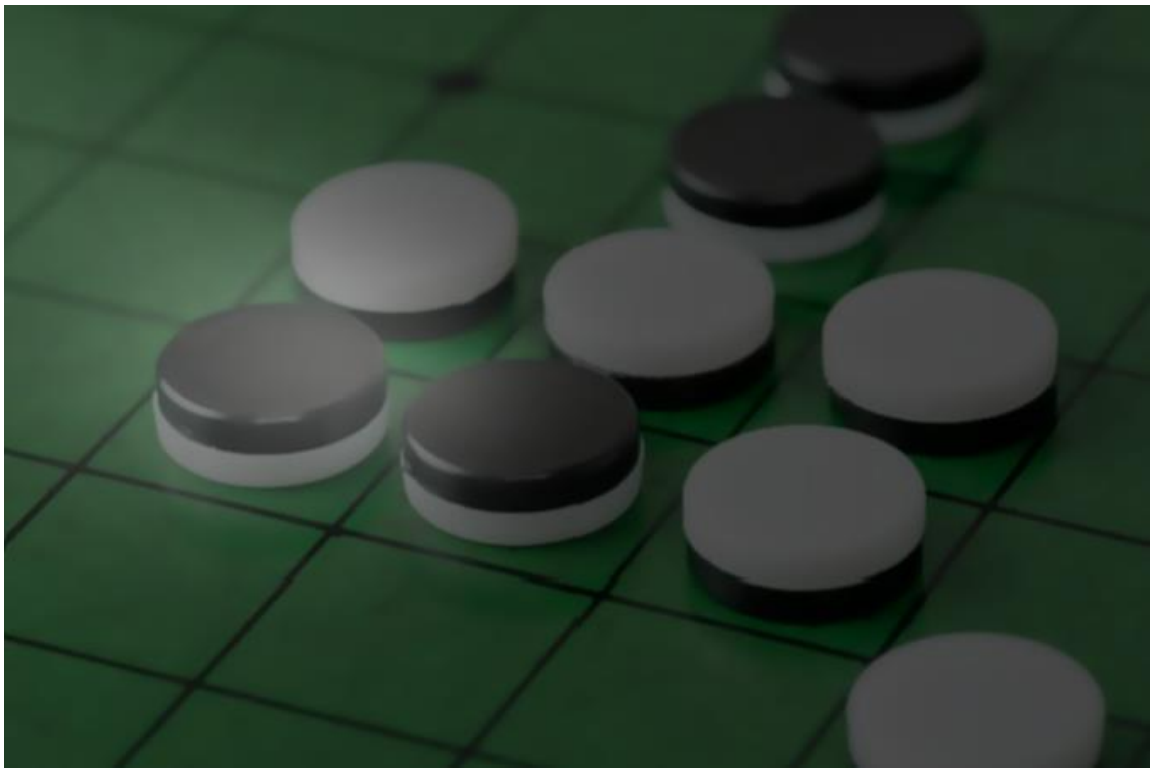


AI Project

(Othello AI Player)

Table of Contents

Team members 3
GitHub link 3
The programming language and framework 3
UML diagrams 4
1. Class Diagram 4
2. State diagram 5
3. Sequential Diagram 6
Game playing algorithms. 7
Used heuristics 7
Benefits of using heuristics: 8
Supported features 9
Maximum difficulty level supported 9
Level hard to play against the AI 10
Video Link 10

Team members

Name	ID	Contribution Percentage
Abdelrhman Abdelaziz Ramadan	1809918	14
Mohamed Gamal Saleh Mohamed	1807570	14
Mohamed Reda Ismail Mohamed	1804229	12
Osama Muhammad Ramadan	1902255	12
Abdelrahman Elsayed Mohamed	1803842	12
Kyrillos Phelopos Sawiris	1804628	12
Androw Ashraf Shenoda attalla	1808004	12
Kirollos Gerges Sobhy	1805147	12

