Report for assignment 4, TDT4316

Eliah Mathias Hudgins

October 20, 2023

1 Introduction

The following report contains the test results for the Minimax agent alphabeta pruning agent. The results from the autograder are added in the order of the tests. As for the creation of the two algorithms, a detailed description of how they work is commented into the code for easier understanding of how the two algorithms function. However, i have added a short breakdown/code overview here as well.

2 Code overview

1. **Minimax Agent Class:** This class implements the classic Minimax algorithm, a decision-making algorithm used for finding the best move in a two-player game.
2. **"getaction" method:** Determines the best action for the Agent 0 (Pacman) given the current "gamestate" by calling the minimax function and then returning the action that is deemed optimal for pacman
3. **"minimax" Method:** Finds the best mnove for the current agent (Pacman or ghosts) and returns the move and its value. This is done by checking if the game has reached a terminal state or fi the macc depth is reached. However, depending on the agent, it will whether maximise or minimize the evaluation function
4. **"maximise" Method:** Finds the action that maximizes the value for Pacman by iterating through all of Pacmans legal moves and selected the one with the highest possible value
5. **"minimize" Method:** Finds the action that minimizes the value for a ghost by iterating through all of the ghosts legal moves and selects the one with the lowest value

1. **AlphaBetaAgnet class:** This class implements the Minimax algorithm with Alpha-Beta pruning. Alpha-beta pruning is an optimization for the minimax algorithm that decreases the number of nodes evaluated in the search tree.

2. **"getaction" Method:** It determines the best action for Pacman given the current gameState by using alpha-beta pruning to initialize alpha and beta values and then searching through Pacman's possible actions. This uses the value function to evaluate the worth of each move. It also performs pruning based on alpha and beta values to optimize the search

3. **"maxValue" function:** This function determines the maximum value Pacman can achieve from the current game state while considering the future moves of both Pacman and the ghosts. This is done by iterating through Pacman's legal moves, evaluating the resulting game state, and updating the best value (and the corresponding move) found so far. It also implements the alpha-beta pruning step for the maximizer.

4. **"minValue" function:** It determines the minimum value the ghosts can achieve from the current game state. It itterates through a ghost's legal moves, evaluates the resulting game state, and updates the best value (and the corresponding move) found so far. Implements the alpha-beta pruning step for the minimizer.

5. **"value" function:** This function directs the evaluation process depending on whose turn it is by calling either "maxValue" for Pacman or "minValue" for a ghost, depending on the "agentIndex".

3 Breakdown of Minimax Agent

The Minimax Agent is very traditional and uses the normal Minimax algorithm blueprint. Here is a snapshot:

- **GetAction Method:** Acts as Pacman's compass, it shows the optimal move given the game's current state.

- **Minimax Method:** This serves as the oracle, predicting the most "optimal" move for the agent in play. It differentiates its strategy based on the agent's identity – Pacman or ghost.

- **Maximize and minimize Methods**: These are the twin engines driving Pacman's and the ghosts' decision-making processes. They sift through the potential moves, using those which lead towards the desired gamestates for their respective agents.

For the complete explanation of the code and how it works, see the uploaded python file for further specification

4 Quick dissection of the AlphaBeta agent

The AlphaBeta Agent still uses the Minimax algorithm, however it introduces the of Alpha-Beta pruning. Here's its essence:

- **GetAction Method:** It navigates through Pacman's landscape, using alpha-beta pruning to find the optimal path, without unnecessary detours.
- **MaxValue and minValue Functions:** These functions make sure Pacman and the ghost are playing "optimally". They appraise potential futures, making choices that are most likely to yield a winning state.
- **Value Function:** Acting as the decision-making tree, it defers to either maxValue or minValue based on the active agent, ensuring each gets their respective "turn".

For the complete explanation of the code and how it works, see the uploaded python file for further specification

