CSC 665 - Multi-agent Pacman Search

Khanh Nguyen

This project implements the evaluation function for Pacman as a Reflex Agent to escape the Ghost(s) while eating as many dots as possible.

Overview

- Link to the project's original specs
- With the new game setup, Pacman now needs to find its way out from being captured by ghost agents. For part 1 of this project, the program will be implementing Pacman to act as a **Reflex Agent**.
- The search agents can be found in `multiagent/multiAgents.py`
- The pacman can be found in `multiagent/pacman.py`
- The utilities class can be found in `multiagent/util.py`

Problem Statements

Part 1 - Reflex Agent

- Improve the ReflexAgent in `multiAgents.py` to play respectably. The provided reflex agent code provides some helpful examples of methods that query the GameState for information. A capable reflex agent will have to consider both food locations and ghost locations to perform well.

How to run

- Download or git clone this repo if not already
- In terminal, go to directory:
    - `cd path/to/CSC665-multi-agent-pacman`
- To test **Depth First Search (DFS)**:
    - `python pacman.py -p ReflexAgent -l testClassic`

Solution Design

Evaluation Function

```
# in multiagents.py
class ReflexAgent(Agent):

    def evaluationFunction(self, currentGameState, action):
```

- Given the list of foods and ghosts, Pacman can easily find the distances to such item or agents on the map. To enhance the evaluation function, Pacman needs to find what would be the immediate best action to take, based on the score from this function.

Solution Implementation

Part 1

- The relevant files for this part are `multiagents.py` and `pacman.py`.
- Essentially, the key to the implementation for this function is to use the **reciprocal** of important values (such as distance to food) rather than just the values themselves.
- This would tremendously affect the score for each action based on the distance to foods/ghost.
- In other words, instead of using a linear combination of distances, the function can utilize this relationship **closestGhostDistance / closestFoodDistance**

Solution Result/Evaluation

- My program was able to successfully enhance the scores for Pacman to eat more dots while avoid the ghosts. The average score from 10 games running by the `autograder.py` was in the 500 threshold.

Conclusion

- This can be thought of a simpler way of thinking of heuristics function, which was the majority of project 1.
- In the future, evaluation function can be enhanced by adding characteristics of heuristics function.
- Reflex Agent, although easy to compute, is not smart or optimal enough. Therefore, we include in the second part of this project to utilize the minimax algorithm.