Advance AI
Nishank Singhal
Chapter 10 Exercise

Q 10.3) The monkey-and-bananas problem is faced by a monkey in a laboratory with some bananas hanging out of reach from the ceiling. A box is available that will enable the monkey to reach the bananas if he climbs on it. Initially, the monkey is at A, the bananas at B, and the box at C. The monkey and box have height Low, but if the monkey climbs onto the box he will have height High, the same as the bananas. The actions available to the monkey include Go from one place to another, Push an object from one place to another, ClimbUp onto or ClimbDown from an object, and Grasp or Ungrasp an object. The result of a Grasp is that the monkey holds the object if the monkey and object are in the same place at the same height.

a. Write down the initial state description.

b. Write the six action schemas.

c. Suppose the monkey wants to fool the scientists, who are off to tea, by grabbing the bananas, but leaving the box in its original place. Write this as a general goal (i.e., not assuming that the box is necessarily at C) in the language of situation calculus. Can this goal be solved by a classical planning system?

d. Your schema for pushing is probably incorrect, because if the object is too heavy, its position will remain the same when the Push schema is applied. Fix your action schema to account for heavy objects.

Solution:

A) Write down the initial state description?
   ➔ Init(At(Monkey,A)∧At(Bananas,B)∧At(Box,C)∧Height(Monkey,Low)∧Height(Box,Low)∧ Height(Bananas,High)∧ P ushable(Box) ∧ Climbable(Box))
B) Write the six action schemas?
   ➔ Six Action Schemas as follows:
      - Go from one place to another-
        - Action(ACTION:Go(x, y),
        - PRECOND:At(Monkey, x),
        - EFFECT:At(Monkey,y)∧¬(At(Monkey,x)))
      - Push object from one place to another-
        - Action(ACTION:Push(b,x,y),
        - PRECOND:At(Monkey,x)∧Pushable(b),
        - EFFECT:At(b,y)∧At(Monkey,y)∧¬At(b,x)∧¬At(Monkey,x))
      - Climb Up onto an object-
        - Action(ACTION:ClimbUp(b),
        - PRECOND : At(Monkey,x)∧At(b,x)∧Climbable(b),
        - EFFECT : On(Monkey,b)∧¬Height(Monkey,Low) ∧Height(Monkey, High)
      - Group an Object-
        - Action(ACTION:Grasp(b),

        - PRECOND:Height(Monkey,h)∧Height(b,h) ∧ At(M onkey, x) ∧ At(b, x),
        - EFFECT:Have(Monkey, b))
    - Climb Down from an object:
        - Action(ACTION:ClimbDown(b),
        - PRECOND:On(Monkey, b) ∧ Height(Monkey, High),
        - EFFECT:¬On(Monkey,b)∧¬Height(Monkey,High) ∧Height(Monkey,Low))
    - Ungrasp an object:
        - Action(ACTION:UnGrasp(b),
        - PRECOND:Have(Monkey, b),
        - EFFECT:¬Have(Monkey, b))

C) Suppose the monkey wants to fool the scientists, who are off to tea, by grabbing the bananas, but leaving the box in its original place. Write this as a general goal (i.e., not assuming that the box is necessarily at C) in the language of situation calculus. Can this goal be solved by a classical planning system?

➔ The general goal state in situation calculus is given below:-
- Have(Monkey,Bananas,s)∧(∃x At(Box,x,s_0)∧At(Box,x,s))
- In STRIPS, we can only talk about the goal state; there is no way of representing the fact that there must be some relation (such as equality of location of an object) between two states within the plan. So there is no way to represent this goal.

D) Your schema for pushing is probably incorrect, because if the object is too heavy, its position will remain the same when the Push schema is applied. Fix your action schema to account for heavy objects.

➔ Actually, solution did include the Pushable precondition in the solution above (A)(B)

Q 10.13) We contrasted forward and backward state-space searchers with partial-order planners, saying that the latter is a plan-space searcher. Explain how forward and backward state- space search can also be considered plan-space searchers, and say what the plan refinement operators are.

Solution:

Planning with forward state space search is similar to progression planning. A forward state-space planner maintains a partial plan that is a strict linear sequence of actions; the plan refinement operator is to add an applicable action to the end of the sequence, updating literals according to the action's effects.

Planning with backward search considered only relevant actions. A backward state-space planner maintains a partial plan that is a reversed sequence of actions; the refinement operator is to add an action to the beginning of the sequence as long as the action's effects are compatible with the state at the beginning of the sequence.