10-6-03 CSE571 notes

Daniel Bryce
Department of Computer Science and Engineering
Arizona State University, Tempe AZ 85287-5406
dan.bryce@asu.edu

Chapter 5: Actions and Planning

- Example: ”load causes loaded” and ”shoot causes ¬ alive if loaded”.
- Write this as an AnsProlog program.
  - Initial State:
    initially loaded. == loaded(0).
    initially alive. == alive(0).
  - Actions:
    load causes load. == loaded(X+1) ← load(X).
    shoot causes ¬ alive if loaded. == ¬ alive(X+1) ← shoot(X), loaded(X).
- Write a Prediction Query
  - Query: ”What happens after shoot in the initial state?” == shoot(0).
  - Derived Answer: ¬ alive(1).
  - The answer is found by using the second action rule because ¬ alive(1) ← shoot(0), loaded(0).
- We’re missing facts that haven’t changed – Need Frame Axioms
  - ”What happens after shoot in the initial state?” == shoot(0).
  - Right Answer: ¬ alive(1), loaded(1).
  - Why loaded(1)? – loaded wan’t changed by shoot.
  - How do we get this? – Add these rules:
    loaded(X+1) ← loaded(X), not ¬ loaded(X+1).
    ¬ loaded(X+1) ← ¬ loaded(X), not loaded(X+1).
  - We can also state this for alive, like so:
    alive(X+1) ← alive(X), not ¬ alive(X+1).
    ¬ alive(X+1) ← ¬ alive(X), not alive(X+1).
- Types of Reasoning
  - Prediction – What happens after I do this set of actions?
  - Hypothetical Reasoning – What happens if I were to do this set of actions?
  - Counterfactual Reasoning – What happens if I did something else besides this set of actions?
  - Explaining Observations – What happened? Do my observations support this fact at this time? For example, we observe ¬ alive(1) and want to know what happened. We must make a constraint ⊥← not ¬ alive(1) and evaluate the program to see what happened. We can also see that for loaded(0), the following holds 1{loaded(0), ¬ loaded(0)}1 and it must be that loaded(0) held for ¬ alive(1) to hold because alive(0) holds.

- Planning from initial state – How is a state achieved from another state?
  How is the goal state $\neg$ alive(X) achieved from the initial state loaded(0), alive(0)?
  Need to add a rules to enforce a solution:
  goal $\leftarrow \neg$ alive(X), time(X).
  $\perp \leftarrow$ not goal.
  time(1).
  time(2).
  $\vdots$
  time(GOAL_HORIZON).
  Also need rules to serialize actions:
  0{loaded(X), shoot(X)}1 $\leftarrow$ time(X).
- Planning from the current state – During execution, the plan dynamically becomes un-executable due to new observations. How is goal state achieved from the current state?
- Diagnosis – I can't find an explanation for some fact, why is there no explanation?