

COMP9814: Artificial Intelligence

Planning

Wayne Wobcke

Room J17-433

wobcke@cse.unsw.edu.au

Based on slides by Maurice Pagnucco

Overview

- Reasoning About Action
- Situation Calculus
- States/Situtations
- Domain Constraints
- Actions
- The Frame Problem
- STRIPS Planner
- Conclusion

Planning

- Many of the environments about which we wish to reason are dynamic in nature
- That is, they are constantly changing due to the performing of actions
- For example, consider a robot whose job it is to deliver packages in an office environment
- In this lecture we shall consider some of the issues that arise when representing and reasoning about a changing world
- References:
 - ▶ Ivan Bratko, [Prolog Programming for Artificial Intelligence](#), Addison-Wesley, 2001. (Chapter 17)
 - ▶ Stuart J. Russell and Peter Norvig, [Artificial Intelligence: A Modern Approach](#), Second Edition, Pearson Education, 2003. (Chapter 11)

Reasoning About Action

- One method to reason about action is to simply change the agent's knowledge base
- Erase some sentence(s) that should no longer be true and add sentences that will now be true (i.e. after performing action)
- However, we can only answer questions about the current state
- It will not be possible to reason about past or future states
- On the other hand, if all we want to do is reason about which actions to perform, this may be a viable approach (shall return to it later)

Situation Calculus

- The **situation calculus** is a way of describing change in first-order predicate calculus
- **Situation/State** — a snapshot of the world at a particular point in time
- We need to consider:
 - ▶ the state of the world
 - ▶ actions that change state of the world

State of the World

- **State** — description of what holds (is true) in the world at a particular point in time
- **Situation** — state + time (includes history)
- Method 1:

$$on(C, A, S_1), on(A, Table, S_1), on(B, Table, S_1)$$

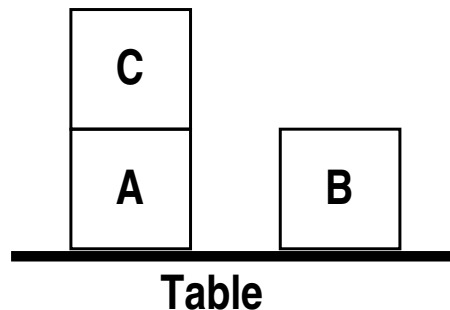
$$clear(B, S_1), clear(C, S_1)$$
- Method 2:

$$holds(on(C, A), S_1), holds(on(A, Table), S_1), holds(on(B, Table), S_1)$$

$$holds(clear(B), S_1), holds(clear(C), S_1)$$
- Initial situation: $S_0/init$
- **Note:** we **reify** states (i.e. make them entities in our formalisation)

The Blocks World

- We shall consider one of the more famous AI domains — the **blocks world**
- In the blocks world we have blocks that can be placed on the table and can be stacked on each other



Domain Constraints

- (Also known as **state constraints**)
- True at all (legal) states even though they involve state-dependent relations:

x is on the table iff it is not on top of another block

$$\forall x \forall s (on(x, Table, s) \leftrightarrow \neg \exists y (on(x, y, s) \wedge (y \neq Table)))$$

x is clear iff there is no block on top of it

$$\forall x \forall s (clear(x, s) \leftrightarrow \neg \exists y on(y, x, s))$$

if y is a block and there is another block on it, then y is not clear

$$\forall x \forall y \forall s (on(x, y, s) \wedge (y \neq Table) \rightarrow \neg clear(y, s))$$

Actions

$do(A(\dots), S)$ — “Do action A in situation S ”

