

Knowledge Representation

Roar Fjellheim

Outline

- General ontology
- Categories and objects
- Events and processes
- Category reasoning
- Default logics
- Internet shopping world
- Summary

Chapter 12: Knowledge Representation

Ontologies

- An *ontology* is a “vocabulary” and a “theory” of a certain “part of reality”
- *Special-purpose* ontologies apply to restricted domains (e.g. electronic circuits)
- *General-purpose* ontologies have wider applicability across domains, i.e.
  - Must include concepts that cover many subdomains
  - Cannot use special “short-cuts” (such as ignoring time)
  - Must allow unification of different types of knowledge
- GP ontologies are useful in widening applicability of reasoning systems, e.g. by including time

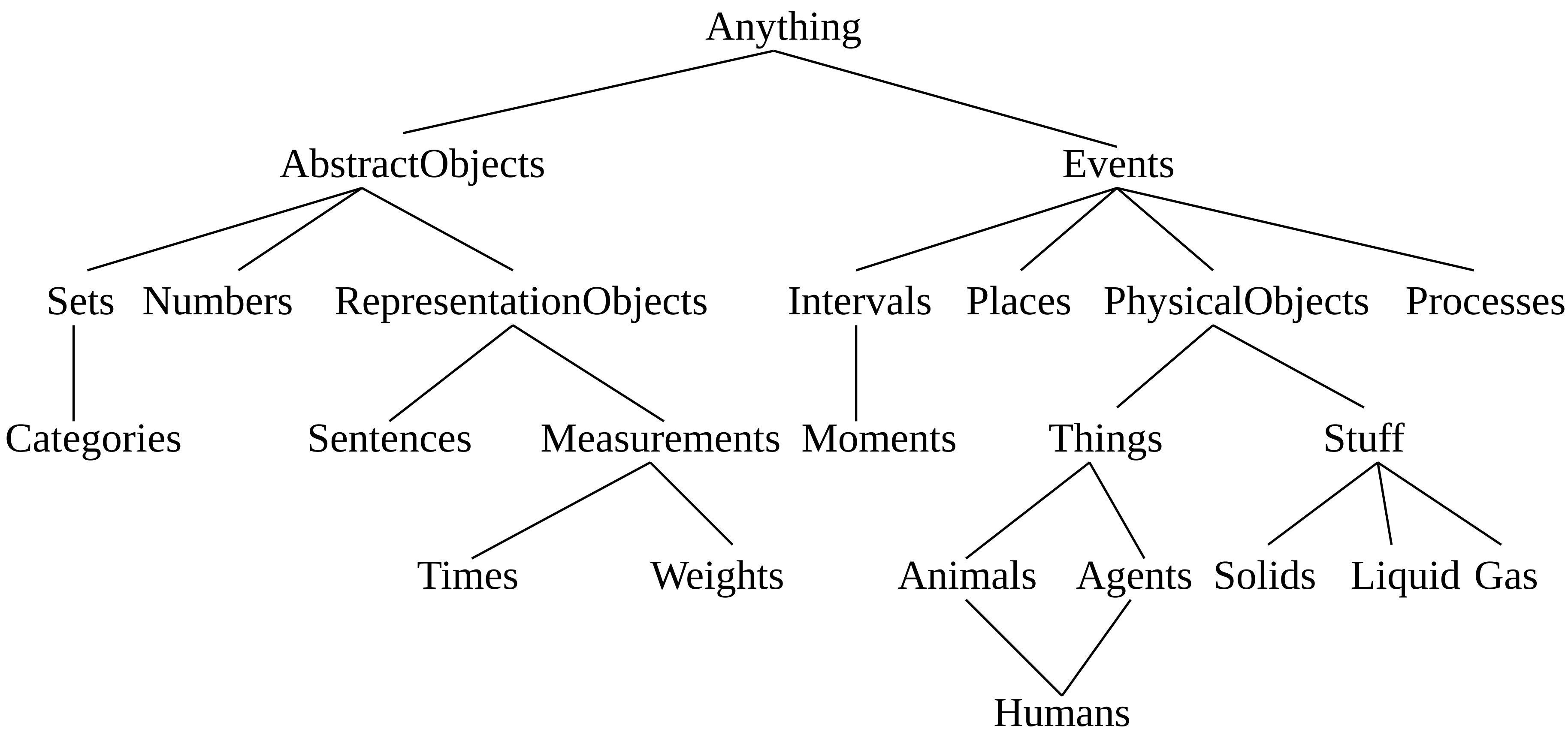
Ontological engineering

- Representing a general-purpose ontology is a difficult task called *ontology engineering*
- Existing GP ontologies have been created in different ways:
  - By team of trained ontologists
  - By importing concepts from database(s)
  - By extracting information from text documents
  - By inviting anybody to enter commonsense knowledge
- Ontological engineering has only been partially successful, and few large AI systems are based on GP ontologies (use special purpose ontologies)

