Project 2: Multi-Agent Search in Pacman

CS 4300: Artificial Intelligence
University of Utah

In this project, you will design agents for the classic version of Pacman, including ghosts. Along the way, you will implement both minimax and expectimax search and try your hand at evaluation function design.

The code base has not changed much from the previous project, but please start with a fresh installation, rather than intermingling files from project 1.

As in project 1, this project includes an autograder for you to grade your answers on your machine. This can be run on all questions with the command:

python autograder.py

The code for this project consists of several Python files, some of which you will need to read and understand in order to complete the assignment, and some of which you can ignore. Download p02.zip which will contain all the code and supporting files.

1 Files to edit

For all the problems in this project, you need to edit just one python file, namely:

- **multiAgents.py:** where all of your multi-agent search agents will reside.

2 Supporting Files

The following python files will help you in understanding the problem and will get you familiar with the different data structures and games states in Pacman.

- **pacman.py:** The main file that runs Pacman games. This file describes a Pacman GameState type, which you use in this project.
- **game.py:** The logic behind how the Pacman world works. This file describes several supporting types like AgentState, Agent, Direction, and Grid.
- **util.py:** Useful data structures for implementing search algorithms.

3 Multi-Agent Search in Pacman (30 pts)

For all the problem titles described below, please refer to this multi-agent project page for the problem description and what is expected of each problem. As always autograder has different test cases against which you can run your program to check the correctness. For the questions asked below, please ensure your response is brief and to the point. Please don't write paragraphs of text as responses to these questions.

3.1 Programming Implementation (25 pts)

1. Reflex Agent (4 pts)
2. Minimax (5 pts)
3. Alpha-Beta Pruning (5 pts)
4. Expectimax (5 pts)
5. Evaluation Function (6 pts)

3.2 Written Questions (4 pts)

3.2.1 Reflex Agent

1. (1 pt) What feature (or features) did you use for your evaluation function?

3.2.2 Minimax

1. (1 pt) When Pacman believes that his death is unavoidable, he will try to end the game as soon as possible because of the constant penalty for living. Give an explanation as to why the Pacman rushes to the closest ghost in this case ?

3.2.3 Expectimax

1. (1 pt) You should find that your ExpectimaxAgent wins about half the time, while your AlphaBetaAgent always loses. Explain why the behavior here differs from the minimax case.

3.2.4 Evaluation Function

1. (1 pt) What features did you use for your new evaluation function?

4 Self Analysis (5 pts)

Each group member must answer these questions individually.

1. What was the hardest part of the assignment for you?
2. What was the easiest part of the assignment for you?
3. What problem(s) helped further your understanding of the course material?
4. Did you feel any problems were tedious and not helpful to your understanding of the material?
5. What other feedback do you have about this homework?

5 Evaluation

Your code will be autograded for technical correctness. Please do not change the names of any provided functions or classes within the code, or you will wreak havoc on the autograder. If your code passes all the test cases in the autograder you will receive full points for the implementation.

However even if your code does not necessarily pass all the test cases, we will evaluate your code and then award you partial points accordingly. In such cases it would be even more beneficial if you could give a short description of what you tried and where you had failed and that would help us in giving you better points.

6 Submission Instructions

- For the final submission you will be turning in the following items (please do not zip them together but instead submit them separately):

  1) the **multiAgents.py** file - include both group members' names and uids in a comment at the top of the file.

  2) one PDF document *per group* containing your responses to questions from previous sections. Please ensure that the pdf has two separate answers for the Self-evaluation section, but one answer for the rest.

- Please ensure all the submissions are done through gradescope. Please do not email the instructor or the TA's with your submission. Submissions made via email will not be considered for grading.

- **Written Answers:** Place all your written answers from Sections 3 and 4 in a single PDF document. This should be clearly named in the format ⟨uid1⟩-⟨uid2⟩-Proj⟨number⟩-answers.pdf, where ⟨uid1 and 2⟩ are the Utah uids of the two group members (if applicable) and ⟨number⟩ is the Project number. Ex: u0004300-u0001337-Proj2-answers.pdf. Please make sure to write your and your partner's name at the top of the document!

- **Naming:** Your python file(s) upload should be kept under the original name(s), i.e. **multiAgents.py**

- For this project fill in portions of the files to edit and answer the complementary questions. Once you have completed the code and the answers, name them as per the conventions stated above and submit the requested files via gradescope.

- Please do NOT install/use additional libraries or packages other than what is provided and the python standard library.