GitHub link

Link: https://github.com/Abdelrhman-Abdelaziz/Othello-AI-Player.git

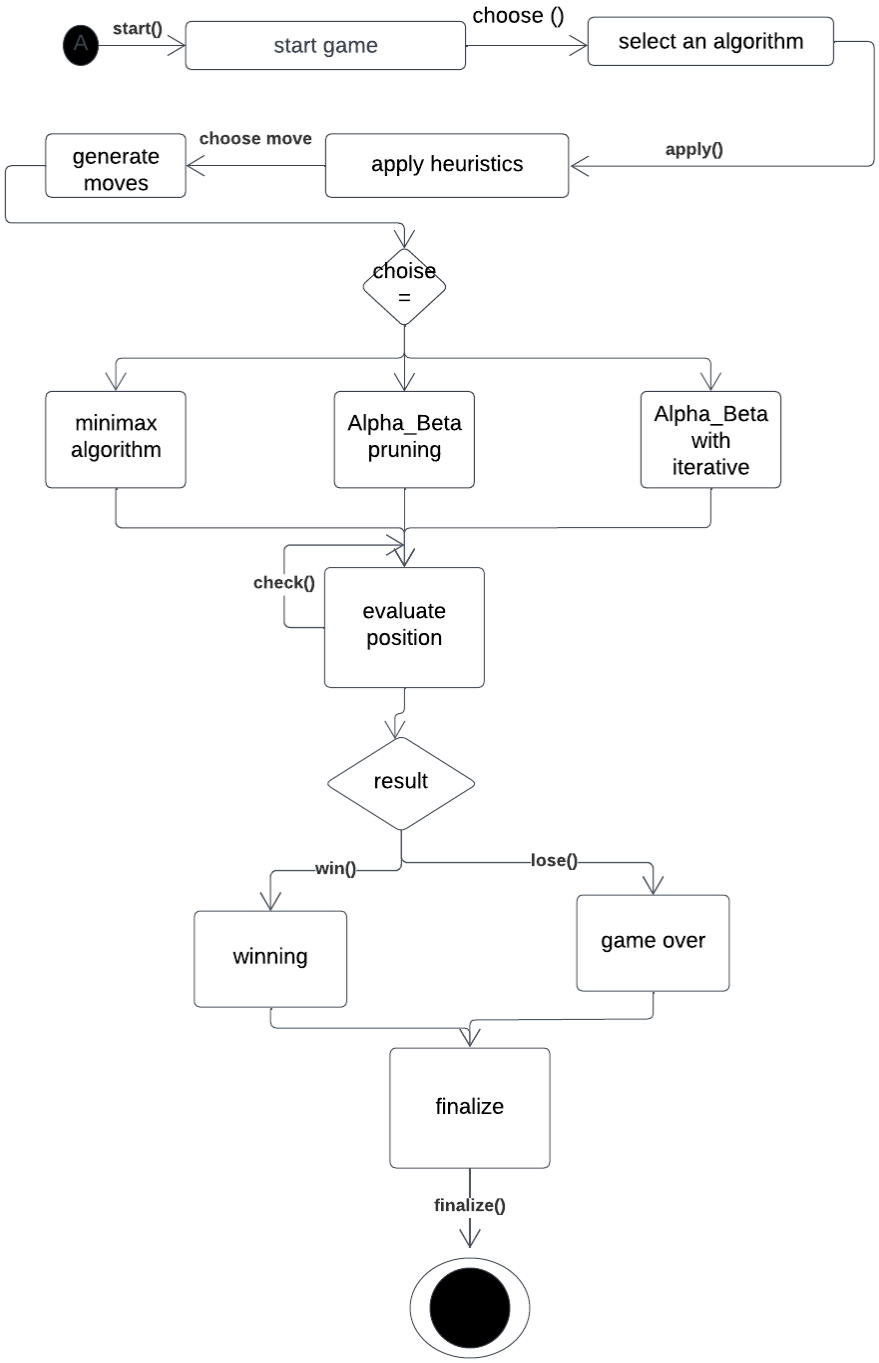
The programming language and framework

The programming language and framework selected for our project consist of **Python** as the primary programming language and **Pygame** as the chosen GUI framework.

