**ABSTRACT**

This paper demonstrates the creation, testing and performance evaluation of a game Connect-Four with the help of three state-of-the art AI algorithms namely Minimax algorithm, Expectimax Algorithm with alpha-beta pruning and Depth-Limited Search for improved performance and Monte Carlo Tree Search algorithm. Connect-Four is a familiar and well-known board game whose objective is to get four slots in a row.

**INTRODUCTION**

Besides constructing intelligent systems, Artificial Intelligence is also the endeavour to understand them. It includes an assortment of applications ranging from general purpose areas to particular tasks like proving mathematical theorems and diagnosing diseases. Game playing is one of the oldest fields of ventures in AI. General Game Playing (GGP) is the composition of AI programs’ ability to play games successfully.

Connect-Four a two-player game, where the objective of the game is to win by making four or more discs of your colour in a line horizontally, vertically or diagonally. The board is fashioned with six rows and seven columns where the discs are dropped vertically down occupying the last available slot within that respective column by taking turns. Although it is a solved game, it has 10^13 possible positions on the board making it absurd and illogical to store the moves in memory. As ardent players of board games and an inclination towards working on Artificial Intelligence, in this paper, we aim to depict and illustrate the different ways the AI can be utilized to create gameplay agents for the Connect-Four board game. Agents in AI are entities which act and directs its movements towards accomplishing goals. However, goals alone are insufficient to develop high-quality behaviour. Utility based agents comprise of a utility function that maps a state onto a real number representing the associated degree of happiness (1). For the agent to achieve the goal, it needs to be equipped with the capability of knowing the states, possible actions, transition model, goal test and path cost. The Connect-Four environment (3) comprises of properties as illustrated in figXXX.

|  |  |
| --- | --- |
| Properties of Environments | Connect-Four environment |
| Accessible vs Inaccessible | Accessible – It is completely observable since it consists of the board with constant dimensions and discs belonging to either the player or opponent. |
| Deterministic vs Non-Deterministic | Deterministic – It is considered deterministic since no random elements are involved. |
| Static vs Dynamic | Static – Since the present state cannot be altered be changed until the agent performs a move it is static. |
| Discrete vs Continuous | Discrete – Since the number of states is finite it is discrete. |
| Episodic vs Non-Episodic | **DOUBT** |

There are many approaches that can be employed to solve the Connect-Four game based on the difficulty level set for the AI. As per (2) they are random, defensive and aggressive AI. Random AI as the name suggests randomly performs an action and is easiest to beat, Defensive AI prioritizes to oppose and block the opponents moves from winning and Aggressive AI gives precedence to only winning.

Connect-Four makes a good candidate for implementing and studying search algorithms to find the best possible solution from all the possible solutions. This paper demonstrates the creation, testing and performance evaluation Connect-Four with the help of three state-of-the art AI algorithms namely Minimax algorithm, Expectimax Algorithm with alpha-beta pruning and Depth-Limited Search for improved performance and Monte Carlo Tree Search algorithm. In this paper, we aim to demonstrate a variation to the program with a mixture of ways to select the type of players to play and solve the game. The possible combinations of players are Human versus Human, Human versus Agent and Agent versus Agent. Our experiment involves five types of agent-players namely: Agent1 – Random, Agent2 – Forward checking, Agent3 – Minimax algorithm with alpha-beta pruning and depth-limited search, Agent4 – Expectimax algorithm with alpha-beta pruning and depth-limited search and Agent5 – MCTS.

Search strategies are evaluated by measuring their effectiveness of performance in solving the problem. Ideally, effectiveness of a search is calculated using the search cost associated with the time and memory consumed to obtain the solution. For the Connect-Four game, we chose to record the time and number of moves taken by both the players.

The rest of the paper is organized as follows: Section 2 mentions the Related Work on approaches used to solve the problem, Section 3 comprises of the Problem Definition and Algorithms implemented, Section 4 describes the Experimental Results obtained by the utilised methodology and lastly, Section 5 includes the Conclusions and final discussions of the main results along with an acknowledgement of future work.

