
# AI PROJECT REPORT

(Intelligent Mancala Game)

**Submitted by:**

Khaled Mohamed Refaat……1600506

Ahmed Mohamed Ibrahim Abdelfattah…….1600154

Ahmed Ehab Mohamed Manaa……1600044

Ghada Ahmed Abd El-kader……1600950

Mai Suody Abd El-ghany Suody……1601601

Cairo, Egypt

## Introduction:

Mancala is a count-and-capture board game. The playing equipment can be as simple as stones or seeds as the playing pieces and holes dug in soil or boards with holes as holding cups. The object of the game is to capture the most stones in your Mancola. The game ends when all six spaces on one side of the board are empty and the player with the most pieces in his Mancola wins.

## Project Description:

The goal of this project is to implement an intelligent Mancala game -using the Minimax algorithm with alpha-beta pruning- that a human player can play against it.
We managed to make a console-based game that support the following features: playing with or without stealing, various difficulty levels (easy, medium, hard), game saving and loading.

## Implementation Details:

First thing first, a brief user guide for our game:

- First, we open our console-based game and a screen will appear asking about if we want to load a previous game and continue playing or want to start a new game. Then asking about if we want to start first or let the AI have that honor (XD), then asking whether we want to play with or without stealing and finally asking to choose between various difficulty levels (“e” for easy, “m” for medium and “h” for hard) and voila! The game is ready.
- Now we start playing in turns between us and AI until we finish the game or decided to quit the game and in this case the game asks if we want to save the game progress so that we can continue playing later or quit the game without saving it.
- All the previous questions answers are simply “y” for “yes” or “n” for “no”


For more information about our project:

https://youtu.be/STpGbts_5CI

Now, let’s get digging into our implementation…….

Our project consists of five main functions (print_board, build_tree, play, empty_pocket and main) and a separate class for building tree nodes. So, let's talk about each one in details:

- class Node:
  - this class is used for building tree nodes and their children with their alpha & beta values and also used for determining whether the parent nodes are maximizers or minimizers.
  - With each call of this class, it make an instance for a new node in the searching tree and then traverse the tree to update alpha & beta values for each node.

- def print_board(board):
  - it simply takes a list containing the number of stones in each pocket and the two Mancolas (piles) and then print the board according to these values.

- def build_tree(current_board, depth, maximizer, steal):
  - this function is simply building the searching tree knowing the depth of the tree and the root node and whether it's a maximizer or minimizer. It also assigns the alpha & beta values for each node.
  - it takes the following parameters:
    - board: to know the childs of each node.
    - depth: the depth of the tree and this is used to determine the difficulty of the game.
    - Maximizer or minimizer: to know whether the root is maximizer or minimizer, and this is determined using True if maximizer or False if minimizer.
    - With or without stealing: this is determined using True in case of stealing or False in case of without stealing.

- def empty_pocket(board, index, steal):
  - this function is called with each play, and it's simply removes the stones in the selected pocket and deposit one stone at a time into neighboring pockets going counter-clockwise until the stones run out. If a player encounters their own store, a stone is deposited in it. If there are enough stones to go past the player's own Mancola, stones are deposited continuing on the other side's pockets. However, if they encounter the other player's Mancola, that Mancola is skipped over.
  - when the "steal" parameter is "True", If the last stone is placed in an empty pocket on the player's own side, the player takes this stone as well as the other player's stones across from the empty pocket landed in, and places them in their own Mancola. And if the "steal" parameter is "False" the stones won't be placed in player Mancola.

- def play(board, index, ai, steal):
  - this function is called with each play, and it takes "True" as a parameter if the AI is the one who play or "False" if the human player is the one who play. And according to that it calls the empty_pocket() function, organize the board and state whether the game is over or not according to the number of stones in pockets.

- main():
  - this function is the backbone of the project as it calls all the implemented functions and state their parameters based on the rules the player wants to make for the game.

**Team members roles:**

| | |
|---|---|
| Khaled Mohamed Refaat | Node class |
| Ahmed Mohamed Ibrahim | Alternative turns |
| Mai Suody Abd El-ghany Suody | build tree |
| Ghada Ahmed Abd El-kader | Save and load game |
| Ahmed Ehab Mohamed Manaa | Print board |

**Appendix:**

- illustration video:
  https://youtu.be/STpGbts_5CI

- Github repo:
  https://github.com/Khalidm98/mancala