KRR6: Knowledge engineering, Situation Calculus

Yannick Pencolé
Yannick.Pencole@anu.edu.au

23 Mar 2005
Outline

1. Knowledge engineering
   - Ontology

2. Situation Calculus
   - Situation calculus: an introduction
   - An example
   - Ontology of situation calculus
   - Frame problem
Ontological engineering

Definition

Ontological engineering: process for the representation of abstract concepts

Abstract concepts:
- Actions, Time, Objects, Knowledge (beliefs, possibilities...)
Upper Ontology
Categories and Objects

Definition

An **object** of the world always belongs to a **category**.

- organisation and simplification of a knowledge base
- use of **inheritance** (similar to object programming)
- **taxonomy**, **taxonomy hierarchy**

To make reasoning about the objects of the world, we generally make reasoning about the corresponding categories.
Categories and Objects: examples

**Example**

- An object is a member of a category
  
  \[ BB_9 \in BasketBalls \]

- A category can be a subclass of another category
  
  \[ BasketBalls \subset Balls \]

- All members of a category have some properties
  
  \[ \forall x \ x \in BasketBalls \Rightarrow Round(x) \]

- Members of a category can be recognised 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
  
  \[ Dogs \in DomesticatedSpecies \]
Composite objects

An object is part of another

To express that we can use terms like \( \text{PartOf} \) to say that one thing is part of another:

- \( \text{PartOf}(\text{Bucharest}, \text{Romania}) \)
- \( \text{PartOf}(\text{Romania}, \text{EasternEurope}) \)
- \( \text{PartOf}(\text{EasternEurope}, \text{Europe}) \)
- \( \text{PartOf}(\text{Europe}, \text{Earth}) \)

How to express \( \text{PartOf}(\text{Bucharest}, \text{Earth}) \)?

We only need to express the transitivity of the \( \text{PartOf} \) relation:

\[
\text{PartOf}(x, y) \land \text{PartOf}(y, z) \Rightarrow \text{PartOf}(x, z)
\]

\[
\text{Part}(x, x)
\]
Composite objects

Example

What is a biped?

\( \text{Biped}(a) \Rightarrow \)

\[
\exists l_1, l_2, b \ \text{Leg}(l_1) \land \text{Leg}(l_2) \land \text{Body}(b) \land \\
\text{PartOf}(l_1, a) \land \text{PartOf}(l_2, a) \land \text{PartOf}(b, a) \land \\
\text{Attached}(l_1, b) \land \text{Attached}(l_2, b) \land \\
l_1 \neq l_2 \land [\forall l_3 \ \text{Leg}(l_3) \land \text{PartOf}(l_3, a) \Rightarrow (l_3 = l_1 \lor l_3 = l_2)]
\]
How to organise categories. We need a description language. There are two families:

1. **Semantic networks**
   - a graphical aids for visualising a KB
   - efficient algorithms for inferring properties of an object on the basis of its category membership

2. **Description logic**
   - Formal language for constructing and combining category definition
   - efficient algorithms for subset/uperset category relationships detection
Semantic network

Example

![Semantic Network Diagram]

**Figure 10.9**  A semantic network with four objects (John, Mary, 1, and 2) and four categories. Relations are denoted by labeled links.

**Links:** $\text{SisterOf}(\text{Mary}, \text{John})$  $\text{Mary} \in \text{FemalePersons}$

$\text{FemalePersons} \subset \text{Persons}$...

**Single-boxed:** $\forall x \  x \in \text{Persons} \Rightarrow \text{Legs}(x, 2)$

**Double-boxed:**
$\forall x \  x \in \text{Persons} \Rightarrow [\forall y \ \text{HasMother}(x, y) \Rightarrow y \in \text{FemalePersons}]$
Description logics

**Definition**

Description logics are notations that are designed to make it easier to describe definitions and properties of categories.

**Example**

“Bachelors are unmarried adult males”

\[
\text{Bachelor} = \text{And}(\text{Unmarried}, \text{Adult}, \text{Male})
\]

which is a “syntaxical sugar” for

\[
\forall x \quad \text{Bachelor}(x) \iff \text{Unmarried}(x) \land \text{Adult}(x) \land \text{Male}(x)
\]
To represent aspects of complex worlds, ontology engineering.

The world is a set of objects that belong to categories.

Categories are inherited from other categories: Taxonomy.

