CS 580: Introduction to Artificial Intelligence

Project 2: Square in 4
with the Minimax Algorithm and Alpha–Beta Pruning

Due: Sun, Apr 9 at 11:59 pm ET

INSTRUCTIONS

- This Project is considered individual effort and the honor code applies when reviewing the implementation.
- Submit your solution as `P2_<username>.py`, and your report `P2_<username>.pdf`, where `<username>` is your Mason account.
- Once the link for submission is closed, we do not accept resubmissions, so it is the responsibility of the student to verify that both files are the correct ones and can be extracted.
- Multiple submissions are allowed, and the last submission is the one that is graded.

NOTE: Late submissions are not accepted, once the submission link is closed it is not reopened, so it is the responsibility of the student to verify that their files are correct and is not a corrupted file.

EXTRA CREDIT

Category	Score
The last submission was at most on April 8, 11:59 pm ET	+5 points

PENALTIES

Category	Score
Wrong file name `.py`	-3 points
Wrong file name `.pdf`	-3 points
Wrong format (it's not a `pdf` file)	-4 points

Introduction

Adversarial search is a kind of search in which we can trace the movement of an "enemy" or "opponent" who is changing the state of the of the problem every step in a direction you do not want.

The **Minimax** is a **backtracking algorithm** used in decision-making and **game theory** to determine the best move for a player, provided that your opponent also plays optimally. The two participants are referred to as the maximizer (**MAX**) and the minimizer (**MIN**). MAX strives to achieve the maximum score, whereas MIN seeks to achieve the lowest score. The Minimax algorithm has been used to solve several games, including **Connect 4**.

Implementation

In this project you will implement in Python the Minimax algorithm and Alpha-Beta Pruning (used in Adversarial Search) to play a modified version of the Connect 4 game.

Access the following link to download the Minimax algorithm code to solve the Connect 4 game and review the videos that will be used for this project:
https://github.com/KeithGalli/Connect4-Python

Game Rules:

- Players must alternate turns, and only one disc can be dropped in each turn.
- On your turn, drop one of your colored discs from the top into any of the seven slots.
- The game ends when there is 4 discs of the same color connected in a square.

User Input - Menu

- Select the color of the board from this palette of 16 colors: https://www.color-meanings.com/shades-of-yellow-color-names-html-hex-rgb-codes/
- Enter the player's name.
- Select who is the first player: `<player's name>` or `<agent's name>`
- How deep is the search for the Minimax `[1-5]`.

Fixed Features

- **Discs**: black and white, **Background**: grey

Output

- Show the name of the winner and the time the game lasted.
- How many moves (player 1 + player 2) were necessary to win.
- Extra data is optional.

Deliverables

Submit two files: the **code** `P2_<username>.py` and a **report**. `P2_<username>.pdf`.
After your implementation, play against the agent to do an experiment. You will always be the first player; in each game you will select each of the different columns `[1-7]` and each of the different values for deep `[1-5]`. Record who is the winner in each of the `35` different plays, how many moves were necessary to win and other data that you consider important to make an analysis of the results inside the Experiment Results section in your report.

NOTE: A good implementation would make the agent win at least 6/7 games when there is a deep = 5 assuming the opponent plays optimally.

Generate a document written with Calibri 12, single spaced with at least the following sections:

- **Introduction:**
  What is adversarial search, the purpose of the agent in this type of search, what are the algorithms that were implemented, what is the problem to be solved, what are the sections of the report and a brief explanation of what is contained in each of them.

- **Background:**
  The definition and template of the minimax algorithm, the Alpha – Beta pruning algorithm and template, the explanation of the square in 4 game and a figure showing a board with a winner state.

- **Proposed Approach:**
  Template of the algorithm (NOT the entire code!) and information about the strategy followed to assign points for the "deep" of the minimax algorithm.

- **Experimental Results:**
  Explain how the experiment was carried out, what is the purpose of the experiment and show a summary table with a summary of the 35 plays. Include at least one relevant Figure related with this experiment.

- **Conclusions:**
  What can be inferred from the experiment and the level of deep? Other relevant conclusions that were derived from your own.

The description of each section is the minimum requirement, feel free to complement each section with data that may be useful to enrich the content.