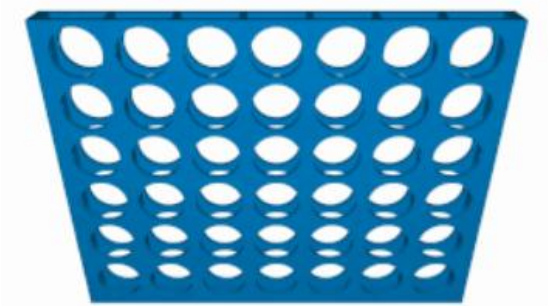
Connect-4 is a strategic two-player game where participants choose a disc colour and take turns dropping their coloured discs into a seven-column, six-row grid.



Victory is achieved by forming a line of four discs horizontally, vertically, or diagonally. Several winning strategies enhance gameplay:

a. Middle Column Placement:

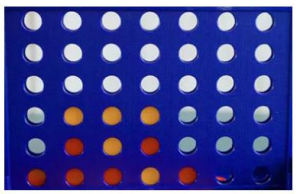
The player initiating the game benefits from placing the first disc in the middle column. This strategic move maximizes the possibilities for vertical, diagonal, and horizontal connections, totalling five potential ways to win.

b. Trapping Opponents:

To prevent losses, players strategically block their opponent's potential winning paths. For instance, placing a disc adjacent to an opponent's three-disc line disrupts their progression and protects the player from falling into traps set by the opponent.

c. "7" Formation:

Employing a "7" trap involves arranging discs to resemble the shape of a 7 on the board. This strategic move, which can be configured in various orientations, provides players with multiple directions to achieve a connect-four, adding versatility to their gameplay.



Connect-4 Implementation using Mini-Max Algorithm:

In this scenario, a user engages in a game against the computer, and the Mini-Max algorithm is employed to generate game states. Mini-Max, a backtracking algorithm widely used in decision-making and game theory, determines the optimal move for a player under the assumption that the opponent also plays optimally. Two players, the maximiser and the minimizer, aim to achieve the highest and lowest scores, respectively. A heuristic function calculates the values associated with each board state, representing the advantage of one player over the other.

Connect-4 Implementation using Alpha-Beta Pruning:

To optimize the Mini-Max algorithm, the Alpha-Beta Pruning technique is applied. Alpha-Beta Pruning involves passing two additional parameters, alpha and beta, to the Mini-Max function, reducing the number of evaluated nodes in the game tree. By introducing these parameters, the algorithm searches more efficiently, reaching greater depths in the game tree. Alpha-Beta Pruning accelerates the search process by eliminating the need to evaluate unnecessary branches when a superior move has been identified, resulting in significant computational time savings.

Strategy Used:

The AI player is chosen as the maximiser and the human as the minimiser. At every instance of the AI's turn, the minmax function is invoked to estimate the best move out of 7 possible moves

Branching factor of the game is 7
No of possible states is 3^{42} which is computationally infeasible

Approximate solutions: To avoid exponential computations, the depth of the search tree is limited to 6. If a winning move is found within this limit, it is chosen otherwise the move resulting in the heuristically best outcome is chosen.

Heuristic Function:

Zugzwang moves: A forced move in which a player has established 3 consecutive pieces horizontally, vertically or diagonally with 1 move to win, and the opponent is forced to block this setup. +900 points for AI and -900 points for human

3 consecutive pieces blocked: When 3 consecutive pieces have been established by a player but the 4 th piece is either blocked or is currently impossible to fill. +100 points for AI and -100 points for human

