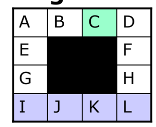
6.825 Techniques in Artificial Intelligence

Planning

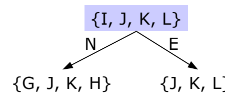
- Planning vs problem solving
- Situation calculus
- Plan-space planning

Planning as Problem Solving

- Planning:
  - Start state (S)
  - Goal state (G)
  - Set of actions
- Can be cast as "problem-solving" problem
- But, what if initial state is not known exactly? E.g. start in bottom row in 4x4 world, with goal being C.
- Do search over "sets" of underlying states to find a set of actions that will reach C for any starting state.

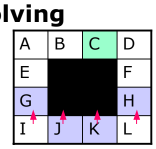


Actions: N,S,E,W

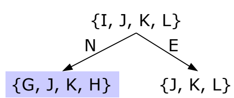


Planning as Problem Solving

- Planning:
  - Start state (S)
  - Goal state (G)
  - Set of actions
- Can be cast as "problem-solving" problem
- But, what if initial state is not known exactly? E.g. start in bottom row in 4x4 world, with goal being C.
- Do search over "sets" of underlying states.

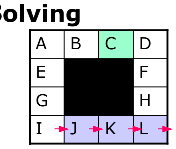


Actions: N,S,E,W

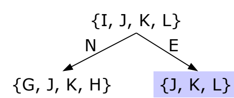


Planning as Problem Solving

- Planning:
  - Start state (S)
  - Goal state (G)
  - Set of actions
- Can be cast as "problem-solving" problem
- But, what if initial state is not known exactly? E.g. start in bottom row in 4x4 world, with goal being C.
- Do search over "sets" of underlying (atomic) states.



Actions: N,S,E,W



Planning as Logic

- The problem solving formulation in terms of sets of atomic states is incredibly inefficient because of the exponential blowup in the number of sets of atomic states.
- Logic provides us with a way of describing sets of states.
- Can we formulate the planning problem using logical descriptions of the relevant sets of states?
- This is a classic approach to planning: situation calculus, use the mechanism of FOL to do planning.
- Describe states and actions in FOL and use theorem proving to find a plan.

Situation Calculus

- Reify situations: [reify = name, treat them as objects] and use them as predicate arguments.
  - At(Robot, Room6, S_9) where S_9 refers to a particular situation
- Result function: a function that describes the new situation resulting from taking an action in another situation.
  - Result(MoveNorth, S_1) = S_6
- Effect Axioms: what is the effect of taking an action in the world
  - $\forall$ x.s. Present(x,s) $\wedge$ Portable(x) $\rightarrow$ Holding(x, Result(Grab, s))
  - $\forall$ x.s. $\neg$ Holding(x, Result(Drop, s))
- Frame Axioms: what doesn't change
  - $\forall$ x.s. color(x,s) = color(x, Result(Grab, s))
  - Can be included in Effect axioms

Planning in Situation Calculus

- Use theorem proving to find a plan
- Goal state: $\exists$ s. At(Home, s) $\wedge$ Holding(Gold, s)
- Initial state: At(Home, s_0) $\wedge$ $\neg$ Holding(Gold, s_0) $\wedge$ Holding(Rope, s_0) …
- Plan: Result(North, Result(Grab, Result(South, s_0)))
  - A situation that satisfies the requirements
  - We can read out of the construction of that situation what the actions should be.
  - First, move South, then Grab and then move North.

Special Properties of Planning

- Reducing specific planning problem to general problem of theorem proving is not efficient.
- We will be build a more specialized approach that exploits special properties of planning problems.
  - Connect action descriptions and state descriptions [focus searching]
  - Add actions to a plan in any order
  - Sub-problem independence
  - Restrict language for describing goals, states and actions

STRIPS representations

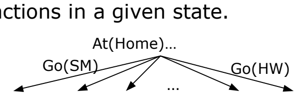
- States: conjunctions of ground literals
  - In(robot, r_3) $\wedge$ Closed($door_6$) $\wedge$ …
- Goals: conjunctions of literals
  - (implicit $\exists$ r) In(Robot, r) $\wedge$ In(Charger, r)
- Actions (operators)
  - Name (implicit $\forall$): Go(here, there)
  - Preconditions: conjunction of literals
    – At(here) $\wedge$ path(here, there)
  - Effects: conjunctions of literals [also known as post-conditions, add-list, delete-list]
    – At(there) $\wedge$ $\neg$ At(here)
  - Assumes no inference in relating predicates (only equality)

Strips Example

- Action
  - Buy(x, store)
    – Pre: At(store), Sells(store, x)
    – Eff: Have(x)
  - Go(x, y)
    – Pre: At(x)
    – Eff: At(y), $\neg$At(x)
- Goal
  - Have(Milk) $\wedge$ Have(Banana) $\wedge$ Have(Drill)
- Start
  - At(Home) $\wedge$ Sells(SM, Milk) $\wedge$ Sells(SM, Banana) $\wedge$ Sells(HW, Drill)

Planning Algorithms

- Progression planners: consider the effect of all possible actions in a given state.



