TRON AGENT PROJECT SUMMARIZATION

Implementation

I implement the TRON game player agent using algorithm based on Monte Carlo Tree Search. Monte Carlo Tree Search algorithm expands the scale of game tree step by step through iteration. The tree grows asymmetrically and its growth order is unpredictable. It guides the extension direction according to the average performance of the sub-node. It means that in the search process, should not only make full use of the existing knowledge and give more chances to the nodes with high win rate, but also consider exploring those brothers with low temporary win rate. There is trade-off between "Exploitation" and "Exploration" which means whether to expand unexplored nodes or promise nodes.

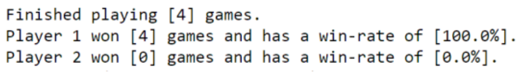
$$\frac{w_i}{n_i} + c\sqrt{\frac{\log n}{n_i}}$$

However, the simulation amount is small in a short time (100-200 on average), this algorithm needs more time to generate a good choice. So I simplified the algorithm to reach a higher amount of simulation.

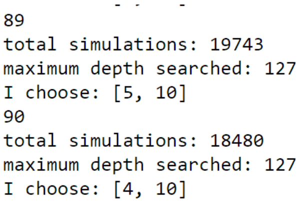
Agent performance

My TRON game agent is able to beat the random agent 100% for every one game, 100% for every four games and 80-100% for every ten games.

To increase the number of simulation, I start 10 threads every time for game simulation, each threads run simulations for 4.5 seconds. I set max action times to 1000 for every simulation. For every choice, the simulations ranges between 100 and 20000.



```
Finished playing [4] games.
Player 1 won [4] games and has a win-rate of [100.0%].
Player 2 won [0] games and has a win-rate of [0.0%].
```



```
89
total simulations: 19743
maximum depth searched: 127
I choose: [5, 10]
90
total simulations: 18480
maximum depth searched: 127
I choose: [4, 10]
```

Takeaways

- Implementing the MCTS algorithm for the TRON game agent
- Improve the algorithm (pruning useless actions)
- Consider RAVE (rapid action value estimate)
- Using multi-threads to expand the simulation amount

Student name: Xinran Tang

Student number: 260936693