Red-Blue Nim Game Implementation Overview

1. Introduction

Overview of the Red-Blue Nim Game:
The Red-Blue Nim Game is a complex strategy game for two players in which players take turns pulling marbles from two different piles—one blue and one red. There are two different versions of the game:

- **Standard Version:** In the event that players are forced to make a move that exhausts any pile, they will lose.

- **Misère Version:** On the other hand, players win if they can force their opponent to take a move that results in any pile being nullified.

Objectives and Goals of Implementing the Game in Python: The main goal is to show how to apply optimal search techniques, complex decision-making algorithms, and game theory ideas inside a computational framework. The Red-Blue Nim Game's algorithmic complexities and strategic dynamics are demonstrated using Python.

2. Game Rules

2.1 Standard Version: The strategic goal in this variation is to prevent a move from leading to a pile's `exhaustion`. The player who is compelled to make a move that eliminates a pile is declared the winner of the round.

2.2 Misère Version: Unlike the Standard version, the goal is to force the other player to make a move that results in a pile. The winner is the player who executes the move effectively.

2.3 Scoring:

- Every red pile that remains adds two points to the overall score.
- Every blue pile that remains adds up to three points towards the overall score.

3. Command Line Usage

3.1 Invocation: Following Command Line is used to start the game

```
python red_blue_nim_gui.py 5 3 standard human 3
```

3.2 Parameters:

- <num-red>: The initial count of red marbles.
- <num-blue>: The initial count of blue marbles.
- <version>: The game variant, either 'standard' or 'misere' (default being 'standard').
- <first-player>: The player who initiates the game, either 'computer' or 'human' (default being 'human').
- <depth>: The depth of the search tree for the AI's decision-making (optional, default being 3).

4. Game Flow

4.1 Turn Order: The way the game is played is with human and machine players taking turns. During their turn, each player is allowed to remove one or two marbles of the same turn.

4.2 Human Move: The player who is human is prompted to enter their move. The software refreshes the game state and verifies the input for legality. A message stating the winner and the final score is shown if the game reaches a terminal stage.

4.3 Computer Move: The computer determines the best course of action by utilizing the Minimax algorithm in conjunction with Alpha-Beta Pruning. Based on the present state of the game, the AI assesses every move that could be made and chooses the one that will maximize its advantage.

5. Minimax Algorithm

5.1 Overview: A framework for strategic decision-making, the Minimax algorithm assesses possible movements to determine which one best maximizes the player's advantage while minimizes the potential gains for the opponent.

5.2 Alpha-Beta Pruning: An optimization method called alpha-beta pruning is used with the Minimax algorithm. By removing the need to investigate search tree branches that have little bearing on the outcome, it improves computing efficiency.

5.3 Move Ordering (Standard): The order of move preferences for the Standard version is as follows:

- 2 red marbles have been removed.
- 2 blue marbles have been removed.
- Elimination of one red marble.
- One blue marble removed.

5.4 Move Ordering (Misère):

The preferred order of moves for the Misère variant is as follows:

- Removal of one blue stone.
- Removal of one red marble.
- 2 blue marbles have been removed.
- 2 red marbles have been removed.

6. Depth Limited Search (Extra Credit)

6.1 Purpose: By limiting the depth of the search tree and thereby optimizing computational resources, depth-limited search aims to speed up the AI's decision-making process.

6.2 Evaluation Function: The evaluation function evaluates non-terminal game states using a heuristic measure. It computes the strategic advantage based on the player's current location and the number of marbles left.

6.3 Extra Credit Details:A complete description of the logic behind its evaluation function and how it affects the AI's strategic decisions is needed. The function checks game states in order to direct the AI's choice of moves, improving the efficiency of its decision-making.

7. End of Game

7.1 Game Over Conditions: The game ends when one of the players makes a move that ends up eliminating either pile, as per the rules of the particular game type.

7.2 Scoring Calculation: The ultimate score is calculated by adding up the scores for every red and blue marble that remain at the end of the game, which is two for each red and three for each blue marble.

7.3 Final Score and Winner:

Final Score: Based on the quantity of marbles left at the end of the game, a decision was made.

Winner Determination:

- **Standard Version:** The winner is the person who did not make the move that caused the pile to empty.
- **Misère Version:** The winner is the person who made the move that caused the pile to empty.

○

8. Implementation Details

8.1 Modules:

- **Command-line Parsing:** Manages input parameters and initializes the game configuration.
- **Game Mechanics:** Executes game rules and updates in the game state.
- **Human and Computer Moves:** Oversees player inputs and AI decision-making processes.
- **AI Decision-Making with Minimax and Alpha-Beta Pruning:**includes the logic behind the Minimax and Alpha-Beta Pruning algorithms.

9. Demonstration

9.1 Walkthrough:

1- Initial State:

- Red marbles: 5
- Blue marbles: 5
- Version: Standard
- First player: Human

2- **Human Move:**

- **Prompt:** "Your move (e.g., '1 red' or '2 blue'):"
- **Input:** "1 red"
- **Updated State:** Red marbles: 4, Blue marbles: 5

3- **Computer Move:**

- The AI evaluates the state and selects the optimal move.
- **Updated State:** Red marbles: 4, Blue marbles: 3

4- Game Continues:

- Switches turns every time a move causes the pile to empty.

5- **Game Over:**

- **Standard Version:** The player who did not deplete a pile wins.
- **Misère Version:** The player who depleted a pile wins.
- **Final Score:** Computed based on remaining marbles.