Elements of a general ontology

- Categories of objects
- Measures of quantities
- Composite objects
- Time, space, and change
- Events and processes
- Physical objects
- Substances
- Mental objects and beliefs

Top-level ontology of the world



Categories and objects

- Categories are used to classify objects according to common properties or definitions

  $$\forall x x \in Tomates \Rightarrow Red(x) \wedge Round(x)$$

- Categories can be represented by
  - Predicates: *Tomato*(*x*)
  - Sets: The constant *Tomatoes* represents set of tomatoes (reification)
- Roles of category representations
  - Instance relations (is-a): $x_1 \in Tomatoes$
  - Taxonomical hierarchies (*Subset*): $Tomatoes \subset Fruit$
  - Inheritance of properties
  - (Exhaustive) decompositions

Objects and substance

- Need to distinguish between substance and discrete objects
- Substance (“stuff”)
  - Mass nouns - not countable
  - Intrinsic properties
  - Part of a substance is (still) the same substance
- Discrete objects (“things”)
  - Count nouns - countable
  - Extrinsic properties
  - Parts are (generally) not of same category

Composite objects

- A *composite object* is an object that has other objects as parts
- The *PartOf* relation defines the object containment, and is transitive and reflexive

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

$$PartOf(x, x)$$

- Objects can be grouped in *PartOf* hierarchies, similar to *Subset* hierarchies
- The *structure* of the composite object describes how the parts are related

Measurements

- Need to be able to represent properties like height, mass, cost, etc. Values for such properties are *measures*
- *Unit functions* represent and convert measures

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

$$\forall l \; Centimeters(2.54 \times l) = Inches(l)$$

- Measures can be used to describe objects

$$Mass(Tomato_1) = Kilograms(0.16)$$

$$\forall d \; d \in Days \Rightarrow Duration(d) = Hours(24)$$

- Non-numerical measures can also be represen-ted, but normally there is an *order* (e.g. >). Used in *qualitative physics*

Event calculus

- *Event calculus*: How to deal with change based on representing points of time
- Reifies fluents (Norwegian: “forløp”) and events
  - √ A fluent: *At(Shankar, Berkeley)*
  - √ The fluent is true at time t: *T(At(Shankar, Berkeley),t)*
- Events are instances of event categories

$$E_1 \in Flyings \wedge Flyer(E_1, Shankar) \wedge Origin(E_1, SF) \wedge Destination(E_1, LA)$$

- Event E_1 took place over interval i
  - √ $Happens(E_1, i)$
- Time intervals represented by (start, end) pairs
  - √ $i = (t_1, t_2)$

Event calculus predicates

- *T(f, t)* Fluent *f* is true at time *t*
- *Happens(e, i)* Event *e* happens over interval *i*
- *Initiates(e, f, t)* Event *e* causes fluent *f* to start at *t*
- *Terminates(e, f, t)* Event *e* causes *f* to cease at *t*
- *Clipped(f, i)* Fluent *f* ceases to be true in int. *i*
- *Restored(f, i)* Fluent *f* becomes true in interval *i*

Using event calculus

- Can extend predicate to cover intervals

  $T(f,(t_1,t_2)) \Leftrightarrow [\forall t\ (t_1 \leq t < t) \Longrightarrow T(f,t)]$

- Fluents and actions are defined with domain-specific axioms, e.g. in the wumpus world

  $Initiates(e, HaveArrow(a), t) \Leftrightarrow e = Start$
  $Terminates(e, HaveArrow(a), t) \Leftrightarrow e \in Shootings(a)$

- Can extend event calculus to represent simultaneous events, continuous events, etc.

Time intervals

- Time intervals are partitioned into *moments* (zero duration) and *extended intervals*

$$Partition(\{Moments, ExtendedIntervals\}, Intervals)$$

$$\forall i \; i \in Intervals \Rightarrow (i \in Moments \Leftrightarrow Duration(i) = 0)$$

- Functions *Start* and *End* delimit intervals

$$\forall i \; Interval(i) \Rightarrow Duration(i) = (Time(End(i)) - Time(Start(i)))$$

