Mini-Project 2

Michael Anderson, Tyler McClung

February 18, 2011

CS533

Prof. Fern

1 Set up

For our environment, we decided to use the state/action examples given. We used a (Location, Occupied, Parked) tuple as described in the assignment, with ten parking spots in each of row 3, giving us a total of 2 * 10 * 2 * 2 = 80 states. We also programmed the actions PARK, DRIVE, and EXIT.

We created two separate reward functions, one for a "normal driver" and one for an "impatient driver". Our normal driver took a penalty of -1 for each unparked state, while our impatient driver took -2. Also each row away from the store was 5 less valuable for the normal driver, 10 less for the impatient driver. The normal driver got a -200 for parking in a handicap spot, the impatient driver -100. Finally, both got -1000 for a collision.

Instead of using a discounting factor, we used a finite horizon of 200 actions. Also, all of our data was averaged over 1000 simulations.

2 Performance of hard-coded policies

We implemented the Random and Drive players as described in the assignment. For our custom hard-coded policies we created a Simple Drive Player that generally parks and drives randomly, but always parks when one of the relatively close spots in rows 2-5 becomes available. We also created a Simple Random Player which simply avoids handicap spots but otherwise acts randomly.

The average score of each agent was:
Random Player: -335.8
Drive Player: 11.6
Simple Random Player: -439.3
Simple Drive Player: -9.6

So, oddly enough, our attempt at improvements in the "Simple" agents actually made the agents perform worse. Always parking in close spots wasn't necessarily good, nor was avoiding handicap spots.

3 Performance of RL agents

For our RL testing we used the on-line averaging learning rate of 1/(1+N), where N is the number of actions so far taken. We used a simple exploration policy of $\epsilon = 0.1$, i.e. take a random action with probability 1/10 and take a greedy action with probability 9/10. We implemented Q-learning as our RL algorithm.

The average score of each of the two agents was:

Normal RL Player: -141.3
Impatient RL Player: -206.3

This was after training each agent over 5000 trials and then measuring their performance over 1000 trials. The fact that the agents ended up performing so poorly suggests a glitch in our update code, although we followed the Q-learning algorithm precisely. Looking at each agent's resulting policies, it appears that they generally prefer to park way too often, even though collisions are heavily negatively weighted at -1000.