CS 221 Project Proposal - 3D FourPlay AI Agent

Timothy Lee (timothyl), Edwin Park (edpark), Anthony Ma (akma327)

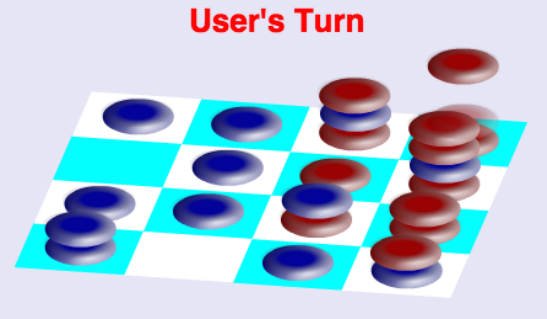
FourPlay, also commonly known as Connect Four, is a two-player board game in which each player takes a turn in dropping a colored disc into a 6-by-7 vertically suspended grid. The objective of this game is to connect four of one's own pieces next to each other horizontally, vertically, or diagonally. Given the adversarial, zero-sum game nature of our project, we will use AI algorithms to have a computer learn to play Connect Four.

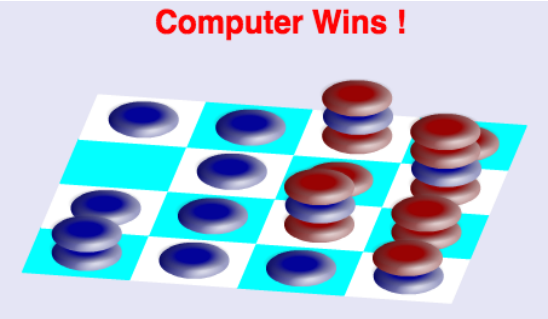
We'd like to make this game more fun and complex, so we will design our project based on a 4x4x4 3-D version of Connect Four. In this case, there is a 4-by-4 base grid, and pieces can be stacked upon each base tile to win in a similar fashion (horizontally, vertically, and diagonally in 2-D and even 3-D!).

Input: The current state of the board
Output: The next move that the model makes
Scope: Solving the 3-dimensional Connect Four game with the constraints that the 3-D grid is 4 x 4 x 4, and limited to *two* players. We hope to generalize to any *i* x *j* x *k* grid and to 3+ players within processing power constraints.

The main evaluation metric would be defined as the winning rate. However, we can also create a more comprehensive set of evaluation metrics by looking beyond the binary win vs loss paradigm by considering the extent of how well we played regardless of outcome. We will introduce further evaluation metrics, namely number of moves it takes to win the round, number of possible winning states that the computer could choose from, running total of contiguous pieces for each length [1, 2, 3, 4], number of defensive moves that prevent the opponent's contiguous chain from growing, and number of offensive moves that grow its own chains. We can also score a particular move by its influential power. We presume that growing or stopping longer chains has more influence on the state than growing or stopping short chains. Consequently, a move that grows or stops longer chains receives a greater reward.

Preliminary Data and Examples of Input and Output
The first figure shows the input of the current state of the board[1]. Output would be the computer making the optimal output is the action of placing a blue tile at (1,0) to win the game.





Baseline vs. Oracle

- Baseline Algorithm: We introduce a simple reflex based approach where given a 3D board with columns of length (0 to n) on a n x n grid base, make a move by randomly placing a piece in one of n^2 possible slots. However, if it recognizes that the current state has a possible winning move (3 contiguous pieces of the same color), it will place the tile in that position to win or block the adversary from a win.

- Oracle: We can have a Connect Four human or computer expert play to the game to determine optimal sequence of moves for maximizing winning percentages.

- The baseline algorithm does not consider the current state of the board and does not try to avoid losing or to win. Our oracle strategizes based on current state and considers future states, such that appropriate action is taken to block adversarial moves and plan / predict future moves. We will use AI techniques to implement a computer that plays reasonably well (i.e. block opponent's chain of 3, or complete a chain of 3).

Challenges

- **Large state space**: For an *n* x *n* x *n* grid, and d number of players the state space is of order O(*d*^*n*^{3}) since there are d possible tiles for each grid and there are *n*^{3} spaces overall.
- **Adversary may not operate deterministically**: Algorithms such as minimax depend on the fact that the adversary always acts to minimize the utility in the best way possible. If this is not the case, the action provided by minimax's policy may not necessarily be the best.

Expectimax and minimax with depth-limited search, alpha-beta pruning and evaluation functions for optimization. Reinforcement learning techniques such as TD learning can be used to learn the evaluation functions. Furthermore, we can utilize learning cross-validation to tune hyperparameters on the degree_of_offense and degree_of_defense to define how much you value building your own contiguous chain versus blocking off the opponent. Neural networks seem promising when a minimax player has been built, and game histories can be used to train neural networks to play FourPlay 3D.

We see that an AI model of the simple 2D version has been popularly implemented, and their works can give us insight on building an AI model for the 3D version.

References

1.) https://www.mathsisfun.com/games/connect3d.html
2.) http://roadtolarissa.com/connect-4-ai-how-it-works/
3.) http://www.computer.org/csdl/proceedings/sbrn/2002/1709/00/17090236.pdf