10. Knowledge Representation: Modeling with Logic

Knowledge Representation and Reasoning

- Often, our agents need knowledge before they can start to act intelligently
- They then also need some reasoning component to exploit the knowledge they have
- Examples:
  - Knowledge about the important concepts in a domain
  - Knowledge about actions one can perform in a domain
  - Knowledge about temporal relationships between events
  - Knowledge about the world and how properties are related to actions

Categories and Objects

- We need to describe the objects in our world using categories
- Necessary to establish a common category system for different applications (in particular on the web)
- There are a number of quite general categories everybody and every application uses
The Upper Ontology: A General Category Hierarchy

Description Logics

- How to describe more specialized things?
- Use definitions and/or necessary conditions referring to other already defined concepts:
  - A parent is a human with at least one child
- More complex description:
  - A proud-grandmother is a human, is female with at least two children that are in turn parents whose children are all doctors

Reasoning Services in Description Logics

- Subsumption: Determine whether one description is more general than (subsumes) the other
- Classification: Create a subsumption hierarchy
- Satisfiability: Is a description satisfiable?
- Instance relationship: Is a given object instance of a concept description?
- Instance retrieval: Retrieve all objects for a given concept description

Special Properties of Description Logics

- Semantics of description logics can be given using ordinary PL1
  - Alternatively, DLs can be considered as modal logics
- Reasoning for most DLs is much more efficient than for PL1
- Nowadays, W3C standards such as OWL (formerly DAML+OIL) are based on description logics
Logic-Based Agents That Act

function KB-AGENT(percept) returns an action
    static KB, a knowledge base
    c, a counter, initially 0, indicating time
    TELL(KB, MAKE-PERCEPT-SENTENCE(percept, c))
    action ← ASK(KB, MAKE-ACTION-QUERY())
    c ← c + 1
    return action

Query (Make-Action-Query): ∃x Action(x,t)
A variable assignment for x in the WUMPUS world example should give the following answers:
  turn (right), turn(left), forward, shoot, grab, release, climb.

Reflex Agents

... only react to percepts.
Example of a percept statement (at time 5):
  Percept(stench, breeze, glitter, none, none, 5)
  1. Step: Abstraction of the basic properties
     ∨b,g,u,c,t [Percept(stench,b,g,u,c,t) ⇒ Stench(t)]
     ∨s,g,u,c,t [Percept(s,breeze,g,u,c,t) ⇒ Breeze(t)]
     ∨s,b,u,c,t [Percept(s,glitter,u,c,t) ⇒ AtGold(t)]
     ...
  2. Step: Choice of action
     ∀t [AtGold(t) ⇒ Action(grab,t)]
     ...
  But: Our reflex agent doesn’t know when it should climb out of the cave and cannot avoid an infinite loop.

Model-Based Agents

... have an internal model
• of all basic aspects of their environment,
• of the executability and effects of their actions,
• of further basic laws of the world,
• of their own goals.
Important aspect: How does the world change?
  → Situation calculus: (McCarthy 63).

Situation Calculus

• A way to describe dynamic worlds with PL1.
• States are represented by terms.
• The world is in state s and can only be altered through the execution of an action: do(a,s) is the resulting situation, if a is executed.
• Actions have preconditions and are described by their effects.
• Relations whose truth value changes over time are called fluents. Represented through a predicate with two arguments: the fluent and a state term. E.g.,
  At(x,s) means, for example, that in situation s, the agent is at position x;
  Holding(y,s), in situation s, the agent holds object y.
• Atemporal or eternal predicates, e.g. Portable(gold)
Example: WUMPUS-World

Let \( s_0 \) be the initial situation and
\[
\begin{align*}
    s_1 &= \text{do(forward, } s_0), \\
    s_2 &= \text{do(turn(right), } s_1), \\
    s_3 &= \text{do(forward, } s_2)
\end{align*}
\]

Description of Actions

Preconditions: In order to pick something up, it must be both present and portable:
\[
\forall x, s \ [\text{Poss}(\text{grab}(x), s) \iff \text{Present}(x, s) \land \text{Portable}(x)]
\]

In the WUMPUS-World:
\[
\text{Portable}(\text{gold}), \forall s \ [\text{At}(\text{gold}, s) \Rightarrow \text{Present}(\text{gold}, s)]
\]

Positive effect axiom:
\[
\forall x, s \ [\text{Poss}(\text{grab}(x), s) \Rightarrow \text{Holding}(x, \text{do(\text{grab}(x), s))}]
\]

Negative effect axiom:
\[
\forall x, s \ [\neg \text{Holding}(x, \text{do(\text{release}(x), s))}]
\]

The Frame Problem

We had: \( \text{Holding(} \text{gold}, s_0) \).

