CS461 Artificial Intelligence
Homework 1

Spring 2023

Due: March 11, 2023, 23:59

Instructions

- You will submit your python codes with .py format to Moodle. Only modify the specified parts (highlighted by `*** YOUR CODE HERE ***` comments) of the provided files, you will lose the whole grade otherwise.
- You will submit your codes individually or in groups of 2. Please also include the name and the student ID of every member in a text file named `members.txt`.
- You will submit only three files: `search.py`, `searchAgents.py`, and `members.txt`. Any other file will be ignored.
- You will need `Python 3` and `pygame` [1] installed on your computer to run the environment. Your implementation must not change the package/version requirements.
- Only one member should upload a .zip file including the mentioned python files.
- The codes will be checked for plagiarism using online tools which are difficult to fool. This check will include publicly available implementations for this homework. Submit your own work.
- Your codes will also be evaluated in terms of efficiency. Make sure you do not have unnecessary loops and obvious inefficient calculations in your code.
- We follow no extension policy. However, the entire group will lose 25 percent of the grade per day of late submission, up to 2 days.
- The files have been tested before being uploaded to Moodle. However, if you still have problems using and running the codes, you can contact the course TAs: Barış Bilgin Şenol (bilgin.senol@bilkent.edu.tr) and Navid Ghamari (navid.ghamari@bilkent.edu.tr).

1 Search Algorithms

In this course you will learn several search algorithms that can be used in many domains. One of these domains is agent based games that include path finding. In this assignment, you will implement some of these search algorithms that will be used by a Pacman agent in the given environment. You will implement/complete these in the `search.py` file. These algorithms are:

- Depth First Search
- Breadth First Search
- Uniform Custom Search
- A* Search

You are provided a folder including multiple python files. Running the pacman.py script, you will be able to play on a sample map of the Pacman game. However, we intend to implement an intelligent agent that can complete the game by itself. The search algorithms you write will be used by the agent to choose the optimal or suboptimal path to achieve a goal. You may also want to take a look at the following files:

- `pacman.py`
- `game.py`
- `util.py`

Although you will find in these files all the information and data structures that you need to complete the homework, you are free to add functions or data structures as long as:

- You don't change any file other than `search.py` and `searchAgents.py`.
- You don't use external packages (other than requirements).

2 Search Agents

Other than the general search algorithms that you write, you need to implement an agent that utilizes these algorithms to decide and act in a given environment/problem. In this case, the problem is finding the closest path for the agent to collect points. You will complete the `searchAgents.py` file and implement/complete the following classes and functions:

- `CornersProblem`: search to find all the corners in a given game map/layout
- `CornersHeuristic`: a heuristic that you will use finding the corners
- `FoodHeuristic`: a heuristic that is used in the `AStarFoodSearchAgent` class. You should not change the codes for the `AStarFoodSearchAgent` and `FoodSearchProblem` classes.
- `ClosestDotSearchAgent`: an agent that finds the closest dot/food using the search algorithms you have implemented.
- `AnyFoodSearchProblem`: you should complete this class so that you can use it to find the closest dot

Note: You can find more detail in the python files as comments, as well as on the course web page of the Berkeley University [2].

3 Grading

Your homeworks will be graded using an autograder already present in the provided .zip file. You can evaluate your code running the `autograder.py` script which tests your code with the given test cases. However, there might be other cases not provided in the zip file. Also, your code will be evaluated for efficiency. The time it takes the autograder to grade your implementation will be measured and the implementations considerably faster than average will receive up to 10 percent bonus and the slow implementations will lose up to 10 percent of the grade accordingly.

References

1. Pygame package https://www.pygame.org/wiki/GettingStarted
2. Berkeley project web page https://inst.eecs.berkeley.edu/ cs188/sp23/projects/proj1/