The description of the world is based on compositionality of objects.

The result of the description in a formal language (like FOL) is a Knowledge base.

The knowledge base contains:
- Facts
- Axioms that allow to make reasoning about the facts in order to get other facts (by inference).
Outline

1 Knowledge engineering
   - Ontology

2 Situation Calculus
   - Situation calculus: an introduction
   - An example
   - Ontology of situation calculus
   - Frame problem
Situation calculus: an introduction

Static versus Dynamic

Previously, we saw FOL as a way to:
- Represent static worlds
- Prove atemporal sentences

Nothing is dynamical

Situation calculus

The language of Situation calculus is specifically designed for representing dynamically changing worlds.
- A dynamical world is a sequence of situations
- A situation is a static world
- All changes to the world are the result of actions
The figure presents the initial situation. If Jerry moves from room 1,1 to room 1,2 the world is changed and we have a new situation. How to express that? This is the purpose of Situation Calculus.
Definition

A **Situation** is a logical term:

1. an initial term $S_0$ called the *initial situation*;
2. any term generated by applying an *action* to a situation:

   $$\text{Result}(a, s)$$

   or

   $$\text{Do}(a, s)$$

where $a$ is an action and $s$ is a situation.
Situation: examples

A new situation could be:

$$\text{Result}(\text{Go}(\text{Room}(1, 1), \text{Room}(1, 2)), S_0)$$

If Jerry is in room (1,1) and goes to room (1,2) then we have a new situation.

Then, we can another situation:

$$\text{Result}(\text{Go}(\text{Room}(1, 2), \text{Room}(2, 2)), \text{Result}(\text{Go}(\text{Room}(1, 1), \text{Room}(1, 2)), S_0))$$
**Fluents**

**Definition**

A **Fluent** is a function or a predicate that vary from one situation to the next. By convention, the situation is always the last argument of the fluent.

$$function(a, b, c, S)$$

or

$$predicate(a, b, c, S)$$

where $a, b, c$ are terms and $S$ is a situation.

**Example**

- Jerry holds some cheese in the situation $s$ (predicate)
  
  \[Holding(Cheese(c), s)\]

- Here is the age of Tom in the situation $s$ (function)
  
  \[AgeOf(Tom, s)\]
An atemporal predicate/function is a predicate/function that do not vary from one situation to the next.

\[ \text{function}(a, b, c) \]

or

\[ \text{predicate}(a, b, c) \]

where \( a, b, c \) are terms (not situations).

Example

- Tom is a cat (predicate)
  \[ \text{Cat}(Tom) \]

- The left leg of Jerry (function)
  \[ \text{LeftlegOf}(Jerry) \]
An action is described with two axioms:

1. **Possibility axiom**: when it is possible to execute the action
2. **Effect axiom**: what happens when the action is executed

**Possibility axiom**

\[ \text{Preconditions} \Rightarrow \text{Poss}(a, s) \]

where \( a \) is the action and \( s \) the situation.

**Effect axiom**

\[ \text{Poss}(a, s) \Rightarrow \text{Changes that result from taking action } a \text{ in situation } s \]

where \( a \) is the action and \( s \) the situation. The changes are expressed with fluents.
Possibility axioms: examples

Example

Our agent: Jerry.

- Jerry can go between adjacent locations
  \[ \text{At}(Jerry, x, s) \land \text{Adjacent}(x, y) \Rightarrow Poss(Go(x, y), s) \]

- Jerry can grab some cheese in the current location
  \[ \text{Cheese}(c) \land \text{At}(Jerry, x, s) \land \text{At}(c, x, s) \Rightarrow Poss(\text{Grab}(c), s) \]

- Jerry can release some cheese that it is holding
  \[ \text{Holding}(c, s) \Rightarrow Poss(\text{Release}(c), s) \]
Effect axioms: examples

Example

Jerry goes between adjacent locations

\[ \text{Poss}(\text{Go}(x, y), s) \implies \text{At}(\text{Jerry}, y, \text{Result}(\text{Go}(x, y), s)) \]

Jerry grabs some cheese in the current location

\[ \text{Poss}(\text{Grab}(c), s) \implies \text{Holding}(c, \text{Result}(\text{Grab}(c), s)) \]

Jerry releases some cheese that it is holding

\[ \text{Poss}(\text{Release}(c), s) \implies \neg \text{Holding}(c, \text{Result}(\text{Release}(c), s)) \]
And now? What happens?

