CS 533: Intelligent Agents and Decision Making

Homework #1 Report

Michael Lam

Due: April 6, 2016

This assignment is about modeling (simplified) Pac-Man as a Markov Decision Process (MDP).

An MDP contains a state space S, action space A, transition function T and reward function R. An MDP for Pac-Man will be specified below. It is assumed that the game has a simpler environment where the ghost movements are less complex.

1 Actions

The set of actions correspond to the joystick controls:

$$A = \{UP, RIGHT, DOWN, LEFT, NO\text{-}OP\} \tag{1}$$

Where *NO-OP* denotes no operation or "do nothing".

2 States

The set of states S will be defined as a tuple containing:

- *board_config*
- *current_heading*
- *pending_heading*
- *ghosts_current_heading*
- *ghosts_is_edible*
- *ghost_edible_timer*

where

- *board_config* is the current board configuration. It is an $H \times W$ array, where each element (a.k.a. cell) takes on the following possible values:
  - 0 if the cell is a wall
  - 1 if the cell is clear

  - 2 if the cell contains a small dot
  - 3 if the cell contains a large dot
  - 4 if the cell contains PacMan
  - 5 if the cell contains ghost 1
  - 6 if the cell contains ghost 2
  - 7 if the cell contains ghost 3
  - 8 if the cell contains ghost 4
  - ... if there are more ghosts

  Assume that elements can overlap each other (i.e. multiple ghosts can occupy one cell like in the beginning) and one can keep enumerating to represent states for cells containing multiple elements.

  Bonus point items in the real game such as the cherry will be ignored since it is assumed that there are no other points possible other than eating a dot, eating a ghost and advancing to the next level.

- *current_heading* is the current heading of PacMan. The possible values are *FACING-UP*, *FACING-RIGHT*, *FACING-DOWN*, *FACING-LEFT*.
- *pending_heading* is the next heading of PacMan. The possible values are *FACING-UP*, *FACING-RIGHT*, *FACING-DOWN*, *FACING-LEFT*.

  This addresses a subtlety in the PacMan controls. If PacMan is currently heading in a certain direction and the user changes direction, PacMan will try to head in the new direction. If successful, then it will change. If unsuccessful, then it will wait until it can, assuming the user does nothing until then. This is addressed more in the transition function.
- *ghosts_current_headings* holds the current headings of the ghosts. It is a vector of K elements assuming K ghosts, each element i holding the current heading of the ith ghost.

  The possible values are *FACING-UP*, *FACING-RIGHT*, *FACING-DOWN*, *FACING-LEFT*.
- *ghost_edible_timer* $\in \mathbb{Z}^{\geq 0}$ is a global timer until ghosts are no longer edible. This is 0 when ghosts are not edible and this is a positive integer when (some, possibly none) ghosts are edible, where the positive integer indicates the remaining time.
- *ghosts_is_edible* is a vector of K boolean elements, indicating whether the ith ghost is edible.

There might be some other subtleties of the game. For instance, perhaps one needs to keep track of the velocities of the ghosts and a timer of the game, since eating a large dot might slow down the ghosts and the ghosts might become faster as the game timer continually increases. However, let us assume we have covered the basics of the game.

3 Transition Function

The transition function specifies the probability of going to state $s' \in S$ after taking action $a \in A$ in state $s \in S$. Since the space is complex, the transition function will be decomposed into some rules:

3.1 Initial State

The beginning of the game can be represented as a state. The *board_config* is how the walls, dots, etc. in the level are laid out in the beginning. Assume PacMan starts somewhere and all the ghosts start in the center. Assume PacMan begins facing in an arbitrary direction $d \in$ *current_heading*. The *pending_heading* is also set to d because the joystick has not moved yet. Finally, *ghost_edible_timer* is set to 0 since PacMan has not eaten a large dot yet. This also means that *ghosts_is_edible* is initialized so that all elements are set to $FALSE$.

The only thing stochastic is that each ghost is initialized to a random direction, which is stored in *ghosts_current_headings*. This can also be represented as several states following immediately from the initial state regardless of whatever action. The transition probabilities would correspond to the probabilities of the ghosts initialized to certain directions.

3.2 Updates

The following updates happen after any kind of action is taken, whether the joystick moves or the joystick is not doing anything.

- If PacMan moves off the boundary of the grid (i.e. if there's no wall at the boundary), it will wrap around to the other side of the grid.
- When PacMan goes over a small dot, PacMan consumes it and so the dot disappears. (In other words, the cell at that point will no longer be 2.)
- When PacMan goes over a large dot, PacMan consumes it and so the dot disappears. (In other words, the cell at that point will no longer be 3.) The *ghost_edible_timer* is also set to τ, the time until the ghosts are no longer edible. Also, the *ghosts_is_edible* vector is set so all elements are $TRUE$. The *ghost_edible_timer* > 0 will decrement until *ghost_edible_timer* $= 0$, at which point the ghosts are no longer edible (and *ghosts_is_edible* is set to all $FALSE$ at that point).
- When there are no more little dots and large dots, PacMan wins the level. This state gets a reward (see reward function in the next section).
- When PacMan collides with a ghost and *ghosts_is_edible* is set to $FALSE$ for that particular ghost, PacMan dies.

- When PacMan collides with a ghost and *ghosts_is_edible* indicates that the ghost is edible, PacMan consumes the ghost. The ghost will travel back to the center and become inedible by setting the corresponding element in *ghosts_is_edible* to $TRUE$.

- Every ghost will continue moving in its current direction as indicated in *ghosts_current_heading*. When a ghost reaches a wall, it will randomly pick a valid direction (so it no longer runs into a wall), update the corresponding element in *ghosts_current_heading* and then continue moving in that new direction. These were given assumptions and simplifications of the game. The positions of ghosts are encoded in *board_config*.

3.3 No-Op Action Updates

Assume the joystick does not move. The following updates also happen in addition to the regular updates:

- PacMan moves as follows. In the current cell, it will attempt to move in the direction of *pending_heading*. If it can, it will move in that direction and then set *current_heading* to *pending_heading*. If it cannot, it will attempt to move in the direction of *current_heading*. If it can, it will move in the direction of *current_heading*. If it cannot, PacMan will simply not move until there is a joystick update. All of this addresses the subtlety in the joystick controls: one can specify the next move ahead of time. The position of PacMan is encoded in *board_config* with a 4 in the cell.

3.4 Joystick Updates

Assume the joystick is pressed (up, right, down or left). The following updates also happen in addition to the regular updates:

- If Pacman can proceed in the new direction, it will update *current_heading* and *pending_heading* to the new direction and start moving in the new direction.

- If Pacman cannot proceed in the new direction, it will update *pending_heading* to the new direction. Pacman will continue moving in the direction of *current_heading* until it can move in the *pending_heading* direction or the joystick is changed or it's stopped because of a wall. This is part of the subtlety of the joystick controls.

4 Reward Function

The reward function is as follows: $R = 1$ if the board contains no dots (small or big) and $R = 0$ otherwise. When the board contains no dots, that means the level is cleared and PacMan survived, ready for the next level. The goal is to keep advancing levels to accumulate more points. Thus designing the reward function like this encourages PacMan to advance levels.