Situation Calculus

Informatics 2D: Reasoning and Agents

Adapted from slides provided by Michael Rovatsos

Using Logic to Plan

- We need ways of:
  - representing the world.
  - representing the goal.
  - representing how actions change the world.
- We haven't said much about the last.
  - Difficulty: After an action, new things are true, and some previously true facts are no longer true.

Situations

- Introduce a notion of *situations*, which are logical terms
  - Consist of initial situation (usually called S_0) and all situations generated by applying an action to a situation.
- State facts about situations.
  - By relativizing predications to situations.
  - E.g., instead of saying just **On(A,B)**, say (somehow) **On(A,B)** in situation $\mathbf{S_0}$
- Actions are thus
  - performed in a situation, and
  - produce new situations with new facts.
  - Examples: Forward and Turn(Right)

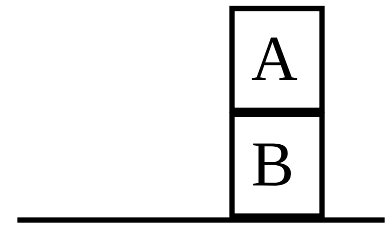
Representing Predications Relative to a Situation

- Can add an argument for a situation to each predicate that can change.
  - E.g., instead of **On(A,B)**, write **On(A,B,S_0)**
- Alternatively, introduce a predicate **Holds** and turn **On**, etc., into *functions*:
  - E.g., **Holds(On(A,B),S_0)**
  - What do things like **On(A,B)** now mean?
    - Either a category of situations, in which **A** is on **B**, or a set of those situations.

How This Will Work

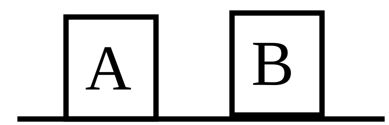
- Before some action, we might have in our KB:

  On(A,B,S_0)
  On(B,Table,S_0)
  ….



- After an action that moves A to the table, say, we add

  Clear(B,S_1)
  On(A,Table,S_1)



- All these propositions are true. We have dealt with the issue of change, by keeping track of what is true when.

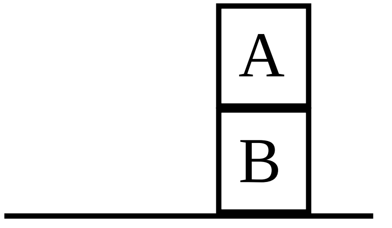
Same Thing, Slightly Different Notation

- Before :

  Holds(On(A,B),S_0)

  Holds(On(B,Table),S_0)

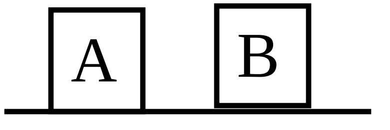
  …



- After, add

  Holds(Clear(B),S_1)

  Holds(On(A,Table),S_1)



Representing Actions

- Need to represent:
  - Results of doing an action
  - Conditions that need to be in place to perform an action.
- For convenience, we will define *functions* to abbreviate actions:
  - E.g., **Move(A,B)** denotes the *action type* of moving **A** onto **B**.
  - These are action *types*, because actions themselves are specific to time, etc.
- Now, introduce a *function* **Result**, designating "the situation resulting from doing an action type in some situation".
  - E.g., **Result(Move(A,B),S_0)** means "the situation resulting from doing an action of type **Move(A,B)** in situation **S_0**".

How This Works

- Keep in mind that things like

  Result(Move(A,B),S_0)

  are *terms*, and denote *situations*.

  They can appear anywhere we would expect a situation.
- So we can say things like

  S_1=Result(Move(A,B),S_0),

  On(A,B,Result(Move(A,B),S_0)),

  On(A,B,S_1), etc.

  (Alternatively, **Holds(On(A,B),Result(Move(A,B),S_0))**, etc.)

Axiomatizing Actions

- Now, we can describe the results of actions, together with their preconditions.
- E.g., 'If nothing is on x and y, then one can move x to on top of y, in which case x will then be on y.'

$$\forall x,y,s\ \mathbf{Clear(x,s)} \wedge \mathbf{Clear(y,s)} \Rightarrow \mathbf{On(x,y,Result(Move(x,y),s))}$$

- Alternatively:

$$\forall x,y,s\ \mathbf{Holds(Clear(x),s)} \wedge \mathbf{Holds(Clear(y),s)} \Rightarrow \mathbf{Holds(On(x,y),\ Result(Move(x,y),s))}$$

- This is an *effect axiom*.
  - It includes a precondition as well.

