ANSWERS FOR THE QUESTIONS

600.335/435 Artificial Intelligence---Homework 2

Bo Lei

•***Grad Students: Create a new cost function and provide the solution it generates with tinyMaze, mediumMaze, mediumDottedMaze, mediumScaryMaze, bigMaze, and openMaze. Why did you choose this cost function and how did it alter the results you see for these environments?***

In *searchAgents.py,* we have:

```
class StayEastSearchAgent(SearchAgent):
    def __init__(self):
        self.searchFunction = search.uniformCostSearch
        costFn = lambda pos: .5 ** pos[0]
        self.searchType = lambda state: PositionSearchProblem(state, costFn, (1, 1), None, False)

class StayWestSearchAgent(SearchAgent):
    def __init__(self):
        self.searchFunction = search.uniformCostSearch
        costFn = lambda pos: 2 ** pos[0]
        self.searchType = lambda state: PositionSearchProblem(state, costFn)
```

So we know the stay-east search agent has the cost function: $f = 0.5^x$. In other words, the more the point near east, where the value of x is bigger, the less the cost function will give the value. On the contrary, the stay-west search agent has the cost function: $f = 2^x$, which means the more the point is near west, where the value of x is smaller, the less the cost function will give the value. In this way, we can create our own cost function as long as it meets the property mentioned above. For example, for the stay-east search agent, we can create a new cost function as $f = \frac{1}{x}$ and for stay-west search agent, we can use a new cost function like $f = x$.

•***Grad Students: Complete question 6, which creates a new heuristic for the corners problem. Provide your solution for tinyCorners and mediumCorners. How did you choose this heuristic and how well did it perform?***

Heuristic function, see *def cornersHeuristic(state, problem)* in *searchAgents.py*.

Considering the goal is just a state with four corners visited, this Heuristic function simply calculate the total Manhattan distance from the current position to the final state. I first calculate the distance between the current position and the closest unvisited corner. Then calculate the distance between the first unvisited closest corner and the new closest unvisited corner until it reaches the state with four corners all visited.

Suppose we have n corners unvisited, the Manhattan distance between any two corners of these n unvisited corners, which is also the shortest distance between them, is fixed. So in order to make the Heuristic function admissible, the only thing we need to do is to make the distance between the point's potion and the first corner to visit to be minimum. Obviously, we need to go to the closest unvisited corner at the beginning. In this way, we prove it is admissible.

Moreover, since we use Manhattan distance to measure the total distance, the function will be consistent.

Following is the result I run the program with my Heuristic function. With the numbers of expanded nodes 693, we see clearly it performs pretty well.

```
Path found with total cost of 106 in 0.1 seconds
Search nodes expanded: 693
Pacman emerges victorious! Score: 434
Average Score: 434.0
Scores:        434.0
Win Rate:      1/1 (1.00)
Record:        Win
```

•Grad Students: Implement foodHeuristic and provide your solutions using this new heuristic for testSearch and trickySearch. How did you choose this heuristic and how well does it perform? Justify why your heuristic is admissible (remember A* relies on the heuristic being admissible to be optimal).

Heuristic function, see *def foodHeuristic(state, problem)* in *searchAgents.py*.

For this heuristic function, the goal state is eating out all the food. We search through Astar method. To make it admissible, we need to guarantee the cost <= actual cost. The function mazeDistance() just returns a distance from one position to the other using the search method we use in the problem. So we could use this function to get the minimum distance from the current position to any of the food point. Since we have to go to the farthest food point in order to eat out all of the food, if we use the farthest distance to be the Heuristc, it will be the minimum actual cost when all the other food points is on the way to the farthest food point. For other situations, the actual cost would just be more than this one since we need to move away from this way to get another food point. So the heuristic function is admissible as well as consistent.

Following is the result I run the program with my Heuristic function. With the numbers of expanded nodes 4240, we see clearly it performs pretty well compared with the standards.

```
Path found with total cost of 60 in 37.3 seconds
Search nodes expanded: 4240
Pacman emerges victorious! Score: 570
Average Score: 570.0
Scores:        570.0
Win Rate:      1/1 (1.00)
Record:        Win
```