- Regression planners: to achieve a goal, what must have been true in previous state.

Have(M) $\wedge$ Have(B) $\wedge$ Have(D)

Buy(M,store)

At(store) $\wedge$ Sells(store,M) $\wedge$ Have(B) $\wedge$ Have(D)

- Both have problem of lack of direction – what action or goal to pursue next.

Plan-Space Search

- Situation space – both progressive and regressive planners plan in space of situations
- Plan space – start with null plan and add steps to plan until it achieves the goal
  - Decouples planning order from execution order
  - Least-commitment
    – First think of what actions before thinking about what order to do the actions
  - Means-ends analysis
    – Try to match the available means to the current ends

Partially Ordered Plan

- Set of steps (instance of an operator)
- Set of ordering constraints $S_i < S_j$
- Set of variable binding constraints v=x
  - v is a variable in a step; x is a constant or another variable
- Set of causal links $S_i \square_c S_j$
  - Step i achieves precondition c for step j

Initial Plan

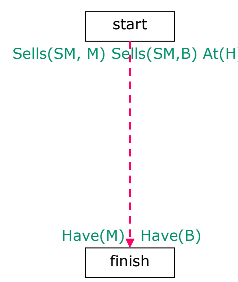
- Steps: {start, finish}
- Ordering: {start < finish}

- start
  - Pre: none
  - Effects: start conditions
- finish
  - Pre: goal conditions
  - Effects: none

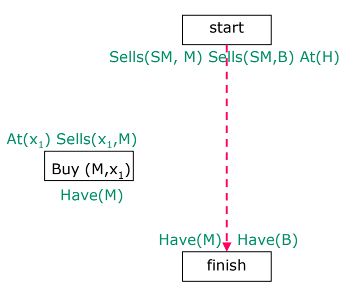
Plan Completeness

- A plan is complete iff every precondition of every step is achieved by some other step.
- $S_i \square_c S_j$ ("step I achieves c for step j") iff
  - $S_i < S_j$
  - $c \in \text{effects}(S_i)$
  - $\neg \exists S_k.\ \neg c \in \text{effects}(S_k)$ and $S_i < S_k < S_j$ is consistent with the ordering constraints
- A plan is consistent iff the ordering constraints are consistent and the variable binding constraints are consistent.

