3.2 Written Questions

In a sentence or two, answer the following questions about your implementation and/or observations of the programming portions above.

3.2.1 Depth First Search

1. (1 pt) Is the exploration order what you would have expected? Does Pacman actually go to all the explored squares on his way to the goal?

   In the case of DFS, the path explored is expected, it appears to "snake" around the board. Pacman does not (necessarily) actually go to all the explored squares on his way to the goal. This is the point of planning type artificial intelligence. This occurs any time the goal is not in the far most left branch.

2. (1 pt) Is this a least cost solution? If not, what do you think depth-first search is doing wrong.

   This is not a least cost solution unless the goal state happens to be in the first branch DFS explores. DFS guarantees a solution if it exists, not the best solution.

3.2.2 Breadth First Search

1. (1 pt) Does BFS find a least cost solution? If so, explain why.

   BFS does find a least cost solution because the fringe explores outward in a circular wave. When it encounters the goal state, it has found the shortest path.

3.3 Varying the Cost Function (Uniform Cost Search)

1. (1 pt) Specify the data structure used from util.py for uniform cost search

   The data structure used from util.py for uniform cost search is a priority queue.

3.4 A* search

1. (2 pts) What happens on openMaze for the various search strategies? Describe your answer in terms of the solution you get for A* and uniform cost search.

   BFS and UniformCostSearch look identical, both find the least cost solution but both also explore all but 1 space.
   A* is the best solution of all because it both expands the least number of spaces while also still finding that least cost path.
   DFS is the worst solution because pacman appears to forget which universe he lives in and snakes around the board as if he was a snake. Thankfully he does eventually make it to food.

3.5 Finding All the Corners

1. (2 pts) Describe in few words/ lines the state representation you chose or how you solved the problem of finding all corners.

   In order to find all the corners, I chose a game state which contained x,y,visitedCornersList. This allowed pacman to “go back” through spaces he had already visited in order to reach a corner that he had not visited yet. I also chose to have the goal state be that visited corners had a length of 4. This is a simple solution that fails if you have a board with 4+ corners. In this goal state, I also check that the corner being added is not already in the list of visited corners.

3.6 Corners Problem: Heuristic

1. (1 pt) Describe the heuristic you used for the implementation.

   The heuristic I used for my implementation heavily relies on the given function mazeDistance. I use this function to calculate the mazeDistance to each corner in the corners list. I append each solution to the end of a list. When all values have been calculated I return the max value. mazeDistance uses bfs to calculate the shortest distance from one point to another given a set of walls.

3.7 Eating All Dots

1. (2 pts) Describe the heuristic you used for the FoodSearchProblem.

   FoodSearchProblem uses a very similar solution as corners. For each food in food.asList, calculate mazeDistance using the given method, append to list, and return max value.

3.8 Suboptimal Search

1. (1 pt) Explain why the ClosestDotSearchAgent won’t always find the shortest possible path through the maze.

   ClosestDotSearchAgent won’t always find the shortest possible path through the maze if there are multiple food dots that are not along the shortest path. In the case of all the mazes, we are given this is not the case.

4 Self Analysis (5 pts)

Each group member must answer these questions individually.

1. What was the hardest part of the assignment for you?

   The hardest part for me was figuring out question 7. For a long time, the tests were saying my manhattan distance heuristic was not good enough. When I eventually switched to the given mazeDistance it took a little modification of my problem object to mazeDistance to run.

2. What was the easiest part of the assignment for you?

   The easiest part of this assignment was getting the searches set up. For the most part they are all exactly identical

3. What problem(s) helped further your understanding of the course material?

   Cornershuerisitc helped me further understand the course material because it opened up my mind about what could be a goal state. Even in pacman it is not necessarily a state within the game itself but a meta state for lack of a better word.

4. Did you feel any problems were tedious and not helpful to your understanding of the material?

   I felt that question 7 was a bit tedious. I felt that there were parts of the test that I could not get to work despite having a good grasp on what I needed or wanted to do in order to solve the issue.

5. What other feedback do you have about this homework?

   It was interesting looking at the mazeDistance method and seeing that it uses bfs. Using bfs within a search method as a heuristic feels… wrong.

   I know that is not really feedback but I needed to say that.