National University of Singapore
School of Computing

Semester 2, AY2023-24

CS4246/CS5446 AI Planning and Decision Making

Tutorial Week 3: Classical Planning

Guidelines

- You can discuss the content of the questions with your classmates.
- However, everyone should work on and be ready to present ALL the solutions.
- Your attendance is marked in the tutorial and participation noted to award class participation marks.

Problem 1 Planning Models

You have been appointed as the lead engineer in the development team of SG Smart Taxi, an autonomous taxi service. Define the characteristics of the task environment in which the taxi has to be deployed. Support your answer with sufficient reasoning behind your choices.

a) Comment on the observability aspect of the environment.
Solution:
The environment would be **partially observable**. Though the taxi might have a map of the overall route, it would not be able to observe many attributes such as traffic conditions, pedestrians, etc.

b) Is the environment a single-agents, collaborative multi-agent or competitive multi-agent environment? Justify your answer.
Solution:
The environment would comprise of **multiple-agents** as components like incoming cars can influence the performance of our smart taxi. In general, these agents can be assumed to be collaborative as incidents such as accidents hurts all. However, there might be traces of competitions between agents. For example, we might want to compete with a fellow company's agent over performance factors like total users transferred per day

c) Will the environment be deterministic or stochastic? Give an example.
Solution:
The environment would be **stochastic**. For example, transition from a state using an action might result in a state where a new pedestrian comes onto the road. This element of chance in the resulting state makes the environment stochastic.

d) Would you model the environment to be episodic or sequential? Explain your choice.
Solution:
The environment should be modelled as **sequential**. This is particularly important for the agent to estimate utilities properly. For example, if the taxi is speeding towards an obstacle, its percept sequence containing temporal information would allow to detect the need to de-accelerate early.

Problem 2 STRIPS Planning Models

[RN3e 10.4] 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.
Solution:
At(Monkey, A) Λ At(Bananas,B) Λ At(Box, C) Λ Height(Monkey, Low) Λ Height(Box, Low) Λ Height(Bananas, High) Λ Pushable(Box) Λ Climbable(Box) Λ Graspable(Bananas)

Note: Colors have no significance here.

b) Write the six action schemas.
Solution:

Action(Action Go(x,y),
  Precond: At(Monkey, x),
  Effect: At(Monkey, y) Λ ¬ (At(Monkey,x)))

Action(Action: Push(b,x,y),
  Precond: At(Monkey, x) Λ At(b, x) Λ Pushable(b) Λ Height(Monkey, Low),
  Effect: At(b, y) Λ At(Monkey, y) Λ ¬At(b, x) Λ ¬At(Monkey, x))

Action(Action: ClimbUp(b),
  Precond: At(Monkey,x) Λ At(b, x) Λ Climbable(b) Λ Height(Monkey, Low)
  Effect: On(Monkey, b) Λ Height(Monkey, High) Λ ¬Height(Monkey, Low))

Action(Action: Grasp(b,x,h),
  Precond: Height(Monkey, h) Λ Height(b, h) Λ At(Monkey, x) Λ At(b, x) Λ Graspable(b),
  Effect: Have(Monkey, b) Λ ¬At(b, x) Λ ¬Height(b, h))

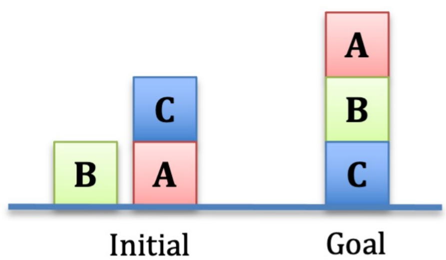
Action(Action: ClimbDown(b),
    Precond: On(Monkey, b),
    Effect: ¬On(Monkey, b) Λ ¬Height(Monkey, High) Λ Height(Monkey, Low))

Action(Action:UnGrasp(b,x,h),
    Precond: Have(Monkey, b) Λ At(Monkey, x) Λ Height(Monkey, h),
    Effect: ¬ Have(Monkey, b) Λ At(b, x) Λ Height(b, h))

Note: This is not the only correct solution, other checks can be added as long as **they make sense and are reasonable**.

c) Suppose the monkey wants to fool the scientists, who are off to coffee, by grabbing the bananas, but leaving the box in its original place. Can this general goal (i.e., not assuming that the box is necessarily at *C*) be solved by a STRIPS-style system?
Solution:
In PDDL, which follows the STRIPS convention, 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.

Problem 3 Sussman Anomaly



[RN3e 10.7] The figure shows a blocks-world problem known as the Sussman anomaly. The problem was considered anomalous because the noninterleaved planners of the early 1970s could not solve it.

a) Write a definition of the problem.

Solution:
The initial state is:
On(B,Table) Λ On(C,A) Λ On(A, Table) Λ Clear(B) Λ Clear(C)
The goal is:
On(A, B) Λ On(B,C)

Action(Action Move(obj, from, to),
  Precond: Clear(obj) Λ Clear(to),
  Effect: On(obj,to) Λ Clear(from) Λ $\neg$ Clear(to))

Action(Action: MovetoTable(obj,from),
  Precond: Clear(obj),
  Effect: On(obj,Table) Λ Clear(from)

b) Solve the problem, either by hand or with a planning program.
Solution:
MoveToTable(C, A)
Move(B, Table, C)
Move(A, Table, B)

c) A noninterleaved planner is a planner that, when given two subgoals *G1* and *G2*, produces either a plan for *G1* concatenated with a plan for *G2*, or vice-versa. Explain why a noninterleaved planner cannot solve this problem.
Solution:
To solve the problem, we need to consider all the subgoals together, and allow for backtracking (or undoing of subgoals) to get the complete plan. An interleaved planner cannot do that. While it solves *G1*, it needs to undo it in order to achieve *G2*. Thus, the noninterleave planner cannot solve that problem.

More reference: https://en.wikipedia.org/wiki/Sussman_anomaly