CIS587 - Artificial Intelligence

Planning, Situation Calculus
STRIPS planning
Partial-Order Planning

Planning

- **Propositional and first-order logic**
  - formalism for representing the knowledge about the world and ways of reasoning
  - Statements about the world are true or false
- **The real-world:**
  - is dynamic; can change over time
  - an agent can actively change the world through its actions
- **Planning problem:** find sequence of actions that lead to a goal
- **Challenges:**
  - Build a representation language for modeling action and change
  - Design of special search algorithms for a given representation

Planning and search

Planning – a special type of a search problem

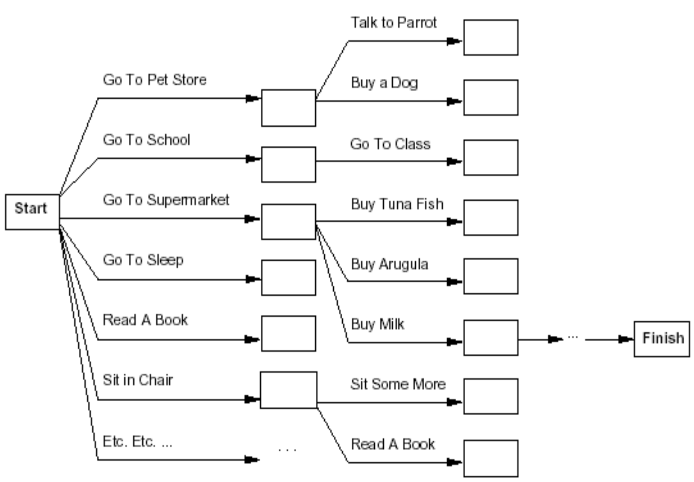
What if we use a standard search formulation?

Search problem:

- State space – a set of states of the world among which we search for the solution.
- Initial state. A state we start from.
- Operators. Map states to new states.
- Goal condition. Test whether the goal is satisfied.

- Assume a simple problem of buying things:
  - Get a quarter of milk, bananas, cordless drill

Planning search - Example



A huge branch factor !!! Goals can take multiple steps to reach!!!

Planning

To address these problems planning systems:

- Open state, action and goal representations to allow selection, reasoning. Make things visible and expose the structure.
  – Use FOL or its restricted subset
- Add actions to the plan sequence wherever and whenever it is needed
  – Drop the need to construct solutions sequentially from the initial state
- Apply divide and conquer strategies to sub-goals when these are independent (SIMPLIFYING ASSUMPTION - otherwise ...)

Planning systems. Representation.

Design of planning systems:

- **Situation calculus**
  – based on FOL,
  – a situation variable models new states of the world
- **STRIPS – like planners**
  – STRIPS – STanford Research Institute Problem Solver
  – Restricted language as compared to situation calculus
  – Allows for more efficient planning algorithms

Situation calculus

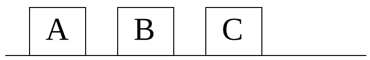
- Logic for reasoning about changes in the state of the world
- **The world is described by:**
  - Sequences of situations of the current state
  - Changes from one situation to another are caused by actions
- **The situation calculus allows us to:**
  - Describe the initial state and goal state
  - Build the KB that describes the effect of actions (operators)
  - Prove that the KB implies the goal state (and thereby allow us to extract a plan)

Situation calculus

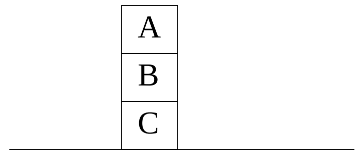
Language:

- **Variables** *s,a* – objects of type situation and action
- **Action functions** that return actions.
  - E.g. *Move(A, TABLE, B)* represents a move action
  - *Move(x,y,z)* represents an action schema
- **Two special function symbols of type situation**
  - s_0 – initial situation
  - *DO(a,s)* – denotes the situation obtained after performing an action a in situation s
- **Situation-dependent functions and relations** (also called fluents)
  - Relation: *On(x,y,s)* – object x is on object y in situation s;
  - Function: Above(x,s) – object that is above x in situation s.