Following situation: \( \neg \text{Holding(} \text{gold}, \text{do(} \text{release}(\text{gold}), s_0)) \)?

We had: \( \neg \text{Holding(} \text{gold}, s_0) \).

Following situation: \( \neg \text{Holding(} \text{gold}, \text{do(} \text{turn(right)}, s_0)) \)?

- We must also specify which fluents remain unchanged!
- The frame problem: Specification of the properties that do not change as a result of an action.

\( \rightarrow \) Frame axioms must also be specified.

Number of Frame Axioms

\[
\begin{align*}
    \forall a, x, s \ [\text{Holding}(x, s) \land (a \neq \text{release}(x)) & \Rightarrow \text{Holding}(x, \text{do(a, s))}]) \\
    \forall a, x, s \ [\neg \text{Holding}(x, s) \land ((a \neq \text{grab}(x)) \lor \neg \text{Poss}(\text{grab}(x), s)) & \Rightarrow \neg \text{Holding}(x, \text{do(a, s))}]
\end{align*}
\]

Can be very expensive in some situations, since \( O(|F| \times |A|) \) axioms must be specified, \( F \) being the set of fluents and \( A \) being the set of actions.
Successor-State Axioms

A more elegant way to solve the frame problem is to fully describe the successor situation:

\[ \text{true after action } \iff [ \text{action made it true} \lor \text{already true and the action did not falsify it} ] \]

Example for grab:

\[ \forall a, x, s [\text{Holding}(x, a, s)] \iff (a = \text{grab}(x) \land \text{Poss}(a, s)) \lor (\text{Holding}(x, s) \land a \neq \text{release}(x))] \]

Can also be automatically compiled by only giving the effect axioms (and then applying explanation closure). Here we suppose that only certain effects can appear.

Limits of this Version of Situation Calculus

• No explicit time. We cannot discuss how long an action will require, if it is executed.
• Only one agent. In principle, however, several agents can be modeled.
• No parallel execution of actions.
• Discrete situations. No continuous actions, such as moving an object from A to B.
• Closed world. Only the agent changes the situation.
• Determinism. Actions are always executed with complete certainty.

\[ \rightarrow \text{Nonetheless, sufficient for many situations.} \]

Qualitative Descriptions of Temporal Relationships

• We can describe the temporal occurrence of event/actions:
  – absolute by using a date/time system
  – relative with respect to other event occurrences
  – quantitatively, using time measurements (5 secs)
  – qualitatively, using comparisons (before/overlap)

Allen’s Interval Calculus

• Allen proposed a calculus about relative order of time intervals
• Allows us to describe, e.g.,
  – Interval \( I \) occurs before interval \( J \)
  – Interval \( J \) occurs before interval \( K \)
• and to conclude
  – Interval \( I \) occurs before interval \( K \)
• 13 jointly exhaustive and pair-wise disjoint relations between intervals

\[ \rightarrow \text{Nonetheless, sufficient for many situations.} \]
Allen’s 13 Interval Relations

- $I < J$, $J > I$
  - before/after
- $I \preceq J$, $J \succ I$
  - starts
- $I \preceq J$, $J \preceq I$
  - meets
- $I \preceq J$, $J \succ I$
  - during
- $I \preceq J$, $J \preceq I$
  - overlaps
- $I = J$

Examples

- Using Allen’s relation system one can describe temporal configurations as follows:
  - $X < Y$, $Y \circ Z$, $Z > X$

- One can also use disjunctions (unions) of temporal relations:
  - $X (\prec, m) Y$, $Y (\circ, s) Z$, $Z > X$

Reasoning in Allen’s Relations System

- How do we reason in Allen’s system
  - Checking whether a set of formulae is satisfiable
  - Checking whether a temporal formula follows logically

- Use a constraint propagation technique for CSPs with infinite domains (3-consistency), based on composing relations

Constraint Propagation

- $X < Y s Z = X Z$
- $X < Y \circ Z = X Z$
- $X m Y s Z = X Z$
- $X m Y \circ Z = X Z$

Do that for every triple until nothing changes anymore, then CSP is 3-consistent
Concluding Remarks: Use of Logical Formalisms

- In many (but not all) cases, full inference in PL1 is simply too slow (and therefore too unreliable).
- Often, special (logic-based) representational formalisms are designed for specific applications, for which specific inference procedures can be used. Examples:
  - Description logics for representing conceptual knowledge.
  - James Allen’s time interval calculus for representing qualitative temporal knowledge.
  - Planning; instead of situation calculus, this is a specialized calculus (STRIPS) that allows us to address the frame problem.

→ Generality vs. efficiency
→ In every case, logical semantics is important!