Monte Carlo Tree Search and Its Applications

Max Magnuson
Division of Science and Mathematics
University of Minnesota, Morris
Morris, Minnesota, USA 56267
magnu401@morris.umn.edu

ABSTRACT

Keywords

ACM proceedings, LaTeX, text tagging

1. INTRODUCTION

2. BACKGROUND

Monte carlo tree search combines the random sampling of traditional monte carlo methods with tree searching. It is important to note that monte carlo tree search is at it's best in games with perfect information. For example, a game like chess has perfect information. All of the information that is needed to play the game is encoded in the game board and pieces, and each player has access to all of that information. In contrast, a game like poker does not have perfect information. Each player does not know the contents of the other player's hands as well as the order of the deck.

2.1 The Tree Structure

Monte carlo tree search structures the game state and its potential moves in a tree. Each node in the tree represents the state of the game with the root node representing the current state. Each line represents a legal move that can be made from one game state to another. In other words, it represents the transformation from the parent node to the child node. Any node may have as many children as there are legal moves. For example, at the start of a game of Tic-Tac-Toe the root node may have up to nine children. One for each possible move. Each following child will only be able to have one less child than its parent since the previous moves are no longer available as options. The nodes of the tree also encode for information other than the game state, but we will get into that in the next section.Brand:2014:SEA:2664591.2664612

Todo: I might want to make a graphic to help explain the tree structure with tic-tac-toe

2.2 The Four Steps of MCTS

The process of monte carlo tree search is split up into four processes: Selection, Expansion, Simulation, and Backpropagation. These four processes are iteratively applied until a decision from the AI must be made.

2.2.1 Selection

In the selection process, the MCTS algorithm traverses the current tree looking for the next node to expand. This traversal uses an evaluation function that will prioritize nodes with the highest estimated value. This is known as a tree policy. When a leaf node is reached, it will then transition into expansion.

2.2.2 Expansion

In expansion a new node is added to the tree as a child of the leaf reached in the previous step. Then a simulation is played out to determine the value of the newly added node.

2.2.3 Simulation

In this step, a simulation is played out according to the simulation policy. The policy randomly selects moves until either an end state or a predefined threshold is reached. Then based on the result of the simulation, the value of the newly added node is established. For example, a simulation of a node for Go would reach the end of a game(the end state), and then determine a value based on whether the player won, drew, or lost.

2.2.4 Backpropagation

Once the value of the node is determined, then as the name of this process implies, the rest of the tree can be updated. The algorithm will traverse back to the root node only updating the values of the nodes that it passes through. Only those nodes are effected because each node's respective value is an estimation of values of the nodes after them. For example, a particular node of a tree that encodes for the game Go contains a value that estimates how likely a player is to win or lose given that line of play. In other words, a higher value means that there are more potential games in which that player will win if that move is chosen.

Todo: I should probably introduce the concept of nodes encoding for values earlier in the background when I talk about the tree structure

2.3 The Upper Confidence Bound(UCT)

3. USING MCTS TO PLAY GO