Summary of project

In this study, we perform a series of tests on the Monte Carlo Tree Search (MCTS) algorithm, aiming to find the most suitable c-value. Our tests cover the results of the competition at different c-values, number of simulations and simulation depth. To perform these tests, we used Windows 13900KF as the test platform.

Based on the test results, we came to the final conclusion that c=0.25 is the most suitable c-value for our MCTS algorithm. This conclusion is based on the following data: the MCTS algorithm has the highest win rate when the c value is in the range of 0.2-0.3. Specifically, for matches with c values from 0-0.7 (in steps of 0.1), the win rates are as follows: 0.0: 95/100, 0.1: 92/100, 0.2: 95/100, 0.3: 95/100, 0.4: 88/100, 0.5: 86/100, 0.6: 91/100, 0.7: 91/100. These data indicate that c values between between 0.2 and 0.3 have the highest win rate, and combined with the findings from point 1.1, we choose c = 0.25 as the default value.

We also found that the win rate varies at different simulation depths. In the future, we may consider using alphabeta player in the simulation to improve the win rate. In addition, we also observed that the number of simulations and c-values can affect the winning and losing during alphabeta simulations and alphabeta player matches. However, since the alphabeta algorithm will only find the optimal solution, both players in these matches take the exact same path, which explains why there are no partial wins.

To present our test results more fully, we also provide some representative data. For example, with parameters c from 0-0.4 (in steps of 0.05), having MC random simulation play against an alphabeta player with depth 3, with a simulation count of 100 and 100 games performed per c, the results are: 0.0: 25, 0.05: 24, 0.1: 30 , 0.15: 27, 0.2: 29, 0.25: 40, 0.3: 28, 0.35: 32, 0.4: 28.

In summary, we found that the c-value, number of simulations and simulation depth have a significant effect on the win rate of the MCTS algorithm. By carefully analyzing these data, we found the optimal c-value for the MCTS algorithm. In the future, we can further optimize the simulation process to improve the win rate of the algorithm.