5 Output from Autograder test for question 2

```
PS C:\Users\Eliah\Documents\Universitetet\AI\ASsignment 3\multiagent>
python autograder.py -q q2
C:\Users\Eliah\Documents\Universitetet\AI\ASsignment 3\multiagent\autograder.py:17:
DeprecationWarning: the imp module is deprecated in favour of importlib and
slated for removal in Python 3.12; see the module's documentation for alternative uses

  import imp
Starting on 10-20 at 19:11:26

Question q2
===========

*** PASS: test_cases\q2\0-eval-function-lose-states-1.test
*** PASS: test_cases\q2\0-eval-function-lose-states-2.test
*** PASS: test_cases\q2\0-eval-function-win-states-1.test
*** PASS: test_cases\q2\0-eval-function-win-states-2.test
*** PASS: test_cases\q2\0-lecture-6-tree.test
*** PASS: test_cases\q2\0-small-tree.test
*** PASS: test_cases\q2\1-1-minmax.test
*** PASS: test_cases\q2\1-2-minmax.test
*** PASS: test_cases\q2\1-3-minmax.test
*** PASS: test_cases\q2\1-4-minmax.test
*** PASS: test_cases\q2\1-5-minmax.test
*** PASS: test_cases\q2\1-6-minmax.test
*** PASS: test_cases\q2\1-7-minmax.test
*** PASS: test_cases\q2\1-8-minmax.test
*** PASS: test_cases\q2\2-1a-vary-depth.test
```

```
*** PASS: test_cases\q2\2-1b-vary-depth.test
*** PASS: test_cases\q2\2-2a-vary-depth.test
*** PASS: test_cases\q2\2-2b-vary-depth.test
*** PASS: test_cases\q2\2-3a-vary-depth.test
*** PASS: test_cases\q2\2-3b-vary-depth.test
*** PASS: test_cases\q2\2-4a-vary-depth.test
*** PASS: test_cases\q2\2-4b-vary-depth.test
*** PASS: test_cases\q2\2-one-ghost-3level.test
*** PASS: test_cases\q2\3-one-ghost-4level.test
*** PASS: test_cases\q2\4-two-ghosts-3level.test
*** PASS: test_cases\q2\5-two-ghosts-4level.test
*** PASS: test_cases\q2\6-tied-root.test
*** PASS: test_cases\q2\7-1a-check-depth-one-ghost.test
*** PASS: test_cases\q2\7-1b-check-depth-one-ghost.test
*** PASS: test_cases\q2\7-1c-check-depth-one-ghost.test
*** PASS: test_cases\q2\7-2a-check-depth-two-ghosts.test
*** PASS: test_cases\q2\7-2b-check-depth-two-ghosts.test
*** PASS: test_cases\q2\7-2c-check-depth-two-ghosts.test
*** Running MinimaxAgent on smallClassic 1 time(s).
Pacman died! Score: 84
Average Score: 84.0
Scores:        84.0
Win Rate:      0/1 (0.00)
Record:        Loss
*** Finished running MinimaxAgent on smallClassic after 15 seconds.
*** Won 0 out of 1 games. Average score: 84.000000 ***
*** PASS: test_cases\q2\8-pacman-game.test

### Question q2: 5/5 ###


Finished at 19:11:41

Provisional grades
==================
Question q2: 5/5
------------------
Total: 5/5

Your grades are NOT yet registered.  To register your grades, make sure
```

to follow your instructor's guidelines to receive credit on your project.

6 Output from Autograder test for question 3

```
PS C:\Users\Eliah\Documents\Universitetet\AI\ASsignment 3\multiagent>
python autograder.py -q q3
C:\Users\Eliah\Documents\Universitetet\AI\ASsignment 3\multiagent\autograder.py:17:
DeprecationWarning: the imp module is deprecated in favour of importlib and
slated for removal in Python 3.12; see the module's documentation for alternative uses

  import imp
Starting on 10-20 at 19:20:05

Question q3
===========

*** PASS: test_cases\q3\0-eval-function-lose-states-1.test
*** PASS: test_cases\q3\0-eval-function-lose-states-2.test
*** PASS: test_cases\q3\0-eval-function-win-states-1.test
*** PASS: test_cases\q3\0-eval-function-win-states-2.test
*** PASS: test_cases\q3\0-lecture-6-tree.test
*** PASS: test_cases\q3\0-small-tree.test
*** PASS: test_cases\q3\1-1-minmax.test
*** PASS: test_cases\q3\1-2-minmax.test
*** PASS: test_cases\q3\1-3-minmax.test
*** PASS: test_cases\q3\1-4-minmax.test
*** PASS: test_cases\q3\1-5-minmax.test
*** PASS: test_cases\q3\1-6-minmax.test
*** PASS: test_cases\q3\1-7-minmax.test
*** PASS: test_cases\q3\1-8-minmax.test
*** PASS: test_cases\q3\2-1a-vary-depth.test
*** PASS: test_cases\q3\2-1b-vary-depth.test
*** PASS: test_cases\q3\2-2a-vary-depth.test
*** PASS: test_cases\q3\2-2b-vary-depth.test
*** PASS: test_cases\q3\2-3a-vary-depth.test
*** PASS: test_cases\q3\2-3b-vary-depth.test
*** PASS: test_cases\q3\2-4a-vary-depth.test
*** PASS: test_cases\q3\2-4b-vary-depth.test
*** PASS: test_cases\q3\2-one-ghost-3level.test
*** PASS: test_cases\q3\3-one-ghost-4level.test
```

```
*** PASS: test_cases\q3\4-two-ghosts-3level.test
*** PASS: test_cases\q3\5-two-ghosts-4level.test
*** PASS: test_cases\q3\6-tied-root.test
*** PASS: test_cases\q3\7-1a-check-depth-one-ghost.test
*** PASS: test_cases\q3\7-1b-check-depth-one-ghost.test
*** PASS: test_cases\q3\7-1c-check-depth-one-ghost.test
*** PASS: test_cases\q3\7-2a-check-depth-two-ghosts.test
*** PASS: test_cases\q3\7-2b-check-depth-two-ghosts.test
*** PASS: test_cases\q3\7-2c-check-depth-two-ghosts.test
*** Running AlphaBetaAgent on smallClassic 1 time(s).
Pacman died! Score: 84
Average Score: 84.0
Scores:        84.0
Win Rate:      0/1 (0.00)
Record:        Loss
*** Finished running AlphaBetaAgent on smallClassic after 15 seconds.
*** Won 0 out of 1 games. Average score: 84.000000 ***
*** PASS: test_cases\q3\8-pacman-game.test

### Question q3: 5/5 ###


Finished at 19:20:20

Provisional grades
==================
Question q3: 5/5
------------------
Total: 5/5

Your grades are NOT yet registered.  To register your grades, make sure
to follow your instructor's guidelines to receive credit on your project.
```