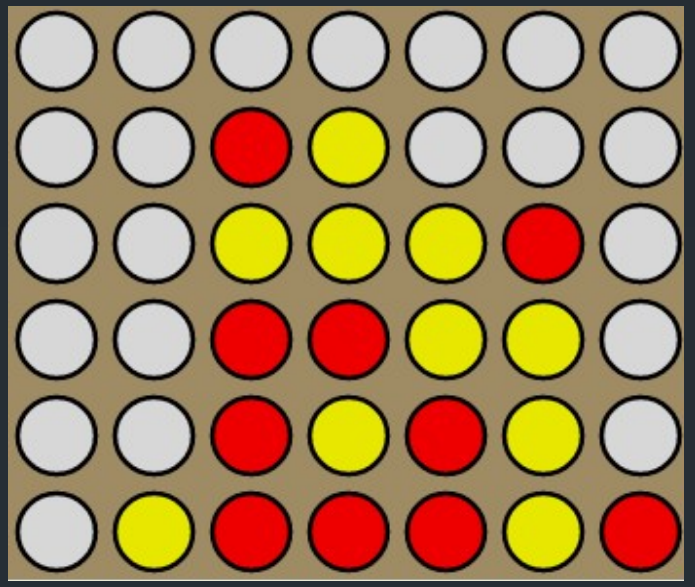
# Monte Carlo Tree Search Connect Four AI

Luis Gustavo De Medeiros lsd53
James Russo jdr289

# Connect Four

- Turn based Perfect Information Game
- Goal of the Game is to connect 4 pieces in any direction
- 4,531,985,219,092 possible distinct legal board states.
- Maximum of seven possible moves at anytime, i.e. maximum branching factor of seven.



# Software Written

- Command Line based implementation of Connect Four in Java
- Easy to create new bots, only need to implement abstract player class and getmove method
- Created own tree class for tree data structure
- No external dependencies.

# Overview

- Wanted to see how well AI bots perform in the game Connect Four
- Implemented an N Random Simulation Bot and a Monte Carlo Tree Search Bot
- Compared the two bots against human opponents and against each other

# Goals

- Create an AI bot that can beat human opponents at the game Connect Four
- Determine which bot is better MCT search bot or N-Pure Monte Carlo search bot
- Also determine which bot is faster at beating opponents

# N-Pure Monte Carlo Search

- Play out N random simulations from each possible move
- Determine which move has the highest win percentage
- Take move with highest win percentage

# Monte Carlo Tree Search

- Start at the root node, then recursively select optimal children nodes until a leaf node is reached
- If the leaf node doesn't end the game then create one or more children nodes and select one
- Run a simulated playout for this children node until a result is reached.
- Update the current move sequence with the simulated result

# UCT

Node selection during tree descent is achieved by choosing the node that maximizes some quantity, analogous to the multi-armed bandit problem in which a player must choose the slot machine (bandit) that maximizes the estimated reward each turn. An Upper Confidence Bounds (UCB) formula of the following form is typically used:

$$v_i + C \times \sqrt{\frac{\ln N}{n_i}}$$

where vi is the estimated value of the node, ni is the number of the times the node has been visited and N is the total number of times that its parent has been visited. C is bias parameter.

# UCT continued

- The formula on the previous slide balances both the exploration of unvisited nodes and the exploitation of known nodes and rewards.
- Balancing these two factors is important, as node values are based on random playouts and thus take several iterations in order to accurately converge.

# MCTS Advantages

- Does not require any domain specific knowledge, only requires knowledge of possible moves and terminal states.
- Can easily to ported to several other games due to its aheuristic characteristics.
- Performs asymmetric tree search, visiting more interesting more often.
- Easy to implement and suitable for games with large branching factors such as Go (b=19), Chess and Connect Four

# MCTS Disadvantages

- Can a require a huge number of iterations in order to find reasonable moves. Due to the fact that games with high branching factors have huge trees and thus many nodes may not be visited often enough in order to provide accurate estimates.
- Playouts are random, don't accurately reflect initially the nature of rational agents.
- In its basic form, does not incorporate any domain-specific knowledge, which can yield significant improvements

# Similarities and Differences

- Similarities
  - Playout random simulations of the same move multiple times
  - Compare relative strength of moves
- Differences
  - MCT search uses a game tree not just random simulation
  - MCT recursively applies Pure-Monte Carlo Search on the tree

# Statistics Used

- Win percentage vs opponent
- Average number of turns to win

# Conclusion

- Two implemented Monte Carlo Search AI's are hard to beat and run very quickly.
- Both implementations can easily be ported to other games and are easy to reason about.
- MCTS AI could be made better by incorporating  Connect Four specific knowledge into them, such as more reasonable gameplay into playouts and by parallelizing the algorithm, increasing the number of visited nodes.
- Next step would be to create an opening playbook with perfect play, as most connect four games are decided before the $8^{th}$ move.