**EE 562 Assignment 3: Kalah Game**

Ray Chen  
Department of Electrical and Computer Engineering, University of Washington

**1 Description**

The game board consists of 12 holes for each player and two larger pits, one at each end of the board. Initially, each of the 12 holes on a player's side contains 6 stones. The two scoring pits are left empty. Players take turns, with the player whose turn it is choosing one of their holes and distributing its stones, one by one, counterclockwise into all houses, including opponent's houses but not scoring pits. The player who has collected the most stones in their scoring pit at the end of the game wins.

**2 Heuristic Function**

**2.1 Simple Heuristic Function**

**A black background with white text

Description automatically generated**The heuristic function calculates the difference between players. Here's the heuristic function:

**2.1.1 Justification**

The heuristic function is designed to provide a rough estimate of the current game state's desirability for player A relative to player B. It focuses on the difference in the number of stones in their respective stores.

**2.1.2 Illustration with example moves and explanations**

Initial Game State:

Player A's pits: [4, 3, 2, 1, 0, 5]; Player A's score: 0

Player B's pits: [5, 4, 3, 2, 1, 0]; Player B's score: 0

Move 1: Player A chooses to move stones from their pit at index 0 (with 4 stones) to the right.

Updated Game State:

Player A's pits: [0, 4, 3, 2, 1, 5]; Player A's score: 0

Player B's pits: [5, 4, 3, 2, 1, 0]; Player B's score: 0

Calculate the heuristic value in Move 1:

heuristic\_value = state.a\_fin - state.b\_fin = 0 - 0 = 0

Explanation: The heuristic value is 0, which indicates that the game state is currently evenly balanced between players A and B. The function didn't favor either player in this case.

Move 2: Player A chooses to move stones from their pit at index 5 (with 5 stones) to the right.

Updated Game State:

Player A's pits: [1, 4, 3, 2, 1, 0]; Player A's score: 0

Player B's pits: [5, 4, 3, 2, 1, 0]; Player B's score: 0

Calculate the heuristic value in Move 2:

heuristic\_value = state.a\_fin - state.b\_fin = 0 - 0 = 0

Explanation: The heuristic value is still 0, indicating that the game state remains evenly balanced between players A and B. The function didn't favor either player in this case either.

**2.1.3 Summary**

In these examples, the heuristic function didn't favor any particular move because both moves resulted in the same state of balance. However, as the game progresses and the difference between the stones in the stores changes, the heuristic function can indicating an advantage for the player with a higher score count.

**2.2 Complex Heuristic Function**

**A computer screen with text and numbers

Description automatically generated**The advanced version of heuristic function. Here's the heuristic function:

**2.2.1 Justification**

Stone Count Difference: This factor accounts for the immediate board state and provides an advantage to the player with more stones, reflecting a potentially better position.

Store Count Difference: This factor encourages the AI to prioritize moving stones into its store, as a positive difference favors the player with more stones in their store, indicating an advantage.

Empty Pits Difference: This factor rewards the AI for creating empty pits on its side.

**2.2.2 Illustration with example moves and explanations**

Initial Game State:

Player A's pits: [4, 3, 2, 1, 0, 5]; Player A's score: 0

Player B's pits: [5, 4, 3, 2, 1, 0]; Player B's score: 0

Move 1: Player A chooses to move stones from their pit at index 0 (with 4 stones) to the right.

Updated Game State:

Player A's pits: [0, 4, 3, 2, 1, 5]; Player A's score: 0

Player B's pits: [5, 4, 3, 2, 1, 0]; Player B's score: 0

Calculate the heuristic value in Move 1:

Stone Count Difference: (0 + 4 + 3 + 2 + 1 + 5) - (5 + 4 + 3 + 2 + 1 + 0) = 18 - 15 = 3

Store Count Difference: 0 - 0 = 0

Empty Pits Difference: (6 - 6) = 0

heuristic\_value = (1.0 \* 3) + (1.5 \* 0) + (0.8 \* 0) = 3.0

Explanation:

The heuristic value is 3.0, indicating that Player A's position is advantageous after moving. The AI chose this move because it increased stone count on its side and didn't change the store count or empty pits.

Move 2: Player A chooses to move stones from their pit at index 5 (with 5 stones) to the right.

Updated Game State:

Player A's pits: [1, 4, 3, 2, 1, 0]; Player A's score: 0

Player B's pits: [5, 4, 3, 2, 1, 0]; Player B's score: 0

Calculate the heuristic value in Move 2:

Stone Count Difference: (1 + 4 + 3 + 2 + 1 + 0) - (5 + 4 + 3 + 2 + 1 + 0) = 11 - 15 = -4

Store Count Difference: 0 - 0 = 0

Empty Pits Difference: (6 - 6) = 0

heuristic\_value = (1.0 \* -4) + (1.5 \* 0) + (0.8 \* 0) = -4.0

Explanation:

The heuristic value is -4.0, indicating that Player B's position is advantageous after moving. The AI chose this move because it reduced the stone count on Player A's side and maintained an even store count.

**2.2.2 Summary**

The heuristic function evaluates game states by considering the difference in stone counts, store counts, and empty pits between players. It assigns higher values to states that are more advantageous for the AI player, guiding the AI to make strategic moves that improve its position in the game.

**3 Result**

**3.1 Time performance**

The game is preformed with simple heuristic function locally on MacBook Pro with M1 Pro chip with 10‑core CPU. The time is average of 10 steps in millisecond.

|  |  |  |
| --- | --- | --- |
| Depth | Average Time (ms) | |
| 1 | 0.0848 |
| 2 | 0.1678 |
| 3 | 0.4011 |
| 4 | 1.1622 |
| 5 | 2.4387 |
| 6 | 5.7709 |
| 7 | 16.155 |
| 8 | 28.483 |
| 9 | 35.262 |
| 10 | 74.457 |

**3.2 Win-Rate performance**

The game is played out in three distinct scenarios, each repeated five times against a human player: one with a simple heuristic function, another with a complex heuristic function, and without heuristic function.

|  |  |
| --- | --- |
| Method | Win Rate against Human |
| Without Heuristic Function | 40% |
| Simple Heuristic Function | 80% |
| Complex Heuristic Function | 80% |

Impact on Performance:

* The complex heuristic is more powerful and versatile.
* It considers a wider range of factors, allowing the AI to make more strategic decisions.
* The AI can prioritize capturing stones, maintaining store count advantages, and creating opportunities for future moves.
* With this heuristic, the AI is likely to perform better in terms of winning games and making optimal moves.

In summary, the choice of heuristic function has a significant impact on the performance of the AI algorithm in the Kalah game. The complex heuristic function provides a more comprehensive evaluation of game, leading to more strategic decision-making by the AI. As a result, the algorithm using the complex heuristic is expected to outperform the one using the simple heuristic in terms of game outcomes and move selection.