Analysis of Control Programs Using Action Models

J. Ramirez & H. Geffner
Universidad Simón Bolívar
Caracas, Venezuela

http://www.ai.usb.ve

Motivation

$$\begin{array}{lcl} \text{reached(goal)} & \rightarrow & \text{done} \\ \text{see(goal)} \wedge \text{facing(goal)} & \rightarrow & \text{move} \\ \text{see(goal)} & \rightarrow & \text{rotate} \end{array}$$

- **Intuition:** Program is executable and leads agent to goal
- **Task:** Formalize this intuition
- **Why:**
  - Useful to build and understand programs
  - Benchmark for theories of action and knowledge

The idea

- Given a control theory $C = (D, P, O)$:
  - D: describes effects of actions
  - P: describes when/what actions are done
  - O: observations
- Establish conditions under which:
  - C is executable (action + know preconds)
  - C leads agent to goal
- Need language and semantics for D, P & O

Example: hitting a nail into a wall

h: distance head to wall

e: random value in $[0, 0.9]$

Δ: small parameter

- Actions D:

$$\begin{array}{lcl} hit \wedge h > \Delta & \rightarrow & h := h \cdot e \\ hit \wedge h \leq \Delta & \rightarrow & h := 0 \\ hit & \rightarrow & obs(h) \end{array}$$

- Program P:

$$h > 0 \rightarrow hit$$

- Observations O:

$$h[0] = 10$$

- Result:
  - Program executable and lead agent to $h = 0$
  - Same true for any $h[0] > 0$

Rest of the talk

- language and semantics of domain theories
- language and semantics of control theories
- actions and knowledge preconditions
- application: simple navigation program

Domain theories [Geffner & Wainer 97]

- **Symbols:** fixed, parameter, random, fluent and action symbols
- **Language:** FOL without variables
- **Rules:**

$$Action \wedge Condition \;\rightarrow\; Fluent := Term$$

$$\text{e.g.,}\;\; hit \wedge h \geq \Delta \;\rightarrow\; h := h \cdot e$$

- **Observations:**

$$Expr[time] = value$$

$$\text{e.g.,}\;\; h[0] = 10$$

Domain Theories (Semantics)

- D determines state trajectories $s_0, s_1, s_2, \ldots$
- Each state s_i: interpretation over language
- Trajectories respect meaning of different symbols:
  - $c^{s_i} = c^{s_{i+1}}$ if c a **parameter**
  - $r^{s_i} \in D_r$ if r is **random symbol**
  - $$f^{s_{i+1}} = \begin{cases} g^{s_i} & \text{if } A \wedge C \rightarrow f := g \\ & \text{and } A^{s_i} = C^{s_i} = \textbf{true} \\ f^{s_i} & \text{otherwise} \end{cases}$$
- Trajectory $s_0, s_1, s_2, \ldots$ satisfies observation

$$Expr[i] = value \text{ iff } Expr^{s_i} = value$$

Definitions

- **Complete definitions**

$$symbol \stackrel{\text{def}}{=} expr$$

meaning

$$symbol^{s_i} = expr^{s_i}$$

e.g.,

$$tall \stackrel{\text{def}}{=} h > 10$$

- **Partial definitions**

$$atom \text{ if } wff$$

meaning

$atom^{s_i}$ **false** unless one such wff **true**

e.g.,

$$blocked \text{ if } facing(obj) \wedge close(obj)$$

Control Programs

- **Language:** sequence of condition-action pairs

$$P : \{c_1 \rightarrow a_1, c_2 \rightarrow a_2, \ldots, c_n \rightarrow a_n\}$$

- **Semantics:** trajectory s_0, s_1, ... satisfies P iff for every s_i, $i = 0, 1, \ldots$

$$s_i \models a_j$$

iff

$$s_i \models c_j \text{ and } s_i \not\models c_k \text{ for } k \leq j$$

Semantics of Control Theories

$C = (D, P, O)$ leads agent to $expr = value$

if in every **model** s_0, s_1, s_2, ... of C

(i.e., every traj. that satisfies D, P & O)

$expr^{s_i} = value \; \forall i \geq$ some k

Example: Simple Navegation Program

- Program

$$
\begin{array}{lcl}
\text{reached(goal)} & \rightarrow & \text{done} \\
\text{see(goal)} \wedge \text{facing(goal)} & \rightarrow & \text{move} \\
\text{see(goal)} & \rightarrow & \text{rotate} \\
\neg \text{ blocked} & \rightarrow & \text{move} \\
\textbf{true} & \rightarrow & \text{rotate}
\end{array}
$$

- Rules:

$$
\begin{array}{lcl}
\text{rotate} & \rightarrow & \text{angle} := \text{angle} + \delta\text{angle} \\
\text{move} & \rightarrow & \text{xpos} := \text{xpos} + \delta\text{dist} \cos(\text{angle}) \\
\text{move} & \rightarrow & \text{ypos} := \text{ypos} + \delta\text{dist} \sin(\text{angle})
\end{array}
$$

Example: Simple Navegation Program (cont)

- Definitions:

$$
\begin{aligned}
dist(obj) &\stackrel{\text{def}}{=} \sqrt{(xpos - x(obj))^2 + (ypos - y(obj))^2} \\
angle(obj) &\stackrel{\text{def}}{=} angle + \theta(obj) \\
\theta(obj) &\stackrel{\text{def}}{=} \sin^{-1} \frac{ypos - y(obj)}{dist(obj)} \\
reached(obj) &\stackrel{\text{def}}{=} facing(obj) \wedge close(obj) \\
close(obj) &\stackrel{\text{def}}{=} dist(obj) \leq \delta dist \\
see(obj) &\stackrel{\text{def}}{=} dist(obj) \leq vdist \wedge angle(obj) \leq vfield
\end{aligned}
$$

- If $vfield = 360^o$ and $vdist = \infty$, program leads to $reached(goal)$

Executability of Programs

- **Action Preconditions**

  $$move \supset \neg blocked$$

  meaning

  $$move^{s_i} = \textbf{true} \text{ must imply } blocked^{s_i} = \textbf{false}$$

- **Knowledge preconditions:**

  if c_j is **first** true condition in s_i of model

  then value of c_1, c_2, ..., c_j **known** in s_i

Example: Same navigation program

$obs(x)$: means value of expr x is observable

- Information gathering clauses:

$$obs(dist(obj)) \stackrel{\text{def}}{=} see(obj)$$
$$obs(angle(obj)) \stackrel{\text{def}}{=} see(obj)$$
$$obs(see(obj)) \stackrel{\text{def}}{=} \textbf{true}$$

- Value of e is **known** in state s

  if denotation of e is equal in all the states that preserve the denotation of observables in s

- From definitions, $reached(goal)$ and $facing(goal)$ always **known**

- Program leads to goal and is **executable**

Summary

- Theory of action and knowledge used to formalize properties of control programs
- Analysis can be extended to richer control structures
- Relevance to **synthesis** not clear yet