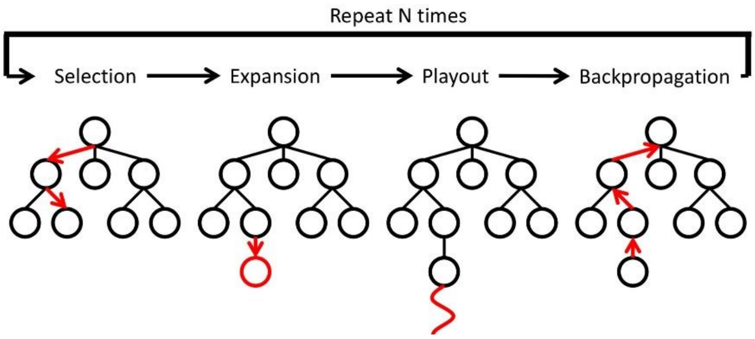
Part I: Implementation of MCTS with UCB policy

Construct a class for nodes, with its current board state, whose turns, last action, # of wins, # of totals, UCB value, a vector of child node, and parent node as node attributes. Next, implement the MCTS in agent.h. It consists of four parts: selection, expansion, simulation, and backpropagation.



i. Selection: use UCB formula as follow and select the child which has max score

$$Score_i = UCB_i = \frac{W_i}{N_i} + C_{bias} * \sqrt{\frac{log_{10}N_p}{N_i}}$$

W_i : The total win playouts of *i*.
N_i : The total playouts of *i*.
C_{bias}: Constant.
N_p : The total playouts of *p*.

ii. Expansion: expand all the valid moves

iii. Simulation: randomly gets one of valid moves and return the winner

iv. Backpropagation: update the nodes' # of wins and # of totals

Finally, according to nodes' # of totals, return the best action.

Part II: Improvement – time management

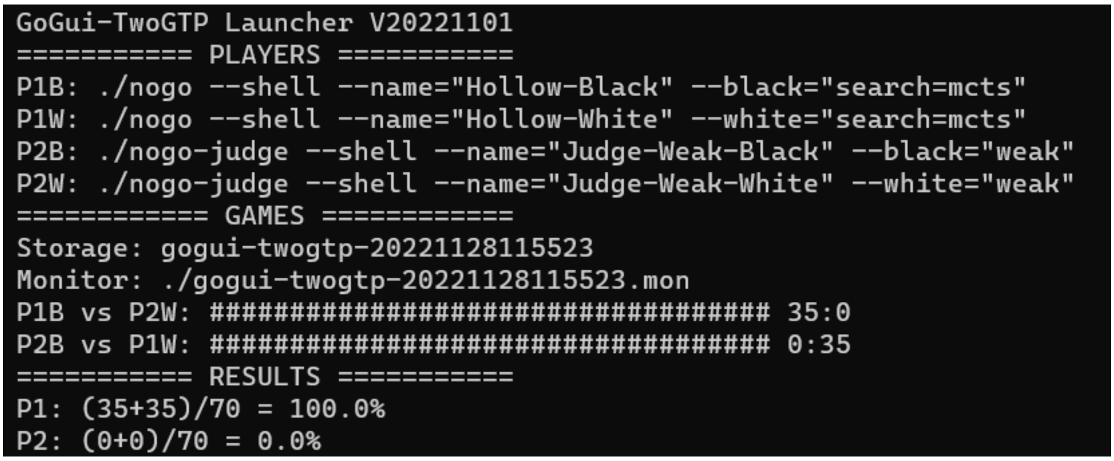
Use a table to manage how much time every step can take. In the early game, it is inefficient to simulate many times; however, in the middle game, the step is more important, so it takes longer time. Moreover, in the endgame, it still takes less time because the number of available actions is few.

Part III: Performance

The table is the win rate of basic MCTS and MCTS with time management run 10 games against three levels AI, respectively.

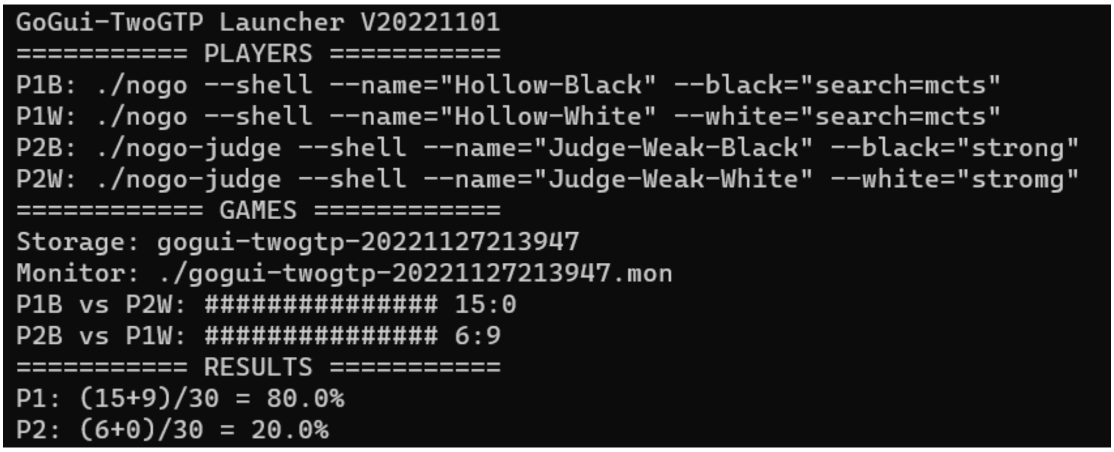
	weak	medium	Strong
basic MCTS	90 %	40 %	13 %
MCTS with time management	100 %	80 %	70 %

run 70 games against weak AI



```
GoGui-TwoGTP Launcher V20221101
=========== PLAYERS ===========
P1B: ./nogo --shell --name="Hollow-Black" --black="search=mcts"
P1W: ./nogo --shell --name="Hollow-White" --white="search=mcts"
P2B: ./nogo-judge --shell --name="Judge-Weak-Black" --black="weak"
P2W: ./nogo-judge --shell --name="Judge-Weak-White" --white="weak"
============ GAMES ============
Storage: gogui-twogtp-20221128115523
Monitor: ./gogui-twogtp-20221128115523.mon
P1B vs P2W: ################################### 35:0
P2B vs P1W: ################################### 0:35
=========== RESULTS ===========
P1: (35+35)/70 = 100.0%
P2: (0+0)/70 = 0.0%
```

run 30 games against strong AI



```
GoGui-TwoGTP Launcher V20221101
=========== PLAYERS ===========
P1B: ./nogo --shell --name="Hollow-Black" --black="search=mcts"
P1W: ./nogo --shell --name="Hollow-White" --white="search=mcts"
P2B: ./nogo-judge --shell --name="Judge-Weak-Black" --black="strong"
P2W: ./nogo-judge --shell --name="Judge-Weak-White" --white="stromg"
============ GAMES ============
Storage: gogui-twogtp-20221127213947
Monitor: ./gogui-twogtp-20221127213947.mon
P1B vs P2W: ############### 15:0
P2B vs P1W: ############### 6:9
=========== RESULTS ===========
P1: (15+9)/30 = 80.0%
P2: (6+0)/30 = 20.0%
```

Part IV: Difficulty

In this project, I also implement parallel MCTS; however, performance of parallel MCTS is worse than basic MCTS. I think maybe I make a mistake in function – getbestaction( ), after checking this part, I still can't improve the performance.