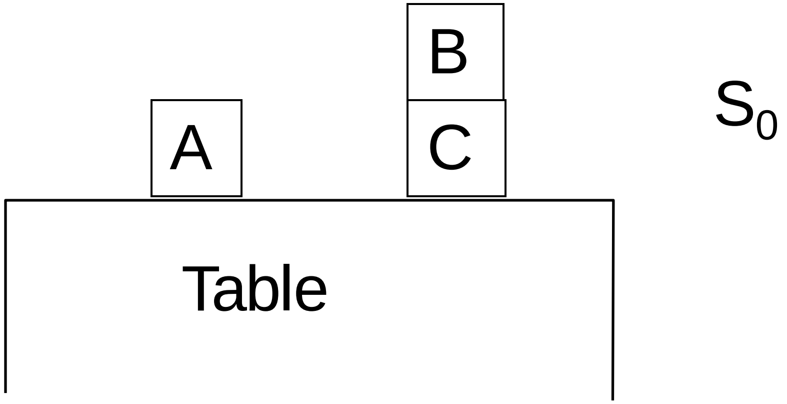
Situation Calculus

- This approach is called the *situation calculus*.
- We axiomatize all our actions, then use a general theorem prover to prove that a situation exists in which our goal is true.
- The actions in the proof would comprise our plan.

A Very Simple Example

- KB:

  On(A,Table,S_0)
  On(B,C,S_0)
  On(C,Table,S_0)
  Clear(A,S_0)
  Clear(B,S_0)
  and axioms about actions, etc.



- Goal:

  ∃s'. On(A,B,s')

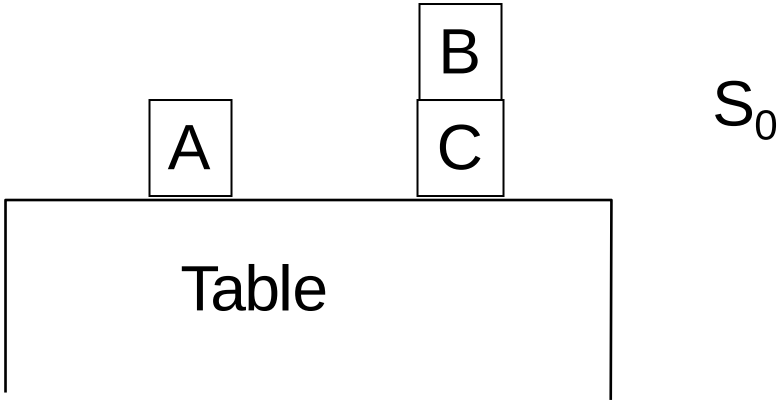
What happens?

- We try to prove **On(A,B,s')** for some s'
  - Find axiom

    $$\forall x,y,s.\ \mathbf{Clear(x,s)} \wedge \mathbf{Clear(y,s)}$$
    $$\Rightarrow \mathbf{On(x,y,Result(Move(x,y),s))}$$

  - By chaining, e.g., goal would be true if we could prove **Clear(A,s)** $\wedge$ **Clear(B,s)** by backward chaining.
  - But both are true in $\mathbf{S_0}$, so we can conclude **On(A,B,Result(Move(A,B),**$\mathbf{S_0}$**))**
- We are done!
- We look in the proof and see only one action, **Move(A,B)**, which is executed in situation $\mathbf{S_0}$, so this is our plan.

Tougher Example: Same Initial World, Harder Goal



- KB:

  On(A,Table,S_0)

  On(B,C,S_0)

  On(C,Table,S_0)

  Clear(A,S_0)

  Clear(B,S_0)

  and axioms about actions, etc.

- Goal:

  $\exists$s' On(A,B,s') $\wedge$ On(B,C,s')

  (Intuitively, really not harder: B already on C, and we just showed how to make A on B.)