- May use e.g. January 1, 1900 as arbitrary time 0

$Time(Start(AD1900))=Seconds(0)$

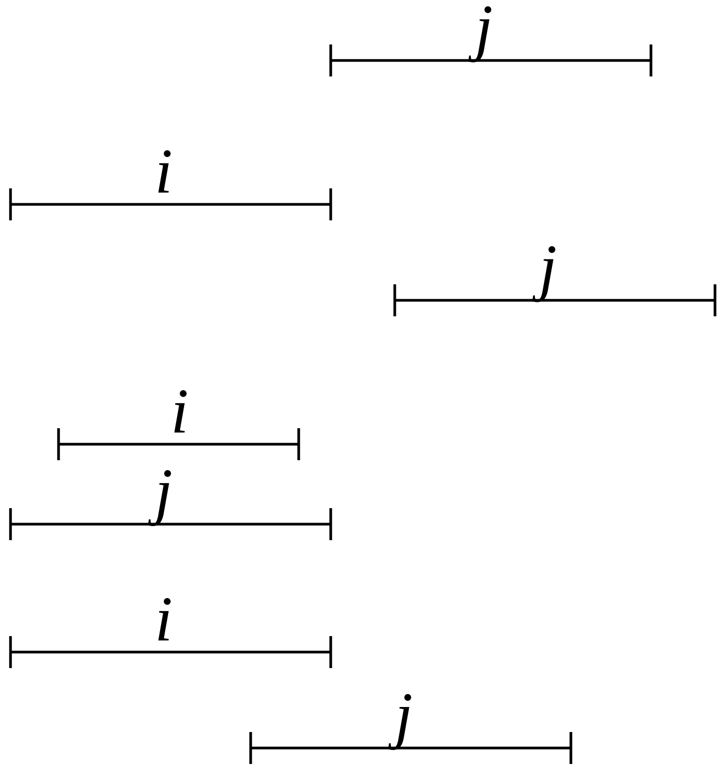
Relations between time intervals

Meet(*i*, *j*)

Before(*i*, *j*)
After(*j*, *i*)

During(*i*, *j*)

Overlap(*i*, *j*)
Overlap(*j*, *i*)



Can be expressed logically, e.g.

$$\forall i, j\ Meet(i, j) \Leftrightarrow Time(End(i)) = Time(Start(j))$$

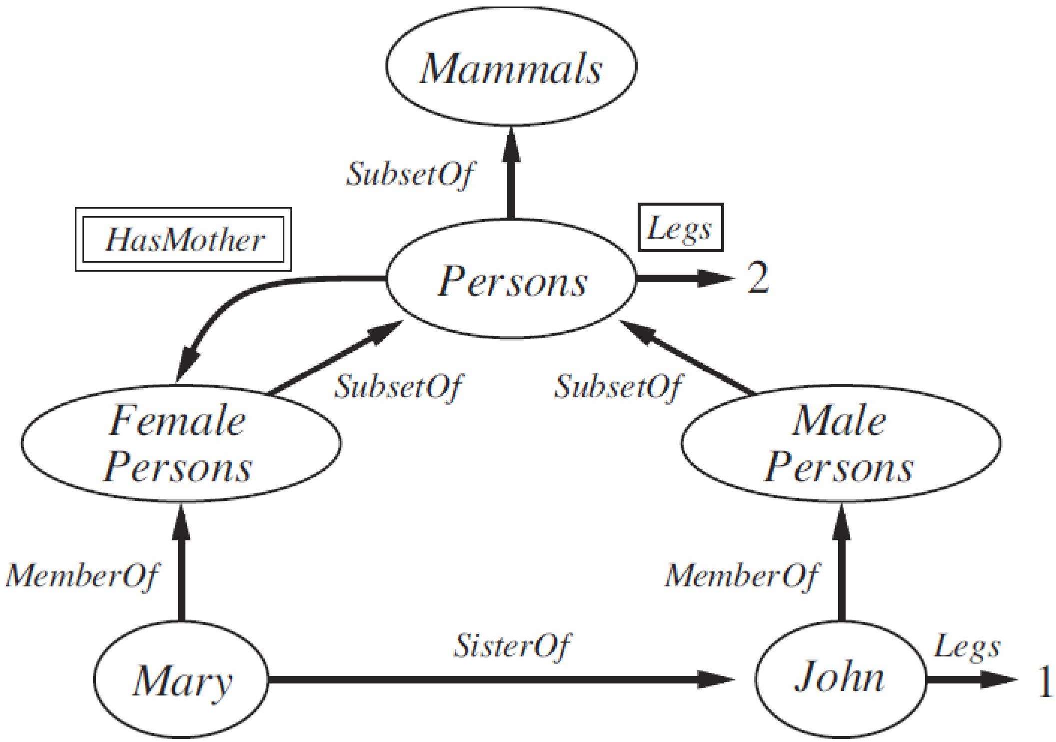
Mental events and mental objects

- Need to represent *beliefs* in self and other agents, e.g. for controlling reasoning, or for planning actions that involve others
- How are beliefs represented?
  - Beliefs are reified as *mental objects*
  - Mental objects are represented as *strings* in a language
  - Inference rules for this language can be defined
- Rules for reasoning about logical agents' use their beliefs

$$\forall a,p,q\ LogicalAgent(a) \wedge Believes(a,p) \wedge$$
$$Believes(a,"p \Rightarrow q") \Rightarrow Believes(a,q)$$
$$\forall a,p \quad LogicalAgent(a) \wedge Believes(a,p)$$
$$\Rightarrow Believes(a,"Believes(Name(a),p)")$$

