Machine Translated by Google

Monte Carlo Tree Search

Monte Carlo Tree Search (MCTS) is a search algorithm primarily used in artificial intelligence (AI) for decision-making in games and planning problems. It is a technique based on random simulations that balances exploration and exploitation to find the optimal actions in a complex decision space.

Operation of MCTS

MCTS is based on **four phases** that are repeated in multiple iterations to improve decision quality:

1ÿÿSelection

It starts from the root node and iteratively chooses the best child node until reaching a node that has not yet been fully explored.

- To select nodes, a strategy of balancing **exploration** (trying new options) and **exploitation** (choosing the best known option) is used. • A common technique is **Upper Confidence Bound for Trees (UCT),**

which selects the node with the best relationship between its mean value and the number of times it has been explored.

UCT formula:

UCT=WiNi+ClnNNiUCT = \frac{W_i}{N_i} + C \sqrt{\frac{\ln N}{N_i}}UCT=NiWi+CNilnN

Where:

- WiW_iWi = Number of wins on that node. • NiN_iNi = Number of times the node has been visited. • NNN = Total number of visits to the parent node. • CCC = Exploration constant (adjusts the balance between exploration and exploitation).

2ÿÿExpansion

When a node that has not been fully explored is reached, new child nodes are added to the tree, representing possible future moves.

If a node has already been visited before, the selection phase continues until a node with unexplored children is found.

Machine Translated by Google

3ÿÿSimulation (Rollout)

From the newly added node, random simulations are performed until a terminal state is reached (e.g., the end of the game in a gaming environment).

- In this phase the search tree is not used, but random or random decisions are made. following a simple strategy. • A

result is obtained that can be a victory, defeat, or draw.

4ÿÿRetropropagation (Backpropagation)

The simulation result is propagated up the tree, updating the values of the visited nodes.

- The number of visits and success rate of each node on the path are updated. return to the root.
- This allows future selections to favor movements that have led to good results.

Example in a game

Suppose MCTS is used in a chess game:

1. **Selection:** A play is chosen from the current options based on the balance between exploration and exploitation.
2. **Expansion:** If the move has not been explored before, a new node is added to the tree.
3. **Simulation:** A quick game is played randomly from that state to that ends.
4. **Backpropagation:** If the simulation resulted in a win, the value of that move in the tree is reinforced.

After many iterations, the algorithm will have a better estimate of the best available move.

Advantages and disadvantages of MCTS

ÿ **Advantages:**
ÿÿ It doesn't need to evaluate every possible move, making it more efficient than other algorithms like Minimax. ÿÿ It works
well in very large search spaces. ÿÿ It can be used with
partial information, adapting to uncertain environments.

ÿ **Disadvantages:**
ÿ It can be slow if the number of simulations is low.

Machine Translated by Google

ÿ It depends on the quality of the simulations to obtain good results. ÿ It doesn't always find the best solution to problems with very complex rules.

MCTS has been key to the success of advanced artificial intelligences, such as **AlphaGo,** which beat the best human Go players.

Conclusion

Monte Carlo Tree Search (MCTS) is a powerful and efficient algorithm for decision-making in highly complex environments, such as games and planning problems. Its simulation-based approach allows it to find solutions without having to evaluate all possibilities, intelligently balancing exploration and exploitation.

MCTS is widely used in artificial intelligence for games such as chess, Go, and video games, as well as in areas such as robotics and optimization. Its ability to adapt to large search spaces and its flexibility make it a key tool in modern AI.

However, its performance depends on the number of simulations performed and the quality of the move selection strategy. Although it does not guarantee the optimal solution in all cases, its ability to approximate robust decisions makes it an effective alternative to traditional methods such as Minimax.