Assignment 2: Play FAUhalma

AI-1 Systems Project (Winter Semester 23/24)

Implementing Chinese Checkers Using Adversarial Search in AI

Introduction to Adversarial Search:

Adversarial search is a method employed in scenarios where multiple agents are planning or making decisions in opposition to each other. This contrasts with single-agent search algorithms, where only one agent is searching for a solution. In multi-agent environments, such as in games, each agent must consider not only their own actions but also the actions of others and how these actions impact their own strategy.

Game Theory in AI:

Games in AI are modeled as search problems and are solved using heuristic evaluation functions. These games can be classified based on two criteria: the type of information available (Perfect vs. Imperfect) and the nature of gameplay (Deterministic vs. Chance Moves).

- Perfect Information and Deterministic Games:
  Examples: Chess, Checkers, Go, Othello
  Characteristics: Players have complete information about the game state and all players' moves are visible. There is no element of chance or randomness in the gameplay.
- Imperfect Information and Chance Games:
  Examples: Bridge, Poker, Scrabble
  Characteristics: Players do not have complete information about the game state, and there may be elements of chance or randomness.

Chinese Checkers:

Chinese Checkers falls under the category of Perfect Information and Deterministic games. In this context, players have complete visibility of the board and there is no randomness in gameplay. The challenge lies in anticipating and countering the moves of the opponents.

Adversarial Search Algorithms:

- Minimax Algorithm:
  Nature: A recursive or backtracking algorithm used in decision-making and game theory.
  Purpose: To find the optimal move for a player, assuming the opponent is also playing optimally.
- Alpha-Beta Pruning:
  Nature: An enhancement of the minimax algorithm.
  Function: Improves efficiency by pruning branches in the search tree that do not affect the final decision. It achieves this through two parameters:
  - Alpha: The best choice found at any point for the maximizer (initially $-\infty$).
  - Beta: The best choice found at any point for the minimizer (initially $+\infty$).

Implementation and Execution:

To implement Chinese Checkers using these algorithms, code should be structured to integrate the adversarial search strategy. For executing the game, specify the target .json file in the 'currentgame' variable in the main section of the code.

The '`play_game`' function, continues the game as player A, first get all possible moves for player A ,then uses '`alpha_beta_pruning_3_player`' to find a score for each move and finally find the best move amount possible ones.

The code that has been implemented demonstrates proficiency in winning simpler games; however, it faces challenges when confronted with more complex and demanding scenarios