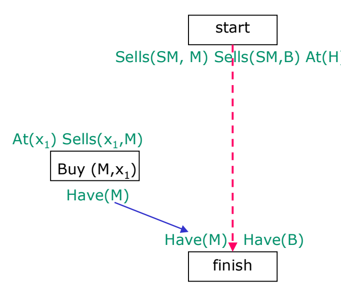
PO Plan Example



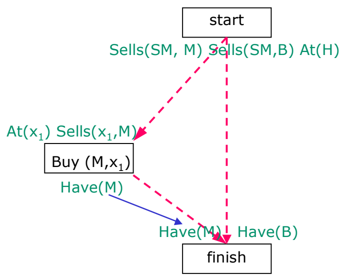
PO Plan Example



PO Plan Example

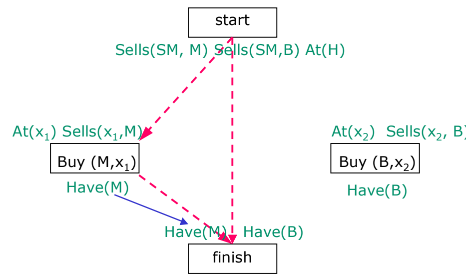


PO Plan Example



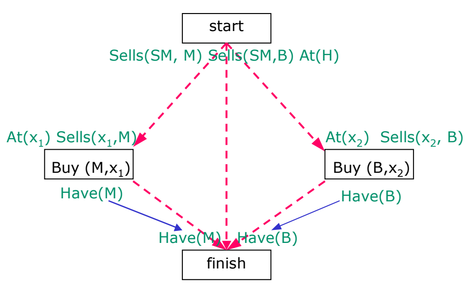
Lecture 10 • 19

PO Plan Example



Lecture 10 • 20

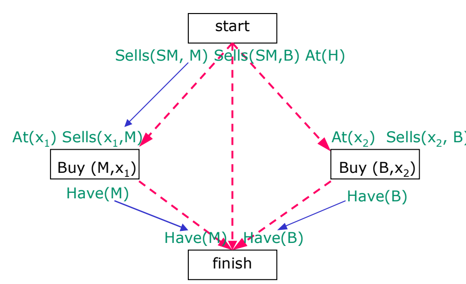
PO Plan Example



Lecture 10 • 21

PO Plan Example

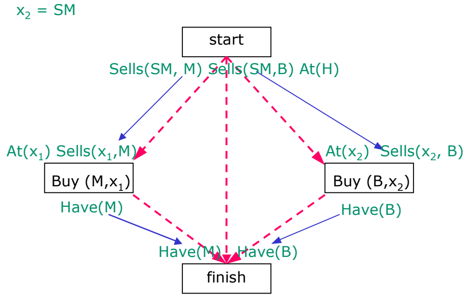
x_1 = SM



Lecture 10 • 22

PO Plan Example

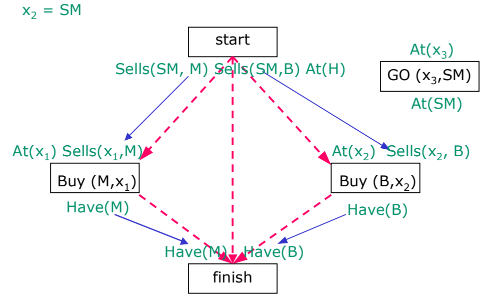
x_1 = SM
x_2 = SM



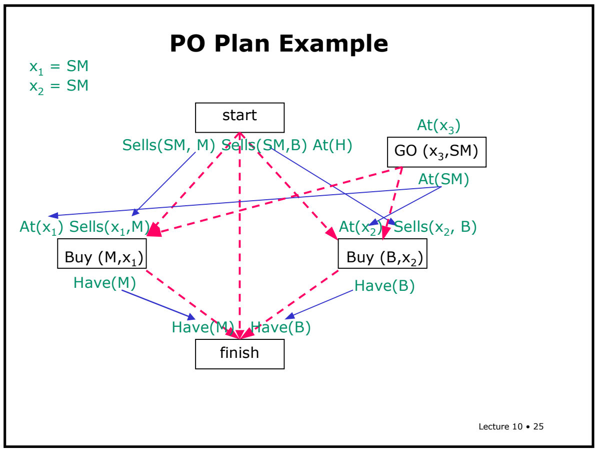
Lecture 10 • 23

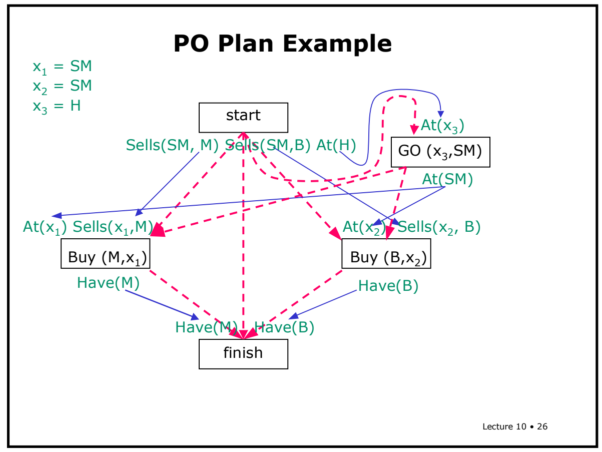
PO Plan Example

x_1 = SM
x_2 = SM



Lecture 10 • 24


PO Plan Example
x1 = SM
x2 = SM
start
Sells(SM, M) Sells(SM,B) At(H)
At(x3)
GO (x3,SM)
At(SM)
At(x1) Sells(x1,M)
At(x2) Sells(x2, B)
Buy (M,x1)
Buy (B,x2)
Have(M)
Have(B)
Have(M) Have(B)
finish
Lecture 10 • 25


PO Plan Example
x1 = SM
x2 = SM
x3 = H
start
Sells(SM, M) Sells(SM,B) At(H)
At(x3)
GO (x3,SM)
At(SM)
At(x1) Sells(x1,M)
At(x2) Sells(x2, B)
Buy (M,x1)
Buy (B,x2)
Have(M)
Have(B)
Have(M) Have(B)
finish
Lecture 10 • 26