Mancala

Serop Kelkelian (888119963), Sydney Sukhabut (888512910),
Gautham Vegeraju (886713619), Cyrus Traeh Baybay (887636819)

1. Problem

In this project, we were tasked with implementing an AI player into the board game Mancala. Mancala is a deterministic 2-player game, where players take turns selecting marbles to spread across the board while trying to collect the most seeds in their assigned stockpile. The game ends when there are no marbles left on one player's side of the board. At this time, the player with more marbles in their stockpile wins. Many implementations of Mancala require two players playing locally or online, which is why our implementation allows users to play against AI.

2. Approach

We created an alpha-beta pruning algorithm for the game Mancala. The alpha-beta pruning algorithm is used in decision-making and game theory. This algorithm quickly eliminates branches of the search tree that are unlikely to lead to a good move. This results in a more efficient and faster game-playing algorithm. With an efficient and faster game-playing algorithm, more moves can be evaluated at a faster rate which leads to the best possible decision each turn.

3. Description of the software

3.1 Programming Language

This project was written using the programming language Python. Python is the leading programming language for AI, and is very popular in game development. It’s clear syntax, flexibility and simplicity allows for group members to collaborate on code, while minimizing coding style differences and syntax errors.

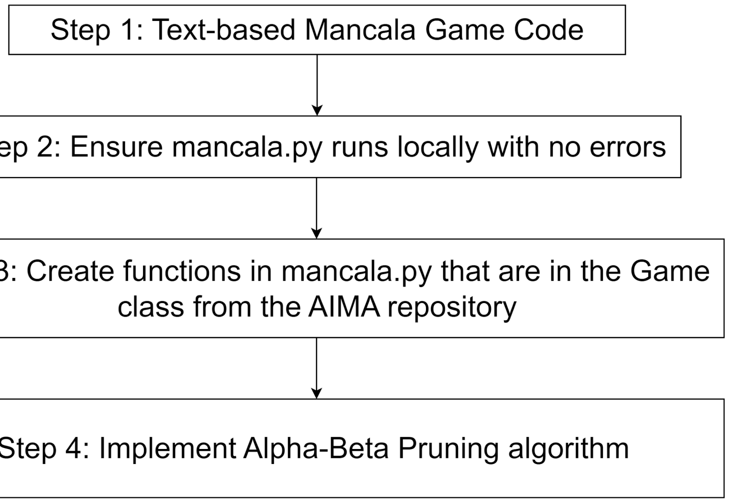
3.2 Software Components

We found an online resource to build our project off of. Al Sweigart has several books related to Python and Game Development with Python. In one of his books, he has a simple, text-based version of the Mancala game that we expanded on. The code can be found on his blog. For the AI algorithm implementation, we used the AIMA code used throughout the semester. It has many different modules that create a good foundation for implementing new games and algorithms. The AIMA Code was used to structure and test the algorithm along with the game. This code can be found on its GitHub Repository.

3.3 Code Layout

There are two Python files in this project: mancala.py and games.py. The original text-based code obtained online is located in mancala.py. This is the main file in this project as it is the file run when playing the game. The second file is games.py. This file was obtained from AIMA’s GitHub repository. It is imported by mancala.py as it contains the Game class along with the alpha-beta pruning algorithm.

3.4 System Diagram


Step 1: Text-based Mancala Game Code
Step 2: Ensure mancala.py runs locally with no errors
Step 3: Create functions in mancala.py that are in the Game class from the AIMA repository
Step 4: Implement Alpha-Beta Pruning algorithm

4. Evaluation

With the original copy of the text-based Mancala code, the game was run as player vs. player. After implementing the alpha-beta pruning algorithm, we could run alpha beta player vs regular player as well as alpha beta player vs random player. Random player is a computer player that picks its moves randomly. This is another feature that was found in the AIMA code. Overall, the gameplay is average. When having the alpha beta player play against the random player, the score is quite even. This is not desired because we expect the alpha beta algorithm to win at a much higher rate when playing against an opponent whose decisions are chosen at random. After 10 test runs, the score between the alpha beta player and the random player was 5-5.

5. Conclusion and Future Work

Overall, this project was a success. While the outcome was not exactly what we desired, there was a lot we learned in the process of implementing an AI algorithm. We learned how to refactor the existing code to properly implement the AIMA code. This will be great for future projects when implementing algorithms. Furthermore, we learned how to research and prepare for a project of this scale. Lastly, we learned how to evaluate the algorithm and assess what needs to be improved on. If given more time, further inspection and debugging into the results function in mancala would be a key factor in improving the win-loss ratio for the alpha beta player.

6. References

Existing Mancala Code:
https://inventwithpython.com/bigbookpython/project43.html

AIMA GitHub Repository Code:
https://github.com/aimacode/aima-python