Amit Nijjar

A11489111

October 10, 2016

CSE 150

Fall 2016

Assignment 1 Report

Description of the problem and the algorithms used to solve all the problems:

The problem of the lab is as follows: to write functions that allow pacman to be played automatically. By implementing different functions using different search algorithms, I was able to achieve the goal of the assignment. The algorithms used to solve all of the problems are depth first search, breadth first search, uniform cost search, A* search, Heuristic and other algorithms to implement functionality allowing pacman to find corners.

Describe the data structure used in each algorithm:

Algorithm	Data Structure
Depth First Search	Stack
Breadth First Search	Queue
Uniform Cost Search	Priority Queue
A* Search	Priority Queue

Perform some analysis on the efficacy of the different algorithms with different puzzles. For instance, you can then measure the number of nodes visited / maximum size of the queue / time it took to run the code/ path length given by algorithm etc., for each case. How do they compare against what you expect from the big-O analysis? How do they compare against each other?

1 - Call used:

```
python pacman.py -l mediumCorners -p SearchAgent -a fn=___,prob=CornersProblem
```

Data 1

Algorithm	Total Path Cost	Nodes Expanded	Average Score
Depth First Search	217	373	323
Breadth First Search	106	1988	434
Uniform Cost Search	106	1988	434
A* Search	106	1988	434

According to the data shown above and referencing the lecture slides showing the algorithms' big-O, the data is consistent with expectations. Similarly, reflected in the differences of expected outcome, depth first search does not perform as well as the other search algorithms when playing pacman. More nodes have to be expanded and the path cost is far from optimal when using DFS. The other search algorithms proved to be more efficient during testing.

Results of the analysis and a short discussion?

As you can see above, the results of my analysis are shown in the graph containing the algorithms and their respective statistics. Breadth first search type algorithms and greedy algorithms work well for making a pacman AI. Depth first search is not a good search algorithm to use for a pacman AI. The game map of pacman allows for many loops to arise when finding a path. Using DFS will cause pacman to trace many spots over and over again in order to find all of the food dots. Using BFS or the other two algorithms instead of DFS would lead to much more optimal results. These algorithms don't span as far as DFS but stay close to the root node. Because of this, there is less backtracking in the solutions provided by these algorithms, leading to a shorter path.

Paragraph from each author stating what their contribution was in this assignment.

I worked on and completed this assignment on my own. I wrote the code myself with guidance from the tutors in the labs and help from posts on Piazza. I implemented the algorithms and completed the report. I am unsure if I need to say more, please let me know if I need to include anything else so that I can include it in my next assignment. Thank you!