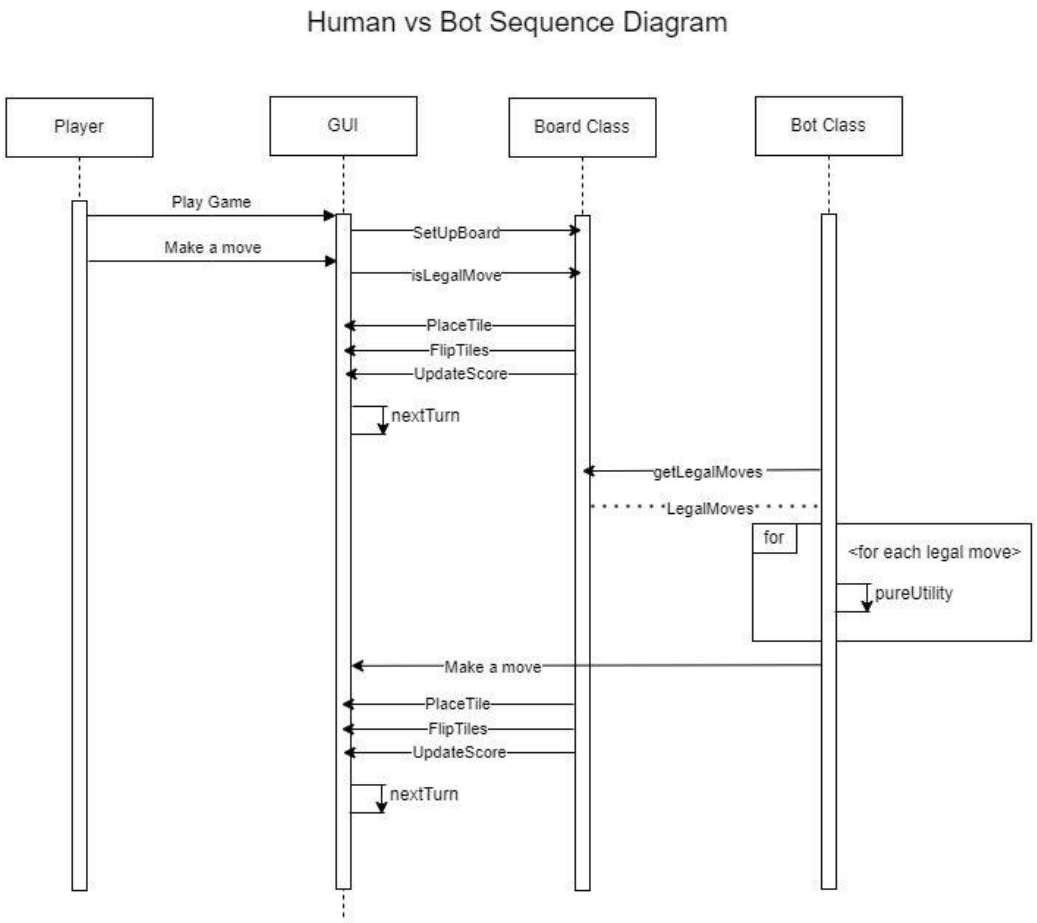
UML diagrams

1. Class Diagram

2. State diagram



3. Sequential Diagram



Game playing algorithms.

In this project, we have implemented two game playing algorithms: the minimax algorithm and the alpha-beta pruning algorithm, along with heuristic functions. These algorithms, combined with the heuristic functions, form a powerful framework for decision-making in game playing scenarios.

Used heuristics

In our game playing system, we have incorporated several heuristic functions that contribute to the decision-making process. Let's discuss three specific heuristics: Mobility, Coin Parity, and Corners Captured.

