CS 5314: Artificial Intelligence

PA2: Game Playing Agents

Lab Report

Estevan Ramos

Lorelyne Chavez

# **Objective:**

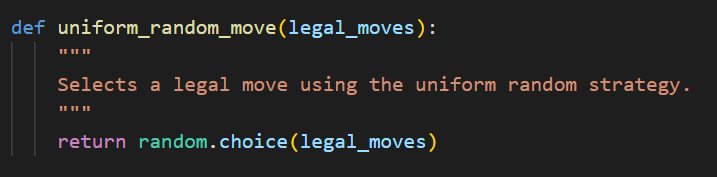
The objective of this lab is to implement and experiment with Monte Carlo Tree Search (MCTS) as a method for planning/strategic reasoning, using the game of Connect Four as a sample domain.

**Part I: Algorithms for Selecting Moves:**

In this part, we implemented three algorithms for selecting moves based on a given board state.

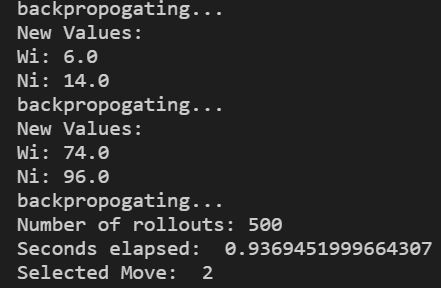
Uniform Random (UR):

* This algorithm selects a legal move for the specified player using a uniform random strategy.
* The move is selected with the same probability for each legal move.
* The code implements this strategy by randomly choosing a move from the available legal moves.



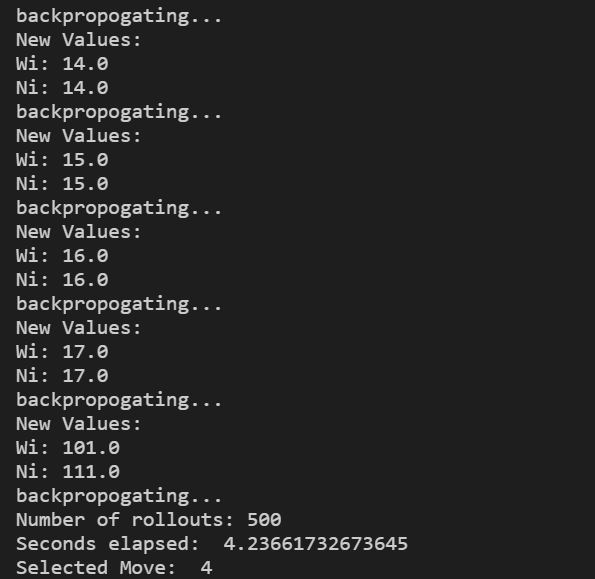
Pure Monte Carlo Game Search (PMCGS):

* This algorithm is a basic form of game tree search based on randomized rollouts, similar to the UCT algorithm without a sophisticated tree search policy.
* It conducts random rollouts from each possible move and selects the move with the best outcome.
* The code performs simulations to estimate the value of each move and selects the move with the highest estimated value.



Upper Confidence Bound for Trees (UCT):

* This algorithm builds on PMCGS and selects nodes within the search tree using the Upper Confidence Bounds (UCB) algorithm.
* It calculates UCB values for all children nodes and picks the one with the highest (or lowest) value based on the player.
* The code implements UCB selection and outputs the UCB values for each move (in Verbose mode) before selecting the move with the highest estimated value.



**Part II: Algorithm Tournaments and Evaluation:**

In this part, we conducted experiments to test the strength of the algorithms by running a round-robin tournament.

* We tested five variations of the algorithms against each other: UR, PMCGS (500), PMCGS (10000), UCT (500), and UCT (10000).
* Each combination of algorithms was run for 100 games, and the number of wins for each algorithm was recorded.
* From the results shown in the screenshot, it's clear that PMCGS (10000) did the best, winning more games and losing fewer compared to the other ways we tried.

# 

# 

# 

# 

# 

# 

# 

# 

**Part III: Enhancements (Extra Credit):**

Modified the code to allow human players to play against AI algorithms UCT and PMCGS. Each team member played 5 games against these algorithms and recorded the results. Playing against these AIs helped us understand various strategies and improve our gameplay tactics.

Although not Implemented we did derive some strategies to increase the effectiveness of the algorithms. First off many of the start and end games can be pre calculated drastically speeding up some of the more costly simulations and calculations. Secondly we can change selection to never pick a child node that will lose the game for the player, in the traditional monte Carlo thats possible to happen that it will knowingly pick a move that will lose the game we can put in a check to ensure that doesn't happen and can even include it in the simulation as well to get more accurate simulations.

## 

| Estevan | Match 1 | Match 2 | Match 3 | Match 4 | Match 5 |
| --- | --- | --- | --- | --- | --- |
| PMCGS 10000 | Loss | Loss | Loss | Loss | Loss |
| UCT 500 | Loss | Loss | Win | Win | Win |

## 

| Lorelyne | Match 1 | Match 2 | Match 3 | Match 4 | Match 5 |
| --- | --- | --- | --- | --- | --- |
| PMCGS 10000 | Loss | Win | Loss | Loss | Loss |
| UCT 500 | Win | Loss | Win | Win | Win |

## 

# Tasks

| Part | Task | Contributor |
| --- | --- | --- |
| Part 1 | Algorithm 1: Uniform Random (UR) | Lorelyne |
| Algorithm 2: Pure Monte Carlo Game Search (PMCGS) | Estevan |
| Algorithm 3: Upper Confidence bound for Trees (UCT) | Lorelyne |
| Part 2 | Running Tournament | Estevan |
| Part 3 | Playing against Agent | Lorelyne |
| Report | Write Report | Lorelyne & Estevan |