Situation calculus - Blocks world example



Initial state



Goal

On(A,Table, s_0*)*
On(B,Table, s_0*)*
On(C,Table, s_0*)*
Clear(A, s_0*)*
Clear(B, s_0*)*
Clear(C, s_0*)*
Clear(Table, s_0*)*

On(A,B, s)
On(B,C, s)
On(C,Table, s)

Blocks world example - Axioms

Knowledge in the KB - Two types of axioms:

- **Effect axioms**
  - changes in situations that result from actions
- **Frame axioms**
  - things preserved from the previous situation

Blocks world - Effect axioms

Effect axioms:

Moving x from y to z. $MOVE(x, y, z)$

Effect of move changes on On relations:

$$On(x, y, s) \wedge Clear(x, s) \wedge Clear(z, s) \rightarrow On(x, z, DO(MOVE(x, y, z), s))$$

$$On(x, y, s) \wedge Clear(x, s) \wedge Clear(z, s) \rightarrow \neg On(x, y, DO(MOVE(x, y, z), s))$$

Effect of move changes on Clear relations:

$$On(x, y, s) \wedge Clear(x, s) \wedge Clear(z, s) \rightarrow Clear(y, DO(MOVE(x, y, z), s))$$

$$On(x, y, s) \wedge Clear(x, s) \wedge Clear(z, s) \wedge (z \neq Table) \\ \rightarrow \neg Clear(z, DO(MOVE(x, y, z), s))$$

Blocks world - Frame axioms

- **Frame axioms**
  - Represent things that remain unchanged after an action.

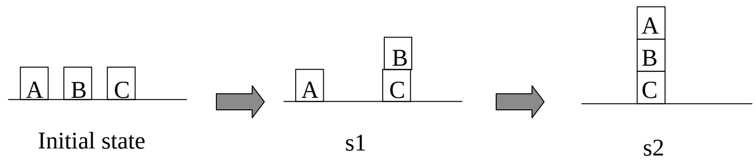
On relations:

$$On(u, v, s) \wedge (u \neq x) \wedge (v \neq y) \rightarrow On(u, v, DO(MOVE(x, y, z), s))$$

Clear relations:

$$Clear(u, s) \wedge (u \neq z) \wedge \rightarrow Clear(u, DO(MOVE(x, y, z), s))$$

Inference - Plan derivation



Action: $MOVE\,(B, Table, C)$

$s_1 = DO(MOVE\,(B, Table, C), s_0)$

$On(A, Table, s_1)$ $\quad Clear\,(A, s_1)$ $\quad Clear\,(Table, s_1)$

$On(B, C, s_1)$ $\quad Clear\,(B, s_1)$

$On(C, Table, s_1)$ $\quad \neg Clear\,(C, s_1)$

Action: $MOVE\,(A, Table, B)$

$s_2 = DO(MOVE\,(A, Table, B), s_1)$

$\quad = DO(MOVE\,(A, Table, B), DO(MOVE\,(B, Table, C), s_0))$

$On(A, B, s_2)$ $\quad Clear\,(A, s_2)$ $\quad Clear\,(Table, s_2)$

$On(B, C, s_2)$ $\quad \neg Clear\,(B, s_2)$

$On(C, Table, s_2)$ $\quad \neg Clear\,(C, s_2)$

Frame problem (general)

Frame problem

- The need to represent a large number of frame axioms
- Solution: combine positive and negative effects in one rule

$$On(u, v, DO(MOVE\,(x, y, z), s)) \Leftrightarrow (\neg((u = x) \wedge (v = y)) \wedge On(u, v, s)) \vee$$
$$\vee\, (((u = x) \wedge (v = z)) \wedge On(x, y, s) \wedge Clear\,(x, s) \wedge Clear\,(z, s))$$

Inferential frame problem:

– We still need to derive properties that remain unchanged

Other problems:

- **Qualification problem –** enumeration of all possibilities under which an action holds
- **Ramification problem –** enumeration of all inferences that follow from some facts

Planning in situation calculus

- Planning converted to theorem proving

  Goal state:

  $\exists s\ On(A,B,s) \wedge On(B,C,s) \wedge On(C,Table,s)$
- **Plan** (solution) is a byproduct of theorem proving.

- Possible inference approaches
  - inference rule approach
  - resolution
- **Problem:**
  - Large search space.
  - Proof may not lead to the best plan. Proof may not be the most efficient one.

