Michael Edelman

Professor Rosenfeld

Artificial Intelligence

12/14/2022

HW # 5 Monte Carlo Agent Explanation

The crux of my Monte Carlo Agent implementation lies in the MCAgent(board, color) function. The function receives the state of the board and its color. The function then determines the columns it can consider putting a token in. It places these in the valid_locations variable. We also create a dictionary where the key is one of the valid columns and the value is the probability of us winning if we placed a token in that column. Next we need to actually get these values. So for each column we simulate 100 games played randomly after we put the token in that column (we are simulating how the game would go if we put the token there). We keep track of each win with the current_wins variable. Finally after those 100 simulations the function will calculate the average (current_wins/100) and place that value in the dictionary. We do this for each of the valid columns and fill the dictionary up accordingly. Finally we return the key in that dictionary with the highest value – in other words we return the column that gives us the highest chance of winning the game. Note the play_game(board, color) function is the function that actually executes one simulation of the rest of that game. The function pits a random agent against a random agent and returns 1 if we win and a 0 if we lose. Personally I tested my Monte Carlo Agent against a random agent, my heuristic agent, and your heuristic agent. It won every game against the Random agent and My heuristic agent. Initially it actually only beat my agent half the time, but that was because I only ran 30 simulations – once I increased to 100 it was no match. Finally I beat your heuristic agent 9/10. I also included photos of these runs.