We have an ontology (symbols, set of axioms) based on a formal system (FOL), let make some reasoning about situations!

**Definition**

*Projection/prediction* task: an agent (like Jerry) should be able to deduce the outcome of a given sequence of actions.

**Example**

- “going to room (2,3) is not a good idea”
- “going to room (1,2) is not dangerous”
- “going to room (4,3) is an excellent idea”
And now? What happens?

**Definition**

**Planning** task: an agent should be able to **find** a sequence that achieves a desired effect

**Example**

**Desired effect**: “I want to get some cheese and Tom must not see me”

A solution: “Go to room (2,1) then Go to room (3,1) then Go to room (4,1) then Go to room (4,2) then Go to room (4,3) then grab the cheese then Go to room (4,2) then Go to room (4,1) then Go to room (3,1) then Go to room (2,1) then Go to room (1,1) then release the cheese”
Planning: example

Example

Initial situation $S_0 = \begin{bmatrix}
\text{Jerry} & \text{Cheese} \\
1 & 2 & 3 & 4 \\
\end{bmatrix}$

Goal: “Jerry wants the cheese in 1,1”

In the initial situation, we must have:

\[
At(Jerry, Room(1, 1), S_0) \land At(Gruyere, Room(1, 2), S_0) \\
Cheese(Gruyere) \land Adjacent(Room(1, 1), Room(1, 2)) \land \\
Adjacent(Room(2, 1), Room(1, 1))
\]

(+ other things...)

The goal is also a sentence: $\exists seq At(Gruyere, Room(1, 1), Result(seq, S_0))$

If the sentence is entailed by $KB$ then we prove there is a plan for Jerry! If the inference algorithm is constructive then $seq$ will be substituted with a term representing one plan!
Example

From $S_0$, we can get: (the action is $Go(Room(1,1), Room(1,2))$)

$$At(Jerry, Room(1,2), Result(Go(Room(1,1), Room(1,2)), S_0))$$

From this new sentence and $KB$ we should be able to prove:

$$At(Gruyere, Room(1,2), Result(Go(Room(1,1), Room(1,2)), S_0))$$

... but we can’t! (no axioms from KB can infer that)

Effect axioms say what changes but don’t say what stays the same. The fact that Jerry moves from (1,1) to (1,2) does not change the fact that the cheese is still in (1.2).

Something is missing: frame problem
Definition

Purpose: representing all the things that stay the same. We need frame axioms that do say what stay the same.

Efficiency

The solution for solving the frame problem must be efficient.

- Frame axioms are numerous (an action generally changes a few fluents)
- One solution is to use successor-state axioms (not complete)
  - It solves the representational frame problem
Definition

Instead of writing effect axioms,

\[ \text{Poss}(a, s) \Rightarrow \text{Changes that result from taking action } a \text{ in } s \]

we use \textit{successor-state axioms}:

\[ \text{Poss}(a, s) \Rightarrow \]

\[ \text{Fluent is true in situation } s \iff \]

\[ [\text{The effect of action } a \text{ made it true } \lor \]
\[ \text{It was true before and action } a \text{ left it alone}] \]
Representational frame problem: example

Example

Poss(a, s) ⇒

\[ \text{At}(\text{Jerry}, y, \text{Result}(a, s)) \iff [a = \text{Go}(x, y) \lor (\text{At}(\text{Jerry}, y, s) \land a \neq \text{Go}(y, z))] \]

Informally: if a is possible in situation s then Jerry is in y in the “next situation” if either the action a is “go to y” or Jerry was “already” in y and a is not a movement action from y.

Poss(a, s) ⇒

\[ \text{Holding}(c, \text{Result}(a, s)) \Rightarrow [a = \text{Grab}(c) \lor (\text{Holding}(c, s) \land a \neq \text{Release}(c))] \]

Informally: if a is possible in situation s then Jerry holds the cheese in the “next situation” if either the action a is “Grab the cheese” or Jerry was already holding the cheese and the current action a is not to release it.
Summary

- Situation calculus: a way to represent actions, events in the world and make reasoning
  - Initial situation + axioms for actions
- Use of inference algorithms to do Deductive planning
- Frame Problem:
  - Use of frame axioms to express what the actions don’t change in the world.