Becca Dura & Joe Vivone
CS143 Final Project

Final Project Report

For our final project, we took code for a basic working implementation of Pacman from UC Berkeley and implemented numerous searches as well as reinforcement learning to make both the ghosts and the Pacman smarter. The UC Berkeley website we took the code off of had a lot of challenges to help implement the searches and reinforcement learning techniques into the code, so we mostly followed the challenges. All the code we added can be found under sections that say "YOUR CODE HERE". One of our folders (search) includes our implementation of breadth-first search, depth-first search, uniform cost search, and A* search to make the Pacman smarter. In that folder, we edited the "search.py" and "searchAgents.py" files. Our other folder (reinforcement) includes our working implementation of value iteration and Q-learning to create a Pacman agent that moves on its own. The Pacman agent we created finds a balance between fleeing from the ghosts and eating all the food. In that folder, we edited the "qlearningAgents.py" and "valueIterationAgents.py" files.

Throughout our final project, we learned a lot about how hard it is to extend code written by another person. We struggled quite a bit with figuring out how to use the existing files and functions to create the new functions and implement various Artificial Intelligence algorithms. We also had to search online for some tutorials on how to implement value iteration and Q-learning, since we did not have a lot of time for examples of those in class. Finally, we learned quite a bit about collaborating with others on code. We ended up doing a lot of work together, since that was easier than trying to email code back and forth every time we implemented something new. However, that ended up being nice, because we learned a lot from each other. It gave us more opportunities to explain concepts to each other, leading to us learning more in a quicker manner and being able to condense our code.

As written previously, we did most of the work together for this project. We both worked through the pre-existing starter code together, so we were on the same page and familiar with the code. On top of that, we both discussed which challenges we wanted to follow from the website and what exactly we wanted to implement together before actually writing any code. Although much of the project was worked on together, Becca did do more of the reinforcement learning (value iteration and Q-learning), while Joe did more of the searches (breadth-first search, depth-first search, uniform cost search, A* search). We both worked on the final report together. Overall, we believe that we both put in a similar amount of work and feel as if we learned a lot from working together on this project.

We believe we put in quite a bit of time on this project and really reinforced much of what we learned in class through the project. We were able to visualize the differences between various searches after implementing breadth-first search, depth-first search, uniform cost search, and A* search. On top of that, we were able to extend our knowledge of value iteration and Q-learning, which we only briefly began learning in class, leading to us gaining a better understanding of reinforcement learning. All of that leads us to believe that we earned the full 100 points on our final project.

Becca Dura & Joe Vivone
CS143 Final Project

Citations

http://ai.berkeley.edu/project_overview.html

This website was used to find the starter code for both the search and reinforcement folders. It is also where we found a lot of the challenges to help us implement breadth-first search, depth-first search, uniform cost search, A search, value iteration, and Q-learning.*

https://www.cs.princeton.edu/~andyz/pacmanRL

This website was used to help us better understand value iteration and Q-learning within the Pacman environment.

Pseudocode from the slides in class was also used to help implement the various searches and reinforcement learning techniques into the Pacman environment.

Instructions

- To see Pacman made smarter through the implementation of searches, cd to the location of your search folder, then run the following lines of code:
  - Depth-First Search
    - python pacman.py -l mediumMaze -p SearchAgent -a fn=dfs
  - Breadth-First Search
    - python pacman.py -l mediumMaze -p SearchAgent -a fn=bfs
  - Uniform Cost Search
    - python pacman.py -l mediumMaze -p SearchAgent -a fn=ucs
  - A* Search (the second option is more impressive)
    - python pacman.py -l mediumMaze -p SearchAgent -a fn=astar
    - python pacman.py -l bigMaze -z .5 -p SearchAgent -a fn=astar,heuristic=manhattanHeuristic
  - You can change the search technique being used to find the closest piece of food in the findPathToClosestDot function in the searchAgents.py file
- To run the Pacman agent using reinforcement learning techniques, cd to the location of the reinforcement folder, then run the following line of code:
  - python pacman.py -p ApproximateQAgent -a extractor=SimpleExtractor -x 50 -n 60 -l mediumClassic