2 consecutive pieces: +10 points for AI and -10 points for human

Middle column advantages: For every middle column piece +5 to AI and -5 to human

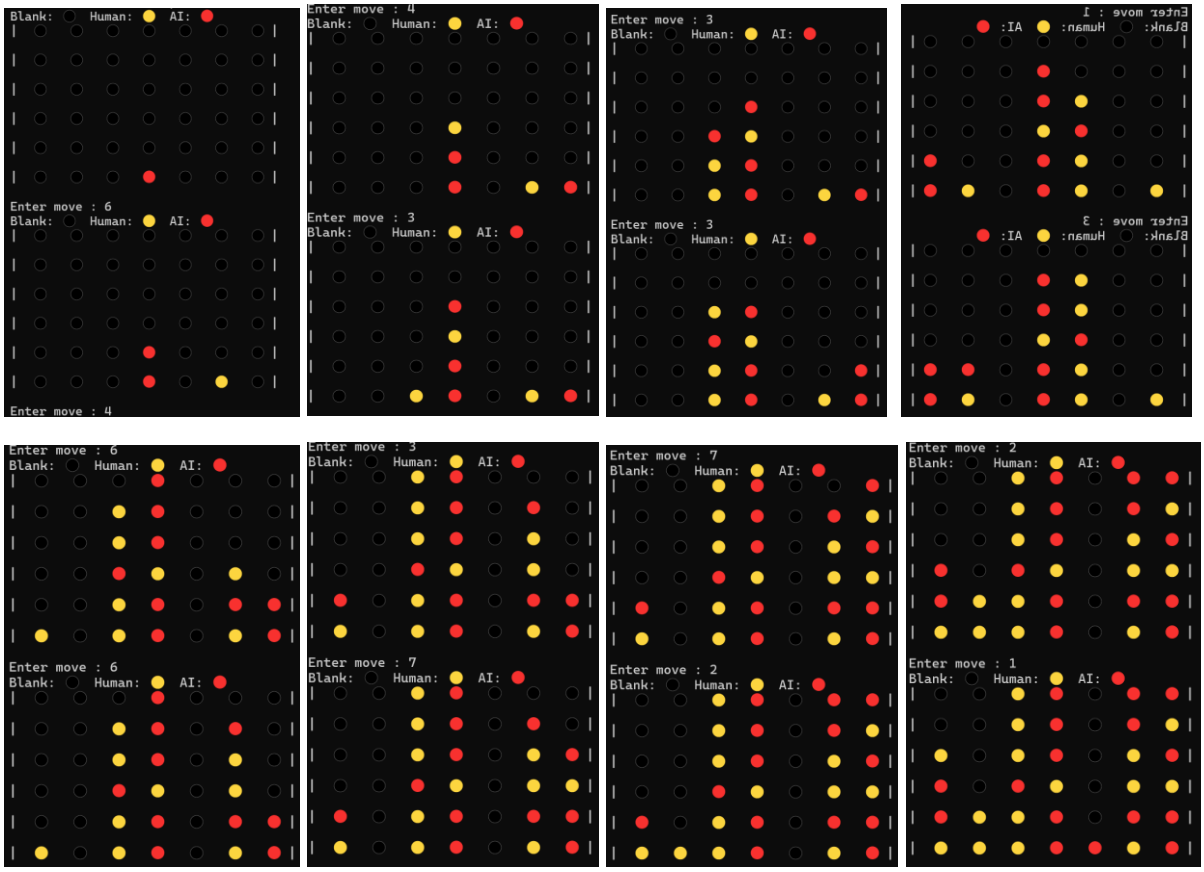
Heuristic discounting: The next player is always at an advantage because in case of a winning move existing for both, the next player will always win. So, the heuristic values are discounted. The next player gets exactly the heuristic values and the other one gets 50 percent discounting on the heuristic values.