In this section you will discuss possible approaches to solve the problem you are addressing, justifying your choice of the 3 you have selected to evaluate. Also, briefly introduce the approaches you are evaluating with a specific emphasis on differences and similarities to the proposed approach(es).

**RELATED WORK**

Game playing is an important area of artificial intelligence. Once the rules, permissible moves and conditions to succeed or lose the game are acknowledged search procedures can be implemented to see good moves to be generated and explored first.

This section formalises the problem you are addressing and the models used to solve it. This section should provide a technical discussion of the chosen/implemented algorithms. A pseudocode description of the algorithm(s) can also be beneficial to a clear explanation. It is also possible to provide one example that clarifies the way an algorithm works. It is important to highlight in this section the possible parameters involved in the model and their impact, as well as all the implementation choices that can impact the algorithm.

**PROBLEM DEFINITION AND ALGORITHM**

Connect-Four being a two-player mathematical 6x7 board game, it can be solved strategically using a mathematical solution. FigXXX depicts the factors that are to be defined to formulate the problem.

|  |
| --- |
| Factors – as a diagram |
| Initial State |
| States |
| Actions |
| Transition Model |
| Goal Test |
| Path Cost |

From a plethora of artificial intelligence algorithms available, a mixture of brute-force techniques and knowledge-based approaches can be utilised to solve the game. There are a few conclusions resolved from solving the game for instance, the first player has a higher probability of winning if the game is started by placing the disc in the middle column. However, this highly depends on the agent’s capacity to look ahead based on the algorithm imposed on it. The layout of the Connect-Four game and design flow of the game is shown in figVVV and figBBB respectively.

**(figures of how the empty connect 4 board game looks like and overall flow of the game as a flow chart– UI, selecting the players)**

Apart from a random and forward checking agent for naïve playing, we demonstrate and evaluate the performance of three other agents respectively equipped with three state-of-the-art artificial algorithms Minimax, Expectimax and Monte Carlo Tree Search algorithm by formulating different combinations of games to be played against each other.

**3.3 Monte Carlo Tree Search Algorithm**

Monte Carlo Tree Search (MCTS) is a deeply utilized heuristic search algorithm known for incorporating an element of randomness into the search. It was highly recommended as a search and planning structure for identifying optimal results in a given situation. Upper Confidence Bounds for Trees (UCT), a manifestation of MCTS was considered revolutionary for game-playing agents in artificial intelligence (4). The expression and pseudocode for UCT is depicted respectively (fig ggg) from which the highest value calculated is utilised in decision making models.

Where,

* *wi* – number of wins for the node after the ith move
* *ni* – number of simulations for the node after the ith move
* *Ni* – total number of simulations after the ith move run by the parent node
* *c* – the exploration parameter, chosen as for the game.

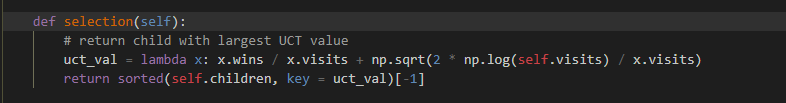


Fig ggg

The MCTS planning methodology incrementally constructs an asymmetric search tree guided in most assuring direction which is estimated iteratively and enhances its performance with an increase in the number of iterations (4). An MCTS iteration usually consists of four consecutive phases: *(1)Selection* where the tree choses the node with largest UCT value, in other words the node which possess the best chance of winning out of all, *(2)Expansion* into new child nodes of the tree after randomly choosing the node, *(3)Rollout* where simulations are conducted on the expanded nodes to review the status of the game - win, loss or tie, *(4)Backpropagation* updates the results obtained in the rollout step along the chosen path like number of wins for the selected node, total number of simulations conducted on the selected node and parent node. Fig XXX depicts a working example of Connect-Four game by following the steps mentioned for the MCTS iteration which is followed by the pseudocode of the above mentioned MCTS steps (fig hhh).