STRIPS planner

- Restricted representation language as compared to the situation calculus
- Leads to more efficient planning algorithms:
  - State-space search with structured representations of states, actions and goals
  - Action representation avoids the frame problem

- STRIPS planning problem
  - much like a standard search problem;

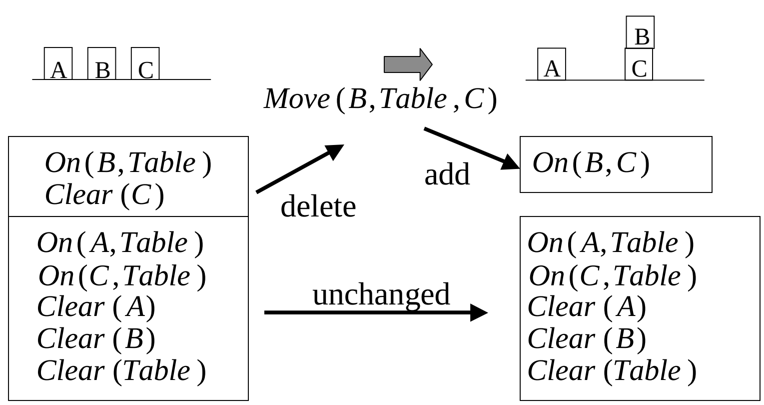
Objective: find a sequence of operators from the initial state to the goal

STRIPS planner

- **States:**
  - conjunction of literals
    On(A,B), On(B,Table), Clear(A)
    represent facts that are true at a specific point in time
- **Actions:**
  - **Action:** *Move (x,y,z)*
  - **Preconditions:** conjunctions of literals with variables
    On(x,y), Clear(x), Clear(z)
  - **Effects.** Two lists:
    - **Add list:** *On(x,z), Clear(y)*
    - **Delete list:** *On(x,y), Clear(z)*

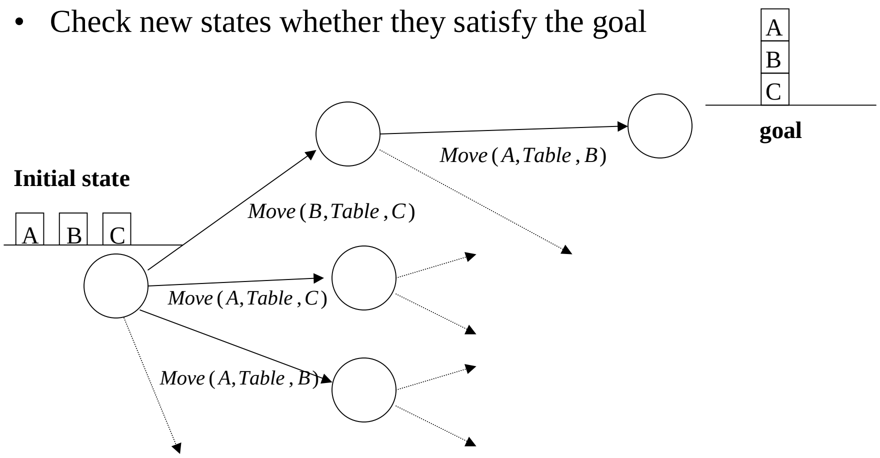
Forward search (goal progression)

- Main idea: Given a state *s*
  - Unify the preconditions of some operator *a* with *s*
  - Add and delete sentences from the add and delete list of an operator *a* from *s* to get a new state (can be repeated)



Forward search (goal progression)

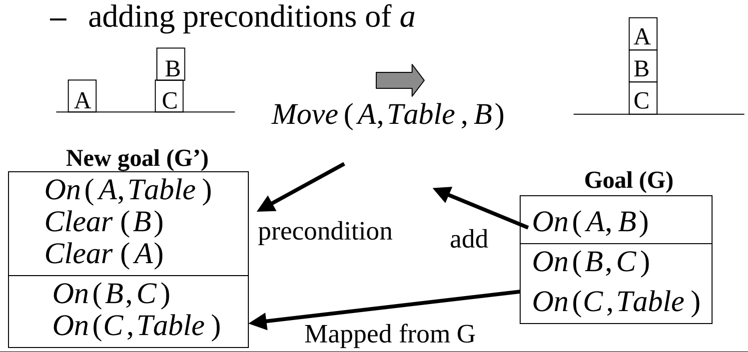
- Use operators to generate new states to search
- Check new states whether they satisfy the goal



Backward search (goal regression)

Main idea: Given a goal *G*

- Unify the addition list of some operator *a* with a subset of *G*
- If the delete list of *a* does not removes elements of *G*, then the goal regresses to a new goal *G'* that is obtained from *G* by:
  - deleting add list of *a*
  - adding preconditions of *a*



Backward search (goal regression)

- Use operators to generate new goals
- Check whether the initial state satisfies the goal