Semantic networks

- Graph representation of categories, objects, relations, etc. (i.e. essentially FOL)
- Natural representation of inheritance and default values
  - E.g. a Person has normally 2 legs, but the default is *overridden* for John with 1 leg



Description logic (DL)

- FOL enables ascribing properties to objects, while DL allows formal specification of and reasoning about **definitions** and **categories**
- DL inference tasks:
  - √ Subsumption – Check if a category is a subset of another
  - √ Classification – Check if object belongs to a category
  - √ Consistency – Check if category definition is satisfiable
- DL evolved from semantic networks as a more formalized approach, still based on taxonomies
- DL in different versions is the logical foundation for the Semantic Web

CLASSIC - DL language

- CLASSIC is an early example of DL, in which definitions can be stated and reasoned about
- Simple category definitions
  - √ *Single = And(Unmarried, Adult)*
  - √ *Bachelor = And(Unmarried, Adult, Male)*
- CLASSIC can answer questions like
  - √ Is category Bachelor subsumed by category Single?
  - √ Is the individual Adam of category Bachelor?
- CLASSIC definitions can be translated to FOL, but inference in DL is more efficient

Defaults and non-monotonic logic

- Classical logic is **monotonic**: true statements remain true after new facts are added to KB
  - If $KB \models \alpha$, then $KB \wedge \beta \models \alpha$
- In the closed-world assumption (facts not mentioned assumed false), monotonicity is violated
  - If α is not mentioned in KB, then $KB \models \neg\alpha$, but $KB \wedge \alpha \models \alpha$
- Non-monotonic reasoning is widespread in common-sense reasoning
  - We assume default in absence of other input, and are able to retract assumption if new evidence occurs
- **Non-monotonic logics** support such reasoning

Circumscription (Norwegian: "begrensning")

- Circumscription is a more powerful version of the closed-world assumption
  - The idea is to specify particular predicates "as false as possible", i.e. false for every object except for those for which they are known to be true
- E.g. for the default that birds can fly
  - $Bird(x) \wedge \neg Abnormal(x) \Rightarrow Flies(x)$
- If *Abnormal* is **circumscribed**, a circumscriptive reasoner can
  - Assume $\neg Abnormal(x)$ unless the opposite is known
  - Infer *Flies(Tweety)* from *Bird(Tweety)*
  - Retract the conclusion if *Abnormal(Tweety)* is asserted

Truth maintenance systems

- Many inferences in the KB may have default status, and may need to be retracted in a process called **belief revision**
  - E.g. KB contains statement P (a default)
  - New evidence that P is not true: TELL(KB, $\neg$P)
  - To avoid contradiction: RETRACT(KB, P)
  - Other statements may have been added by P, e.g. Q if the KB contains P $\Rightarrow$ Q, and Q may also have to go
  - However, Q may also be true if the KB contains R $\Rightarrow$ Q, in which case Q need not be retracted after all
- Systems to handle such "book keeping" are called **Truth Maintenance Systems (TMS)**

Internet shopping world

- An agent that understands and acts in an internet shopping environment
- The task is to shop for a product on the Web, given the user’s product description
- The product description may be precise, in which case the agent should find the best price
- In other cases the description is only partial, and the agent has to compare products
- The shopping agent depends on having product knowledge, incl. category hierarchies

PEAS specification of shopping agent

- Performance goal
  - √ Recommend product(s) to match user's description
- Environment
  - √ All of the Web
- Actions
  - √ Follow links
  - √ Retrieve page contents
- Sensors
  - √ Web pages: HTML, XML

Outline of shopping agent behavior

- Start at home page of known web store(s)
  - √ Must have knowledge of relevant web addresses, such as www.amazon.com etc.
- Spread out from home page, following links to relevant pages containing product offers
  - √ Must be able to identify page relevance, using product category ontologies, as well as parse page contents to detect product offers
- Having located one or more product offers, agent must compare and recommend product
  - √ Comparison range from simple price ranking to complex tradeoffs in several dimensions

Summary

- An *ontology* is an encoding of vocabulary and relationships. *Special-purpose* ontologies can be effective within limited domains
- A *general-purpose* ontology needs to cover a wide variety of knowledge, and is based on categories and an event calculus
- It covers structured objects, time and space, change, processes, substances, and beliefs
- The general ontology can support agent reasoning in a wide variety of domains, including the Internet shopping world