Connect 4

Joh15256

April 1, 2022

Abstract

Connect 4 is a classic board game where players battle head to head in an attempt to create four in a row with their colored pieces. They do so by taking turns dropping discs of their own color into a 7x6 grid. The first player to achieve four in a row, either horizontally, vertically, or diagonally, wins. The game was first released in February 1974, and since then there has been ample time for people to develop various strategies and approaches to optimize play in this classic game. This paper will analyze some of the prominent work that has already been done in this area, along with the different approaches used with AI algorithms.

1 The Basics

First off, let's learn a bit about how a Connect 4 AI actually works. [Pea] gives an in depth description of the basics behind a Connect 4 algorithm. Essentially, it uses a min-max algorithm, where player one's goal is to achieve minimum values, while player two's goal is to achieve maximum values. When considering a move to play, the algorithm pretends as if the move in question has actually been played. Let's say, for example, player one has just made a move, the current state of that move is unknown, so it is set at some minimum value. Now, the algorithm tries all possible follow up moves for player two, and updates the current value to the maximum value achieved from the follow up moves. If one of the follow up moves achieves a four in a row for player two, then the algorithm would return some maximum value for the move played by player one, thus making the move played by player one undesirable. The algorithm would continue to iterate through each column, generating values for each move using the same process. This process is repeated for whatever depth the AI is using in order to evaluate the strength of any given move. [Sul19] also gives a good summary of the general min-max algorithm. In short, once the algorithm completes, the AI will be left with a list of values representing the potential columns that can be played, with corresponding values representing the strength of playing in each column. Assuming the AI is player one (who plays off of the minimum values), it then chooses the column with the minimum strength value, and makes its move.

2 Improving the General Approach

Of course, the above general algorithm can be improved. It currently is simply using a breadth first search, checking every possible move at each of the 7 columns results in a time complexity of $O(7^n)$. This adds up quickly as using a small depth of only 9 can lead to $7^9 = 40{,}353{,}607$! Depth represents moves, and there are two moves per turn (one per player), thus a depth of 9 only covers 4.5 turns. Increasing the depth can quickly become a problem for most computing systems. There are several approaches for reducing the number of searches needed to traverse depths of greater value. [LTA17] recommends an alpha beta pruning and MTD(f) algorithm, and [KWH19] offers several heuristics to help make the algorithm more efficient. Let's first examine the approach taken by [LTA17].

2.1 Alpha Beta Pruning

[LTA17] explores AB pruning. Essentially, this algorithm utilizes the general min-max approach, but eliminates sub trees which can be proved irrelevant. Similar to the generic AB pruning algorithm, this Connect 4 algorithm determines which branches are not worth exploring if it has already found sufficient results prior to searching the other branch. This allows the algorithm to search at a much

greater depth, as certain branches are removed and the standard rate of 7n̂ is reduced with each pruned branch.

2.2 Heuristics

[KWH19] discusses various heuristics that can be used when developing an AI algorithm. These heuristics fall into to main categories, defensive strategies and offensive strategies. Defensive heuristics attempt to find moves which limit the opponent's ability to make four in a row, while offensive heuristics attempt to make four in a row faster than the opponent. The approach taken by [KWH19] was the aggressive approach. They use several features to create their heuristic function. For example one feature is: "A move can be made on either immediately adjacent columns → Infinity". Simply, if there is a guaranteed win by playing on either adjacent columns to make four in a row, it is assigned infinity as no other values matter. Whereas another feature simply calculates the value of a column based on its position, as in the middle columns are worth more than the outer columns. These features are worth significantly less, with values under 200. There are other features in between, ranging from 200 up to 900,000. Each feature has a specific value, and these values at the end of each turn are summed up to generate the heuristic value, which the algorithm can use to make decisions.

3 The Perfect Strategy

Does the perfect strategy exist for Connect 4? The answer is yes. There are 4,531,985,219,092 possible states that any game of Connect 4 can have. While this number is substantial, it is not impossibly large. Connect 4 has in fact been solved by John Tromp[Tro]. Building off of previous work completed by James Allen[All90] and Victor Allis[All88] in 1988, Tromp was able to generate a database of all 8-ply positions. Not only did he solve the classic 7x6 version of Connect 4, Tromp has solved many variations of grid sizes as well, ranging from 6x9 to 9x6. Despite the fact that the game is solved, the task of developing strategies and heuristics is still worth exploring, as a typical computer can not compute 4.5 trillion different positions in real-time. [HK18] goes into this explaining that Connect 4 has a maximum of 21 turns and is much shorter than more complex games such as Chess, but "[Connect 4] retains the desirable features highlighted by researchers in problem solving and decision making of chess including turn-taking and competition leading to goal-oriented positive moves (solutions) and negative problem finding experiences." Essentially, studying Connect 4 offers multiple repeated solving experiences within a relatively short period of time.

References

[All88] Victor Allis. A knowledge-based approach of connect-four. *Department of Mathematics and Computer Science Vrije Universiteit, Amsterdam, The Netherlands*, 10 1988.

[All90] James Allen. Expert play in connect-four. 1990.

[HK18] Gillian Hill and Shelly M Kemp. Connect 4: A novel paradigm to elicit positive and negative insight and search problem solving. *Frontiers in psychology*, 9, 10 2018.

[KWH19] Xiyu Kang, Yiqi Wang, and Yanrui Hu. Research on different heuristics for minimax algorithm insight from connect-4 game. *Journal of Intelligent Learning Systems and Applications*, 11:15–31, 01 2019.

[LTA17] Mardi Hardjianto Lukas Tommy and Nazori Agani. The analysis of alpha beta pruning and mtd(f) algorithm to determine the best algorithm to be implemented at connect four prototype. 2017.

[Pea] Adam Pearce. Connect 4 ai: How it works.

[Sul19] Jay Sullivan. Game ai - creating a connect 4 ai. *Jay's Journal*, 10 2019.

[Tro] John Tromp. John's connect four playground.