Additional optimization:

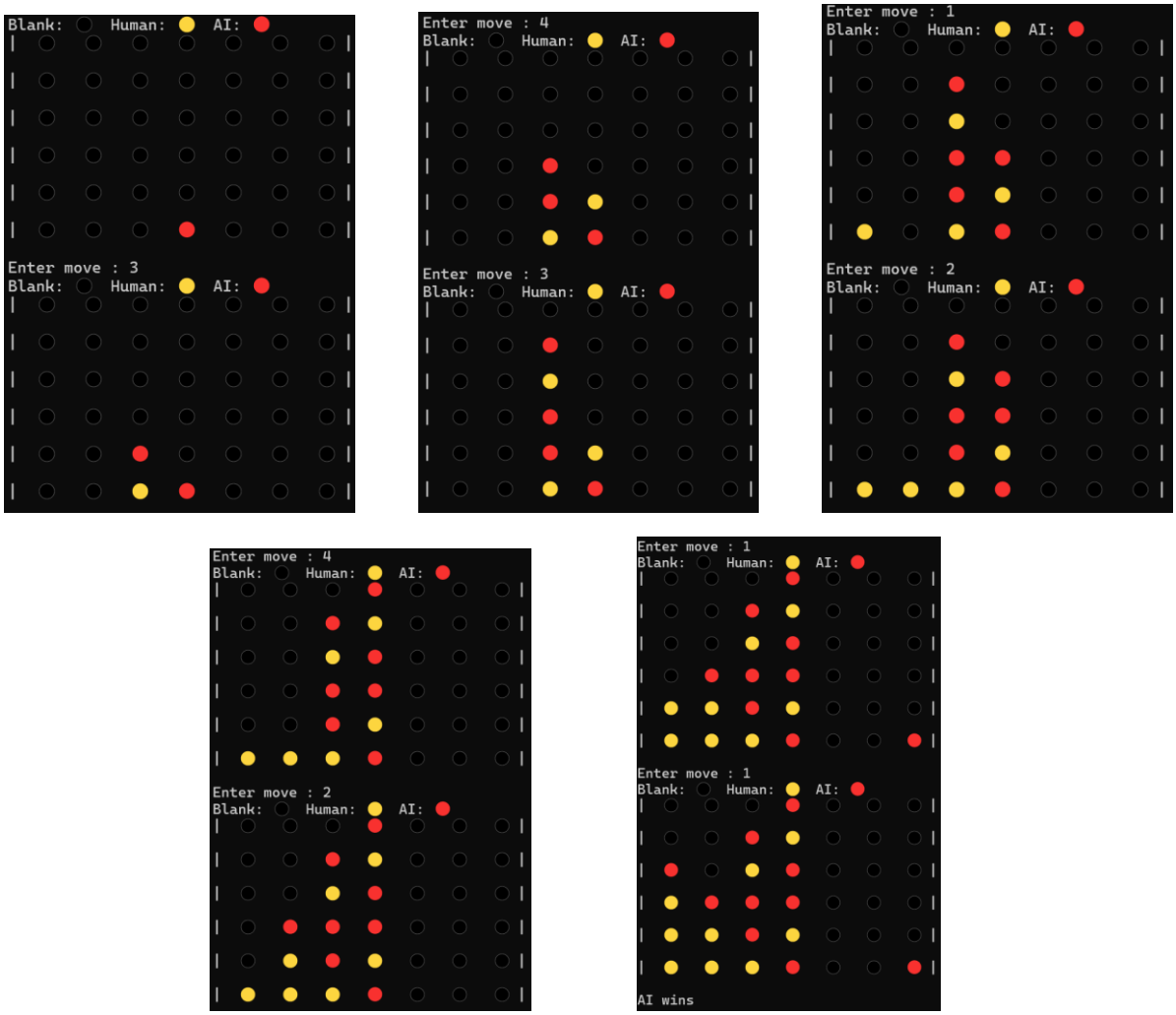
Alpha beta pruning has been used to improve search time. In this way the game tree can be searched much deeper.

Output:

Human Wins (Optimal AI of connect4.gamesolver.org is human)



AI Wins (Actual human)



Further Improvements:

- The heuristic function can be altered, to handle complicated edge cases more efficiently
- Iterative deepening minmax can be used to improve upon time complexity
- Transposition table can be used to cache the minmax values of already explored states.
- Running this simulation in a GPU can improve search time tremendously.