1. Mobility: The Mobility heuristic measures the number of legal moves available to a player. It captures the notion that having more options at a given state increases the player's flexibility and potential for strategic maneuvers. By considering the difference in mobility between the AI agent and the opponent, we can assess the relative advantage in terms of available moves. Maximizing mobility is often desirable as it provides more opportunities to control the game and respond to changing circumstances.
2. Coin Parity: The Coin Parity heuristic focuses on the difference in the number of tiles controlled by the AI agent and the opponent. It calculates the coin differential, which represents the disparity in the number of pieces on the board. This heuristic promotes the idea of gaining and maintaining control over a greater number of tiles than the opponent. A positive coin parity indicates that the AI agent has a numerical advantage, which can lead to stronger positions, greater influence over the game, and potentially higher chances of success.
3. Corners Captured: The Corners Captured heuristic evaluates the number of corner positions occupied by each player. Corners are often considered strategically important in many games, as they provide stability, control over edges, and potential for expansion. By assigning a higher value to corners captured by the AI agent and a lower value to corners controlled by the opponent, the heuristic incentivizes the AI agent to prioritize capturing and protecting corners. Acquiring corners can enhance the AI agent's positional advantage, limit the opponent's options, and increase the potential for long-term control of the game.

Benefits of using heuristics:

- Efficient Evaluation: Heuristics allow for a quick assessment of game states without exhaustively exploring the entire game tree. By considering specific aspects of the game, heuristics provide an estimate of the desirability or utility of a state, allowing the AI agent to make informed decisions more efficiently.
- Domain Knowledge: Heuristics incorporate domain-specific knowledge about the game and its strategies. By encoding this knowledge into the heuristics, the AI agent can leverage expert insights and prioritize moves that align with winning strategies or advantageous positions.
- Focus on Relevant Factors: Heuristics enable the AI agent to focus on relevant factors that contribute to the game's outcome. By considering specific heuristics like Mobility, Coin Parity, and Corners Captured, the AI agent can prioritize aspects of the game that are crucial for success, leading to more effective decision-making.
- Flexibility and Adaptability: Heuristics can be adjusted and weighted based on the game dynamics, player preferences, or strategic considerations. This flexibility allows the AI agent to adapt its decision-making process to different scenarios and playstyles, enhancing its ability to respond to varying conditions.

By utilizing heuristics such as Mobility, Coin Parity, and Corners Captured, our game playing system can assess game states, prioritize moves, and make strategic decisions. These heuristics, along with the minimax algorithm and alpha-beta pruning, form a comprehensive framework for our AI agent to excel in game playing scenarios and provide a challenging and engaging experience for players.

Supported features

- Bots: Test as much as you like, but each bot typically beats the one below.
  - The level 0 bot is different. It chooses a move randomly and won't skip unless forced to.
  - The level 10 bot will be a bit slow, but level 8 will almost certainly beat you most of the time.
  - depends on combination of heuristics (Mobility, Coin Parity, Corners Captured, Stability)
- Undo and Skip can be deactivated (in real life they don't exist)
  - If skip is off, the Bots will avoid skipping unless necessary.
  - If skip is on, Bots will skip if it's strategic, but only a few times in a row before getting bored and making a move.
  - If skip is off, it turns back on temporarily if you have no available moves.
  - After "game over", Undo button resets the board.
- Scorekeepers show current scores.
- Progress bar with animation show current scores and game progress.
- Quit button is always active unless a Bot is thinking.

Maximum difficulty level supported

Our game playing system supports a maximum difficulty level of 10. The difficulty level is designed to reflect the depth of search in the game tree. As the difficulty level increases, the AI agent explores the game tree to a greater depth, considering more future moves and potential game states. This deeper search allows the AI agent to make more informed and strategic decisions.

However, it's important to note that beyond a difficulty level of 10, the time required to make a decision becomes significantly longer and may not be visibly acceptable for users. Balancing computational resources and user experience is crucial in game playing scenarios, and therefore, we have set a practical limit of 10 difficulty levels.

Level hard to play against the AI

After **level 7** will almost certainly beat you most of the time, and it becomes very hard to play against the AI

Video Link

Link: https://youtu.be/kpjs2XXbc20