With Goal **On(A,B,s') ∧ On(B,C,s')**

- Suppose we try to prove the first subgoal, **On(A,B,s')**.
  - Use same axiom

    ∀x,y,s. Clear(x,s) ∧ Clear(y,s)

    ⇒ On(x,y, Result(Move(x,y),s))
  - Again, by chaining, we can conclude **On(A,B,Result(Move(A,B),S_0))**.
  - Abbreviating **Result(Move(A,B),S_0)** as **S_1**, we have **On(A,B,S_1)**.
- Substituting for s' in our other subgoal makes that **On(B,C,S_1)**. If this is true, we're done.
- But we have no reason to believe this is true!
- Sure, **On(B,C,S_0)**, but how does the planner know this is still true, i.e., **On(B,C,S_1)**?
- In fact, it doesn't, so it fails to find an answer!

The Frame Problem

- We have failed to express the fact that everything that isn't changed by an action stays the same.
- Can fix by adding *frame axioms*. E.g.:

  $\forall$x,y,z,s.

  Clear(x,s) $\Rightarrow$ Clear(x, Result(Paint(x,y),s))

  ...
- There are *lots* of these!
- Is this a big problem?

Better Frame Axioms

- Can fix with neater formulation:

  $\forall x,y,s,a.$

  $On(x,y,s) \wedge (\forall z.\ a=Move(x,z) \Rightarrow y=z)$

  $\Rightarrow On(x,y,Result(a,s))$

- Can combine with effect axioms to get *successor-state axioms*:

  $\forall x,y,s,a.$

  $On(x,y,Result(a,s)) \Leftrightarrow$

  $On(x,y,s) \wedge (\forall z.\ a=Move(x,z) \Rightarrow y=z)$

  $\vee\ (Clear(x,s) \wedge Clear(y,s) \wedge a=Move(x,y))$

How Does This Help Our Example?

- We want to prove

  On(B,C,Result(Move(A,B),S_0)) given that **On(B,C,S_0)**

  Axiom says

  ∀x,y,s,a. On(x,y,Result(a,s)) ⇔

  On(x,y,s) ∧ (∀z. a=Move(x,z) ⇒ y=z)

  ∨ (Clear(x,s) ∧ Clear(y,s) ∧ a=Move(x,y))

- So need to show

  On(B,C,S_0) ∧ (∀z. Move(A,B)=Move(B,z) ⇒ C=z) is true, which is easy

  - The first conjunct is in the KB.
  - The second one is true since actions are the same only if they have the same name and involve the exact same objects i.e.

$$A(x_1, \ldots, x_m) = A(y_1, \ldots, y_m) \textit{ iff } x_1 = y_1 \wedge \ldots \wedge x_m = y_m$$

so **Move(A,B)=Move(B,z)** is false.

Note: Another assumption in KB: $A(x_1, \ldots, x_m) \neq B(y_1, \ldots, y_n)$

These are known as Unique Action Axioms

For Refutation Theorem Proving: (Dual) Skolemisation

- Suppose $\forall x. \exists y. G(x,y)$ is goal in resolution refutation.
- So, we need to negate the goal:

$$\neg \forall x. \exists y. G(x,y) \equiv \exists x. \forall y. \neg G(x,y)$$

- Then skolemise (i.e drop existential quantifier):

$$\neg G(X_0,y)$$

- Intuition:
  - y is to be unified to construct *witness*.
  - X_0 must not be instantiated.
- Similar story for GMP, but goal not negated, i.e. $G(X_0,y)$, for some y, is used as the goal.

KB and Axioms as Clauses

Constants: A, B, C, S0
Variables: a, x, y, s

Initial State

On(A,Table,S0)

On(B,C,S0)

On(C,Table,S0)

Clear(A,S0)

Clear(B,S0)

Goal

$\neg On(A,B,s') \lor \neg On(B,C,s')$

Effect Axiom

$\neg Clear(x,s) \lor \neg Clear(y,s) \lor$
$\quad On(x,y, Result(Move(x,y),s))$

Frame Axioms

Skolem function (→ Z)