- In fact, **do** is a function returning a situation which is the result of performing (doing) action A in situation S (if that is possible)
- For example, “move block x from y to z ” and “clear x ”
 - ▶ $\forall x \forall y \forall z \forall s (on(x, y, s) \wedge clear(x, s) \wedge clear(z, s) \wedge (x \neq z) \rightarrow on(x, z, do(move(x, y, z), s)))$
 - ▶ $\forall x \forall y \forall z \forall s (on(x, y, s) \wedge clear(x, s) \wedge clear(z, s) \wedge (x \neq z) \rightarrow \neg on(x, y, do(move(x, y, z), s)))$
 - ▶ $\forall x \forall y \forall z \forall s (on(x, y, s) \wedge clear(x, s) \wedge clear(z, s) \wedge (x \neq z) \wedge (y \neq z) \rightarrow clear(y, do(move(x, y, z), s)))$
 - ▶ $\forall x \forall y \forall z \forall s (on(x, y, s) \wedge clear(x, s) \wedge clear(z, s) \wedge (x \neq z) \wedge (z \neq Table) \rightarrow \neg clear(z, do(move(x, y, z), s)))$

Ramification Problem

- What are the ramifications (direct and indirect effects) of performing an action?

Qualification Problem

- What qualifications (preconditions) do we require in specifying actions and their effects?

The Frame Problem

- Action descriptions are not complete:
 - ▶ they describe what changes
 - ▶ they do **not** specify what stays the same
- The **Frame Problem**:
 - ▶ The problem of characterising those aspects of the state that are not changed by an action
- One solution — **Frame Axioms**
 - ▶ $on(x, y, s) \wedge (x \neq u) \rightarrow on(x, y, do(move(u, v, z), s))$
 - ▶ $\neg on(x, y, s) \wedge ((x \neq u) \vee (y \neq z)) \rightarrow \neg on(x, y, do(move(u, v, z), s))$
 - ▶ $clear(u, s) \wedge (u \neq z) \rightarrow clear(u, do(move(x, y, z), s))$
 - ▶ $\neg clear(u, s) \wedge (u \neq y) \rightarrow \neg clear(u, do(move(x, y, z), s))$

Reasoning with the Situation Calculus

- If we would like to determine a plan to achieve goal $\Gamma(s)$, prove $\exists s \Gamma(s)$ using action theory and initial state
- For example, $\exists s on(B, Table, s)$ using:
 - ▶ $on(B, A, S_0)$
 - ▶ $on(A, C, S_0)$
 - ▶ $on(C, Table, S_0)$
 - ▶ $clear(B, S_0)$
 - ▶ $clear(Table, S_0)$
 - ▶ $on(x, y, s) \wedge clear(x, s) \wedge clear(z, s) \wedge (x \neq z) \rightarrow on(x, z, do(move(x, y, z), s))$
- (Also require **equality axioms**: $A = A$, $\neg(A = B)$, etc.)
- Obtain $s = do(move(B, A, Table), S_0)$

Planning

- **Plan** — sequence of actions to achieve some goal
- **Planner** — problem solver that produces plans
- Planning in the situation calculus can be achieved by query answering
e.g. $\exists s \text{ on}(B, C, s)$ —
answer $s = \text{do}(\text{move}(B, \text{Table}, C), \text{do}(\text{move}(C, A, \text{Table}), S_0))$

STRIPS Example

- **Action Description:** $\text{move}(x, y, z)$
- **Preconditions:** $\text{on}(x, y), \text{clear}(x), \text{clear}(z)$
- **Delete List:** $\text{clear}(z), \text{on}(x, y)$
- **Add List:** $\text{on}(x, z), \text{clear}(y), \text{clear}(\text{Table})$
- (Last of these is added so that table is always clear)
- Use search to achieve goal (can also use theorem proving)
- Recursive STRIPS — when faced with a conjunctive goal, satisfy each of the goals in turn

STRIPS

- An early planner — drop mention of states
- **Action Description** — name of action; returned to environment in order to do something
- **Preconditions** — action can be performed in a state only if precondition holds in state prior to action being performed
- **Delete List** — literals to be deleted from the state description after action is performed
- **Add List** — literals to be added to the state description after action is performed
- **STRIPS Assumption** — any relations (predicates) not mentioned in the delete list remain true after the action is performed (a way of solving the frame problem)

Conclusion

- Reasoning about actions is a very interesting area of Artificial Intelligence
- We have seen that a number of challenging problems arise that we must deal with in order to reason effectively
- One of the problems, however, is the possible proliferation of axioms
- The search continues for a **concise** solution to the frame problem (and associated problems)
- Other formalisms include the **event calculus**