AI vs Human Game - Design and Analysis

Emir Kantul (27041)

May 21, 2023

1 Introduction

In this assignment, we will design and analyze an AI vs Human game. The game is called "Slants" and is played on a board consisting of NxN slants, which corresponds to a slightly larger (N+1)x(N+1) grid of intersections. The game is zero-sum, meaning that the combined score of both players is always zero. Each successful move by a player results in that player gaining points while the opponent loses an equivalent amount.

2 Game Components

2.1 Slant Representations

Each slant on the board can be in one of three possible states: empty, filled by Player 1, or filled by Player 2. We represent these states using the following symbols:

- 0: Represents an empty slant.
- 1: Represents a slant filled by Player 1 ('/').
- 2: Represents a slant filled by Player 2 ('\').

2.2 Board Representations

The game board is made up of intersections, which can be either empty or have a number on them. We use the following notations for board representations:

- '*': Denotes an empty intersection.
- Any other number: Represents the value assigned to a filled intersection.

2.3 Players

The game involves two players:

1. Player 1: Represents the human participant.
2. Player 2: Represents the AI opponent.

2.4 Game States

The state of the game at any moment is defined by the following components:

- Board Intersections: The status of the intersections on the board.
- Slants: The configuration of slants on the board.
- Turn: The current turn, indicating which player is to move.
- Scores: The scores of the players.

2.5 Initial State

The game starts with the following initial state:

- Board: Empty board with no slants.
- Slants: All slants set to 0 (empty).
- Turn: Player 1 (human player).
- Scores: Both players have a score of 0.

2.6 Terminal State

The game reaches a terminal state when all possible moves have been made, i.e., all slants are placed on the board. At this point, the game is over, and the scores are evaluated to determine the winner.

2.7 State Transition Function

The state transition function governs the rules of the game. Given the current state, it returns a new state where the next player has placed a slant. If the placement of the slant completes an intersection, the player who placed the slant receives a score increase equivalent to the number on the intersection, while the opposing player's score decreases by the same amount.

2.8 Payoff Function

Once the game reaches its terminal state, the payoff function calculates the total points of each player to determine the winner. The payoff is calculated by subtracting the second player's score from the first player's score. As the game is zero-sum, the combined score of both players is always zero.

3 Game Modes

The AI vs Human game supports different game modes and provides various analytics metrics to evaluate the game's performance.

3.1 AI vs AI Mode

In this mode, the game is played between two AI players. The minimax algorithm with alpha-beta pruning is used to determine the best move for each AI player. The game supports the option to record and save the moves made by each AI player to a text file.

3.2 AI vs AI Mode with Moves Recording

In this mode, the game is similar to the AI vs AI mode, but the moves made by each AI player are recorded and saved to a text file. This mode allows for a detailed analysis of the game and the ability to replay the game moves later.

3.3 Human vs AI Mode

In this mode, the game is played between a human player and an AI player. The human player interacts with the game by entering their moves in the format 'x y', where 'x' and 'y' represent the coordinates of the slant on the board. The AI player uses the minimax algorithm to determine its moves. Similar to the AI vs AI mode, the moves made by both players can be recorded and saved to a text file.

4 Analytics Metrics

The game provides various analytics metrics to evaluate its performance. These metrics include:

- Test Files: The game can be run with different test files that define the initial state of the game.
- Board Size: The size of the game board.
- Grid Size: The size of the slant grid.
- Possible States: The total number of possible states in the game.
- Max. Possible Search Depth: The maximum search depth allowed for the minimax algorithm.
- Current Search Depth: The current search depth used by the minimax algorithm.
- Time Taken: The time taken by the game to complete.
- Final Scores: The scores of the players at the end of the game.

4.1 Trade-offs: Depth Limit and Speed

In the AI vs AI mode, the depth limit parameter plays a crucial role in balancing the search depth and the speed of the game. A higher depth limit allows for a more thorough exploration of the game tree and potentially better moves. However, a deeper search also requires more computational resources and time. On the other hand, a lower depth limit reduces the search space and speeds up the decision-making process but may result in suboptimal moves.

Choosing the appropriate depth limit involves considering the trade-offs between the game's complexity, the available computational resources, and the desired response time. In the implemented game, the depth limit is set to a predefined value (MAX_DEPTH), which can be adjusted to find a suitable balance between the search depth and the game's speed.

4.2 Efficiency with Alpha-Beta Pruning, Heuristic Evaluation, and Move Ordering

The minimax algorithm used in the AI vs AI mode can be further optimized with techniques like alpha-beta pruning, heuristic evaluation, and move ordering to improve its efficiency.

4.2.1 Alpha-Beta Pruning

Alpha-beta pruning is a technique used to reduce the number of nodes evaluated by the minimax algorithm. It allows pruning branches of the game tree that are guaranteed to be worse than the already evaluated branches, effectively reducing the search space. By eliminating these unnecessary evaluations, alpha-beta pruning significantly improves the algorithm's efficiency, especially in games with large search spaces.

In the implemented game, the minimax algorithm is enhanced with alpha-beta pruning, which greatly reduces the number of nodes explored during the search process, resulting in faster decision-making and improved performance.

4.2.2 Heuristic Evaluation

Heuristic evaluation is a technique used to estimate the desirability of a game state without exhaustively searching all possible moves. It assigns a heuristic value to each game state, indicating the expected utility or advantage for the player. By using heuristic evaluation, the minimax algorithm can make informed decisions based on partial information and avoid unnecessary computations.

In the implemented game, the payoff function serves as a simple heuristic evaluation function. It calculates the total points of each player at the terminal state to determine the winner. Although basic, this heuristic evaluation provides a reasonable estimate of the game's outcome and helps guide the minimax algorithm's search.

4.2.3 Move Ordering

Move ordering is a technique used to prioritize the evaluation of moves during the minimax search. By considering more promising moves earlier in the search process, move ordering can improve the effectiveness of alpha-beta pruning. Promising moves are typically identified based on heuristics or previous search results.

In the implemented game, move ordering is not explicitly implemented. However, the use of alpha-beta pruning naturally prioritizes more advantageous moves, as they are more likely to lead to cutoffs and further reduce the search space. This implicit move ordering contributes to the efficiency of the minimax algorithm.

5 AI vs AI Metrics

The following table presents the performance metrics for the AI vs AI mode with moves recording:

Test File	Board Size	Grid Size	Possible States	Max. Possible Search Depth
test0.txt	3	2	16	4
test1.txt	5	4	65536	16
test2.txt	5	4	65536	16
test3.txt	7	6	68719476736	36
test4.txt	9	8	18446744073709551616	64
test5.txt	11	10	1267650600228229401496703205376	100

Table 1: AI vs AI Mode Metrics

The metrics include the test file name, board size, grid size, the total number of possible states, the maximum possible search depth, the current search depth, the time taken to complete the game, and the final scores of the players. These metrics provide insights into the game's complexity, search space, and computational requirements.

6 Conclusion

The AI vs Human game "Slants" provides an engaging and challenging experience for players. It offers different game modes, including AI vs AI and Human vs AI, along with various analytics metrics to assess the game's performance. The implementation uses slant and board representations, state transition function, and payoff function to ensure fair and strategic gameplay. By analyzing the game through different test files, players can gain insights into the game's dynamics and explore different strategies.