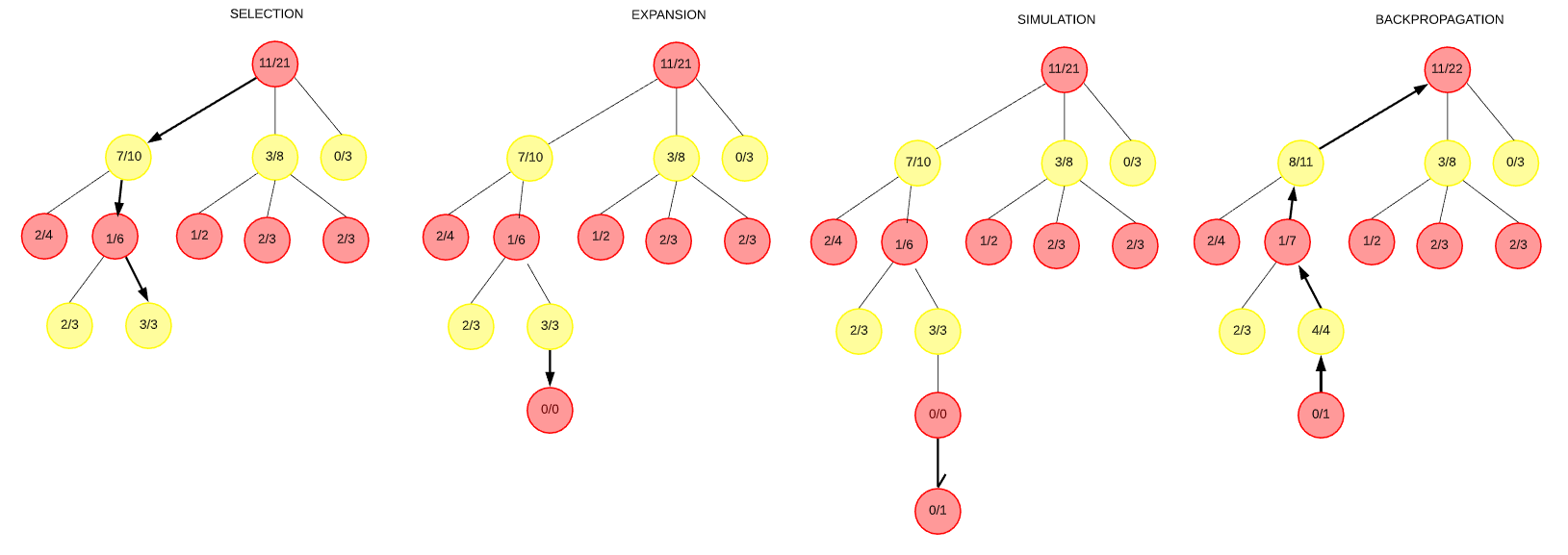


Fig XXX

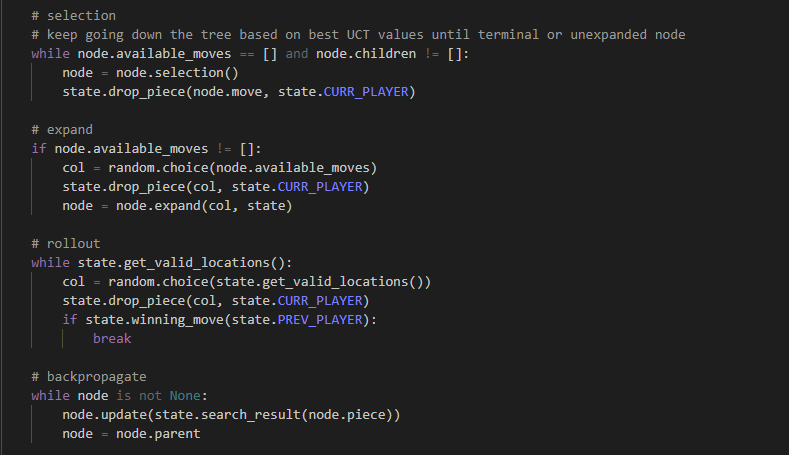


Fig hhh

**REFERENCES**

1 - Artificial Intelligence A Modern Approach

2 - Real-Time Connect 4 Game Using Artificial Intelligence

3 - <https://0xadada.pub/2003/12/15/connect-four-playing-ai-agent/>

4 – On Monte Carlo Tree Search and Reinforcement Learning