A Dive to MCTS

liuhanzuo

December 9, 2024

Abstract

TODO

1 BackGround

1.1 MDP

We start the background part by introducing Markov Decision Process(MDP). The basic elements of MDP are as follows:

- State Space $\mathcal{S}$: The set of all possible states.
- Action Space $\mathcal{A}$: The set of all possible actions.
- Transition Probability $T(s', s, a)$: The probability of transitioning to state s' given that the current state is s and the action taken is a.
- Reward Function $R(s, a, s')$: The reward received after transitioning from state s to state s' by taking action a.

The goal of MDP is to find a policy π that maximizes the expected cumulative reward. The policy π is a mapping from states to actions. Which indicates our strategy or policy for a given state. Here, we need to note that we usually choose the π to mamximize the reward.

1.2 Monte Carlo Methods

From the MDP we can see that: when we are making a decision, what we really care about the the "expected reward" or "potential reward" for taking an action a in state s. The Monte Carlo Methods is a class of algorithms that estimate the expected reward by sampling the environment. The basic idea is to simulate the environment and collect the reward. The expected reward is then estimated by the average of the collected rewards. The Monte Carlo Methods are widely used in reinforcement learning, game theory, and other fields.
To be more secific, we define the Q-value function as:

$$Q(s, a) = \frac{1}{N(s, a)} \sum_{i=1}^{N(s)} \mathbb{I}_{s,a} R_i$$

Where the definition is:

- $N(s, a)$ is the number of times action a has been taken in state s.
- $N(s)$ is the number of times state s has been visited. i.e. $N(s) = \sum_a N(s, a)$
- $\mathbb{I}_{s,a}$ is the indicator function that is 1 if the action a is taken in state s and 0 otherwise.
- R_i is the reward received after taking action a in state s.
- $Q(s, a)$ is the estimated Q-value of taking action a in state s (expected reward).

Until here, we have already reached the inner core of the whole system – exploration. It means that how can we make a exploration strategy to adequately explore the state and get the corresponding Q-value(and the optimal policy). The Monte Carlo Methods is a good start for this problem.

2 Monte Carlo Tree Search

2.1 Introduction

This section introduces the family of algorithms known as Monte Carlo Tree Search (MCTS). MCTS rests on two fundamental concepts: that the true value of an action may be approximated using random simulation; and that these values may be used efficiently to adjust the policy towards a best-first strategy. The algorithm progressively builds a partial game tree, guided by the results of previous exploration of that tree. The tree is used to estimate the values of moves, with these estimates (particularly those for the most promising moves) becoming more accurate as the tree is built.
To be more specific, we will introduce some typical algorithm and list the corresponding result, also explain the intuition behind the algorithm.

2.2 General MCTS

The general MCTS algorithm is as follows:

Algorithm 1 General MCTS

```
1: function MCTS($s_0$)
2:     create a node $v_0$ with state $s_0$
3:     while within computational budget do
4:         $v_l \leftarrow$ TreePolicy($v_0$), $\Delta \leftarrow$ DefaultPolicy($s(v_l)$)
5:         Backup($v_l, \Delta$)
6:     end while
7:     return BestChild($v_0$)
8: end function
```

Basically, we have four steps:

1. `Selection`: Start from the root point, we use a policy – child selecting policy to select the next node to expand. (run the `TreePolicy` function)
2. `Expansion`: Expand the selected node and get the next node.
3. `Simulation`: A simulation about what would happen to run on the new point. (run the `DefaultPolicy` function). Typically, we would use a simple(default) strategy to find out the result.
4. `BackPropagation`: Update the information of the nodes on the path from the root to the leaf node.

This four steps can be also expained as a psuedo code at 1.
After the training steps (what we mentioned above), we get a strategy that label the reward of each action in the current state. For each state, we should run a `BestChild` function to get the best action to take.
The choice of `BestChild` function is mainly listed below:

1. `reward-based`: choose the child with highest-reward.
2. `visited-based`: choose the child with highest-visited times.
3. `hybrid-based`: mix the above two policy with a factor, balance the reward and visited times.

Afterwards, we will enumerate some specific algorithms based on the general MCTS algorithm – mainly focus on how to choose `TreePolicy` and `DefaultPolicy`.

2.3 UCT

The Upper Confidence bounds for Trees (UCT) algorithm [1] is a popular variant of MCTS. It uses the UCB1 formula to balance exploration and exploitation. The UCB1 formula is given by:

$$UCB1 = \frac{Q(s,a)}{N(s,a)} + C\sqrt{\frac{\ln N(s)}{N(s,a)}}$$

where C is a constant that controls the balance between exploration and exploitation.

References

[1] David Tolpin and Solomon Shimony. Mcts based on simple regret. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 26, pages 570–576, 2012.