# Planning

## 1 Problem setting

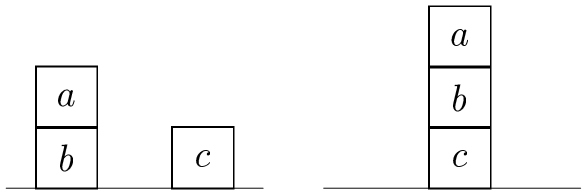
- A special case of the state space search problem, in which we use logical representation of goals and operators.
- States have structure, which allows for more efficient planning.
- Example: blocks world (Figure 1)

## 2 Situation Calculus

- Using predicate calculus to formalize *situations* and *actions* (McCarthy& Hayes, 1969).
- Actions and facts are terms, for example: $puton(A, B)$, $on(A, B)$.
- The true facts in a situation are represented by the prediate $holds$. For example: $holds(on(A, B), S)$ means that object $A$ is on object $B$ in situation $S$.
- *Situations* are terms describing the state of the world. Given a situation $s0$, other situations are obtained by applying the function $result$ to it. For example, $result(puton(a, b), s0)$ is the situation obtained by applying the action $puton(a, b)$ to the situation $s0$. More complex situations can be descrined too: $result(puton(c, a), result(puton(a, b), s0))$.
- Convenient Prolog representation: applying action $puton(A, B)$ in situation $S$ leads to situation $[puton(A, B)|S]$. Then starting with the initial situation (empty list []), the curent situation is a list of the actions that have led to it (in a reverse order).
- *Axioms* are represented by implications connecting preconditions and effects described with $holds$. For example:

  $$holds(clear(A), S) \wedge holds(clear(B), S) \rightarrow holds(on(A, B), result(puton(A, B), S))$$

- It is important to represent also the facts that remain true in the new situation, i.e. what does not change after applying an action. This is done through the so



| Initial state | Goal state |
|---|---|
| $on(a, b)$ | $on(a, b)$ |
| $on(b, table)$ | $on(b, c)$ |
| $on(c, table)$ | $on(c, table)$ |
| $clear(a)$ | $clear(a)$ |
| $clear(c)$ | |

Plan: $puton(a, table), puton(b, c), puton(a, b)$

Figure 1: Planning in the blocks world

called *frame axioms.* For example:

$$holds(clear(X), S) \wedge \neg eq(X, B) \rightarrow holds(clear(X), result(puton(A, B), S))$$

- *Frame problem:* a frame axiom is needed for each fact used in the representation.
- Search is not efficient (due to the frame problem).

## 3 STRIPS approach to planning

- An attempt to solve the frame problem (Fikes&Nilsson, 1971).
- Each action is represented by three components:
  - *Precondition*: a formula that must be true in order to perform the action.
  - *Add list*: formulas that become true after performing the action.
  - *Delete list*: formulas that become false after performing the action.
- Example: $puton(X, Y)$
  - Precondition: $clear(X) \wedge clear(Y) \wedge on(X, Z)$
  - Add list: $\{on(X, Y), clear(Z)\}$
  - Delete list: $\{clear(Y), on(X, Z)\}$
- *Basic assumption*: each formula that has been true before the action and does not belong to the delete list is true after the action.
- *Independent goals*: goals that can be reached independently in any order. For example:

- Initial sate: $\{on(a, table), on(b, table), on(c, table), on(d, table), clear(a), clear(b), clear(c), clear(d)\}$
- Goals: $\{on(a, b), on(c, d)\}$.

- *Dependent goals*: depend on the order of execution. For example:
  - Initial state: $\{on(a, table), on(b, table), on(c, table), clear(a), clear(b), clear(c)\}$
  - Goals: $\{on(a, b), on(b, c)\}$.

- *Sussman anomaly*: Initial state = $\{on(a, table), on(b, table), on(c, a), clear(b), clear(c)\}$; Goal = $\{on(a, b), on(b, c)\}$. Noninterleaved planners (plans for subgoals are concatenated) cannot solve it.

- STRIPS algorithm (+ and − denote set operations):

```
strips_plan(State,Goals,NewState,Plan) :-
% [Goal|Rest_of_unsatisfied_goals] = Goals - State
  planning_rule(Action,Precondition,Add_list,Delete_list),
  member(Goal,Add_list),
  strips_plan(State,Precondition,State1,Plan1),
% State2 = State1 + Add_list - Delete_list
  strips_plan(State2,Rest_of_unsatisfied_goals,NewState,Plan2),
% Plan = Plan1 + [Action|Plan2]
strips_plan(State,_,State,[]).
```

- Satisfying dependent goals: Applying the STRIPS algorithm multiple times.

- Optimality (minimal number of actions) and efficiency of planning:
  - Reordering goals.
  - Choose a rule with `Add_list` that includes other goals. For example: Initial state = $\{on(a, table), on(b, table), on(c, a), clear(b), clear(c)\}$, Goal = $\{on(a, c), on(c, b)\}$).
  - Partial-Order Planning (POP): searching the space of partial plans (Figure 2):
    * Plan1 < Plan3 (action of Plan1 provides precondition of plan3).
    * Plan1 < Plan2 (action of Plan2 destroys precondition of plan1).
    * Plan2 < Plan3 (action of Plan3 destroys precondition of plan2).
    * Then the optimal plan is: {Plan1, Plan2, Plan3 }.

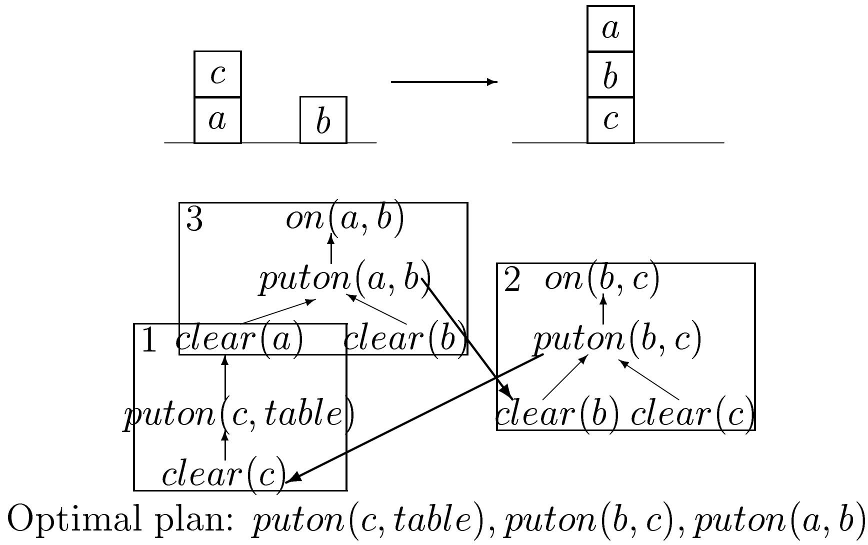



Figure 2: Partial-Order Planning