James Villemarette

CISC 481 Homework 1

Due: March 15, 2022

Extension: March 20, 2022

**Problem 1**

There is no exercise 2.9 in our textbook, I assume this is problem is referencing a different textbook.

A simple reflex is to quote 2.4.2, “the simplest kind of agent […] select[ing] actions on the basis of the current precept, ignoring […] percept history” (Russel & Norvig 121). The example given in the textbook looks like

Function reflexVacuumAgent(location, status)

If status == dirty then return Suck

Else if location == A then return Right

Else if location == B then return Left

The pseudocode for the thermostat would look like

Function reflexThermostat(setting, temperature)

If temperature < setting – 3 then TurnOnFurnace

Else if temperature > setting + 3 then TurnOffFurnance

Else DoNothing

A model-based reflex agent would be stateful, moving between states based on what it *can* observe to keep track of things it can’t observe, or things it needs to remember.

A goal-based agent is the “sub class” of a model-based agent, where it not only has states, but goals and some element of path finding to that goal.

A model-based reflex and goal-based agent can both probably solve this problem.

But I argue it is best to solve this problem with the simplest option that we have available to us that completely solves it—a simple reflex agent. It might be nice to have a goal-based agent that can possibly “learn” from the ebbs and flows of the environment, maybe the temperature outside, and any possible energy saving goal. But as the problem is described, a simple reflex agent nicely solves the problem. My pseudocode above even looks almost example like the pseudocode provided in the textbook.

**Problem 2**

I am referencing Chapter 3.1.1 and 3.2.1 as an example for solving this problem.

*Define the problems*

* NxN grid of squares
  + States = N \* N \* 3
    - For the N \* N grid, there is a state for at least each square
    - There are 3 states for every square:
      * Unpainted,
      * painted,
      * and bottomless pit
    - Ex: State N[0, 0]unpainted; N[0,0]painted; N[0,0]bottomlessPit; …
  + Initial State = Any state that is not
  + Actions = { PaintSquare, MoveUp, MoveUpRight, MoveRight, MoveDownRight, MoveDown, MoveDownLeft, MoveLeft, MoveUpLeft }
  + Transition Model =
  + Goal States = When all non-bottomlessPit tiles are painted
  + Actions cost = Each action cost 1
* Container ship in port
  + States = {}
  + Initial State =
  + Actions = {}
  + Transition Model = {}
  + Goal States = {}
  + Actions cost = Each action cost 1

**Problem 3**

*Define the problem*

* States
  + 1: PetGoldenFox, PetBlueHen, and BushelOfBlueCorn are all at SouthCampus
  + 2: PetGoldenFox at LairdCampus; PetBlueHen and BushelOfBlueCorn at SouthCampus
  + 3: PetGoldenFox and PetBlueHen at LairdCampus; BushelOfBlueCorn at SouthCampus
  + 4: PetGoldenFox, PetBlueHen, and BushelOfBlueCorn are all at LairdCampus
* Initial State
  + State 1 (everyone at south campus)
* Actions
  + 1: Bring PetGoldenFox to LairdCampus
  + 2: Bring PetBlueHen to LairdCampus
  + 3: Bring BushelOfBlueCorn to LairdCampus
* Transition Model
* Goal States
  + State 4 (all pets at Laird Campus)
* Action cost
  + Each action costs 1

*Which uninformed search strategy would I recommend, and why*: …

*Trace search tree produced by our search method, assuming a graph search implementation (prune repeated states):*

…