Steve Babcock
6/23/2017

Connect-Four Player

This project was part of an assignment in CSCI-B551 Elements of Artificial Intelligence - a course at Indiana University. A course wide competition was conducted by Dr. Tor Lattimore, and my player finished third out of around forty. Many adversarial search algorithms employ breadth first search. However, I chose a unique version of DFS. First, I order the advantage of all possible moves at a position. Then the best several moves are recursively called until a maximum depth is reached. Early in the algorithm, four moves are selected and later only two. This allows me to easily reach a depth of twelve within the time limit of three seconds. To calculate the advantage of a particular position, I sum up all the "attacks." Say a player has three tokens in a row next to an empty space. His or her advantage is considerably improved. Whereas, a player having two tokens with two empty spaces around them will see modest advantage improvement. The fault of my algorithm is that I fail to consider the orientation of attacks relative to other attacks. For example, one player has three in a row right above the other player's three in a row with the empty spaces right on top of each other. The bottom player's attack is significantly stronger. So this is a great area to improve the algorithm - perhaps by adding a wonColumns array populated by 0, 1, or 2. Before improving advantage, this array is checked first.

To run this project…

1) Open a terminal
2) CD to folder containing board.py, player.py, etc
3) type "python runGame.py" and press enter

You will see slightly altered versions of my player compete against each other. The first player has search depth of 12, but the second player is set at 10. Notice that when player one goes first, it wins. Whereas, player two will draw when going first.