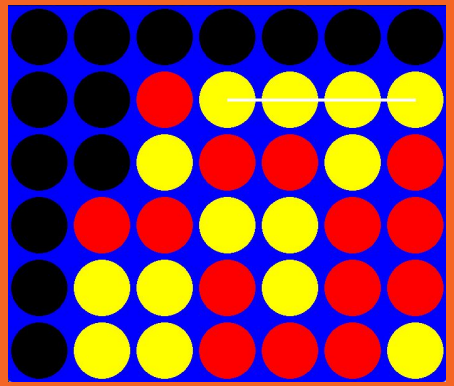


Connect 4: Artificial Intelligence Agents

Minimax and Alpha-Beta Algorithms

Tobin Joseph, Almas Amjad, Keshav Haranath • 08.26.2022

Overview

Evaluation function

Mathematical Definition

Changes in testing and fine tunning

- Auxiliary functions
- Weights calibration

Evaluation function

Strategy behind its design

Performance Optimization

- Depth calibration

Demonstrations

- minimaxAI
- alphaBetaAI

Evaluation Function - Definition

- The countInColumn
  - counts how many chips the specified player has in the specified column and gives a point for each chip. Column 3 is the middle column.
- The consecutive2 and consecutive3 functions
  - look for consecutive chips in all horizontal, vertical, and diagonal positions and give a point for each pair with a space before or after the pair adding another point if spaces on both sides
- The seven
  - looks for the 7 positions normal, reflected and inverted giving a point for each one found.
- GameOver
  - looks to see if the player has a winning move on the board giving a point if it finds one.

Evaluation Function - Definition

Basic Form

value =
w_1 * countInColumn(3, player)
+ w_2 * consecutive2(player)
+ w_3 * consecutive3(player)
+ w_4 * seven(player)
+ w_5 * gameOver(player)

Zero Sum

value =
w_1 * countInColumn(3, player)
+ w_2 * consecutive2(player)
+ w_3 * consecutive3(player)
+ w_4 * seven(player)
+ w_5 * gameOver(player)
+ w_6 * countInColumn(3, opponent)
+ w_7 * consecutive2(opponent)
+ w_8 * consecutive3(opponent)
+ w_9 * seven(opponent)
+ w_{10} * gameOver(opponent)

Changes in Testing & Fine Tuning

Auxiliary Functions

- Functions to identify certain chip combinations - example: seven
- Created new gameOver function to substitute the one from connect4 class to search everywhere instead of just from the last move

Weights Calibration

- We used different weights for some of the opponents auxiliary functions to force defensive moves
- We found better performance with the bottom of the three being a max function

Evaluation Function - Strategy

Strategy Research

- We researched different strategy recommendations
- We assessed how good of a position we are in the game based on the recommended strategies

Identify Combinations

- Playing the middle column is good
- Having 3 consecutive chips with a space to win is good
- Having 2 consecutive chips with spaces is good
- Having a 7 combination is almost a certain win

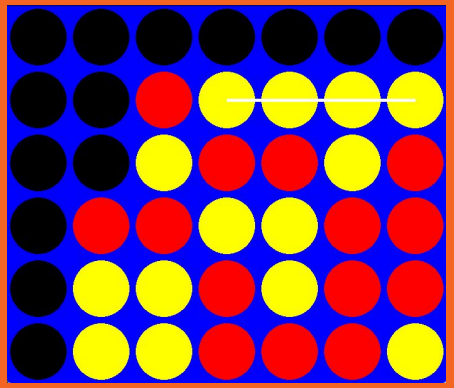
Performance Optimization

Depth Calibration

- A depth of 4 with minimax took too long to run
- A depth of 3 was faster but lost a lot in alphaBetaAi
- A depth of 2 performed better with lots of wins in alphaBetaAI

Reordering of nodes

- The states being evaluated are new each time due to the depth of 2
- The depth of 2 provided 73% win rate in 0.5 secs allowed for computing being competitive with Monte Carlo *without reordering of nodes*



Demonstrations

Minimax and Alpha-Beta Algorithms

alphabetaAI vs stupidAI

alphabetaAI wins both as player 1 and player 2

alphabetaAI vs randomAI

Alphabeta wins out of 5 games

Wins	Seed				
Player	0	1	2	3	4
As 1	5	5	5	5	5
As 2	5	5	5	5	5

Alphabeta wins 100 % of the time

alphabetaAI vs montecarloAI

Alphabeta wins out of 10 games

Wins	Seed				
Player	0	1	2	3	4
As 1	7	10	10	3	10
As 2	9	10	1	1	10

Wins	Seed				
Player	5	6	7	8	9
As 1	0	10	10	10	10
As 2	6	10	1	8	10

Total Wins as player 1 = 80/100 = 80%
Total Wins as player 2 = 66/100 = 66%

Total Wins = 146/200 = 73%