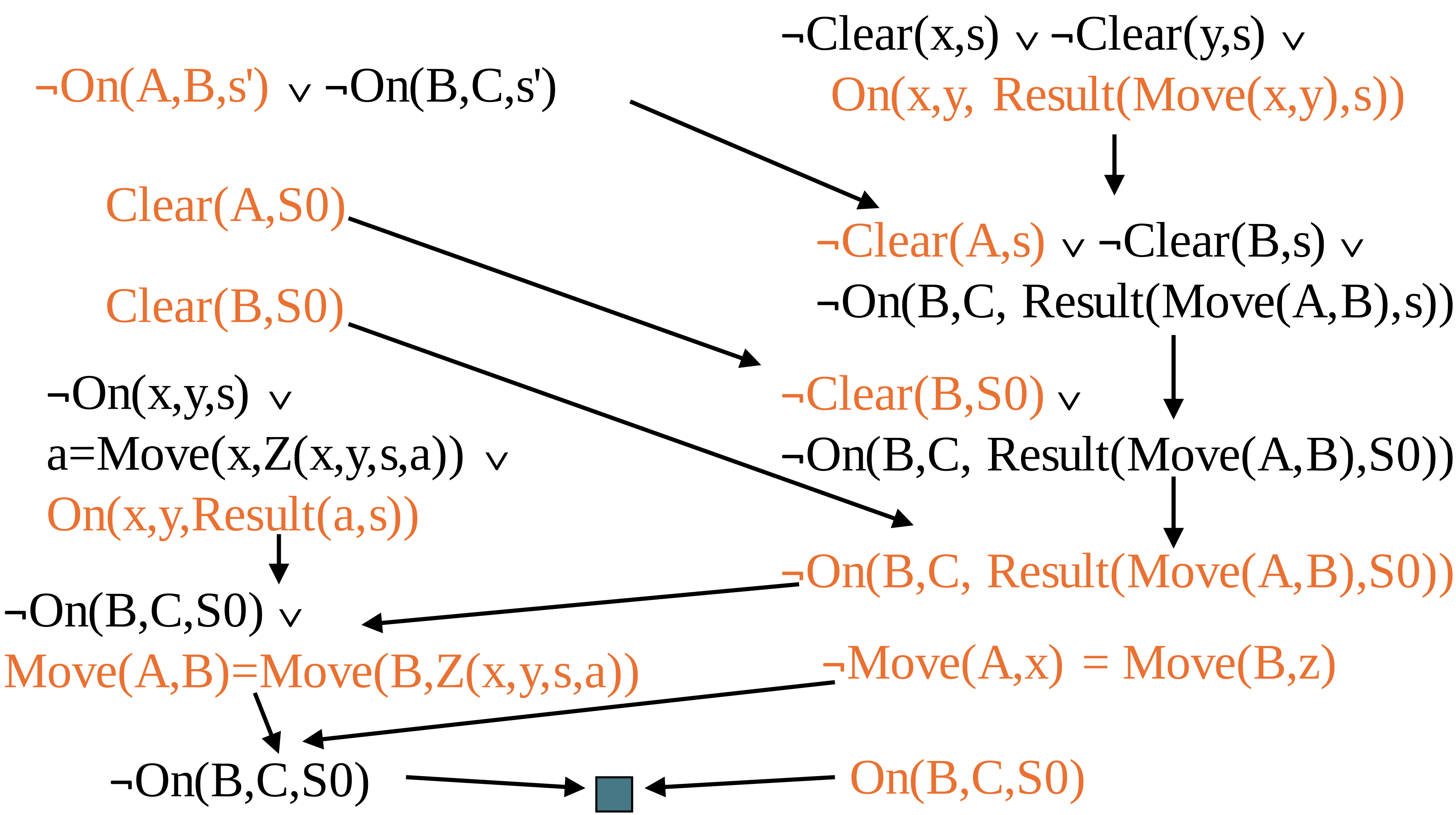
$\neg On(x,y,s) \lor a=Move(x,Z(x,y,s,a)) \lor On(x,y,Result(a,s))$

$\neg On(x,y,s) \lor \neg y=Z(x,y,s,a) \lor On(x,y,Result(a,s))$

Unique Action Axioms: $\neg\ Move(A,B)=Move(B,z)$, etc

Unique Name Axiom: disequality for every pair of constants in KB

Resolution Refutation



Frame problem partially solved

- This solves the representational part of the frame problem.
- Still have to compute that everything that was true that wasn't changed is still true. Inefficient (as is general theorem proving).
- Solution: Special purpose representations, special purpose algorithms, called *Planners*.

Summary

PLANNING

SITUATIONS

FRAME PROBLEM