Monte Carlo tree search

YAO ZHAO

Monte Carlo Method vs Alpha-Beta

- **Alpha-Beta**: Due to time constraints, the depth of the recursion is limited ; If the depth of the recursion is shallow, the evaluation is not accurate.
- Monte Carlo Method： How to make decision? The following chess moves are simulated several times, with both players making random moves, until the winner is determined. For example, make a move, then simulate 1000 times, black wins 800 times, white wins 200 times. This move make black's win rate 0.8, White's win rate 0.2, This move favors Black.

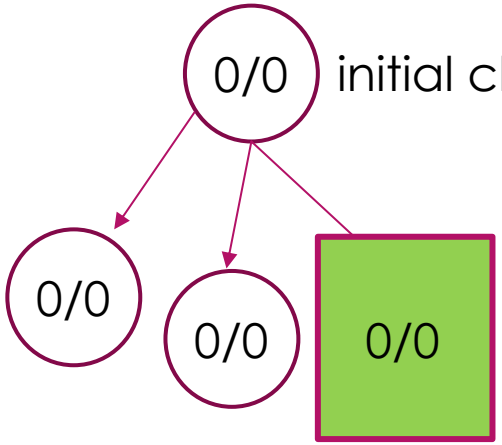
Monte Carlo Method VS Monte Carlo tree search

- game

initial root

select → root

expand(root)

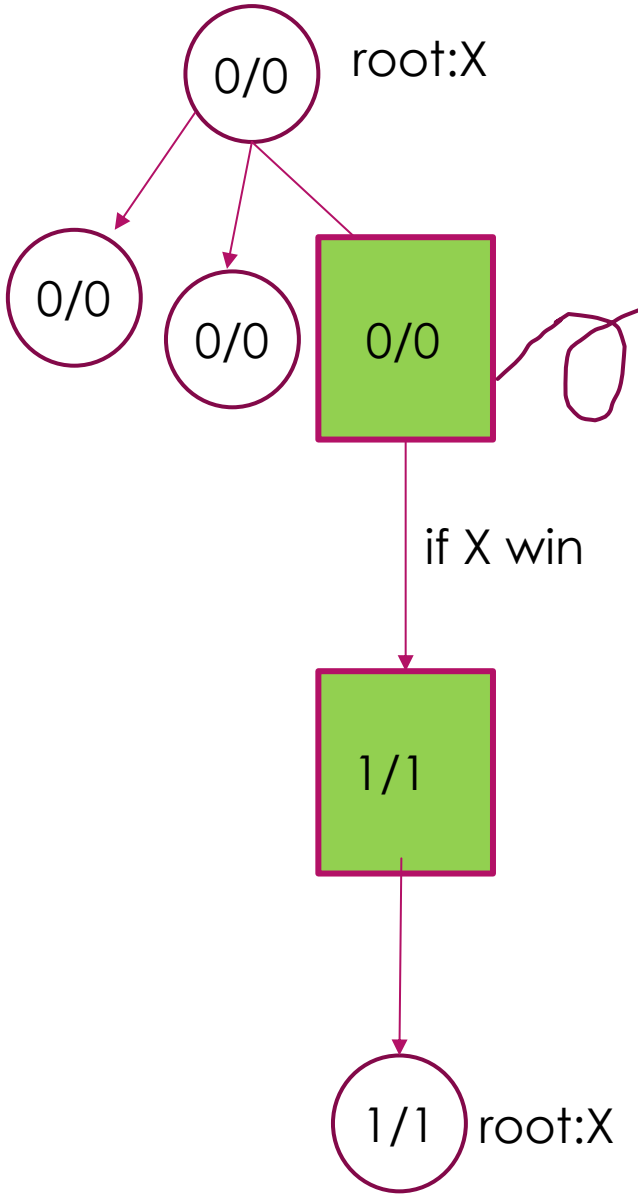


then using UCT select a child node ,
return the **child**

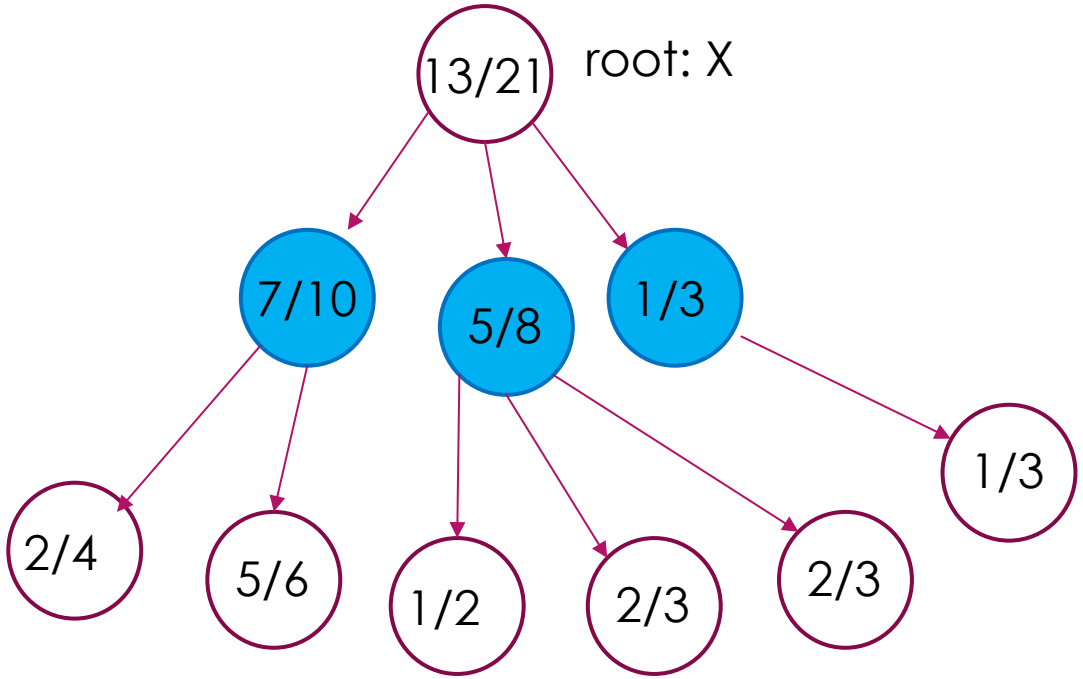
simulate(game, **child**)

using a random strategy to go on the
game until the game terminate.

simulate(game, **children**)



backprop(**children**, **result**)


13/21
root: X
7/10
5/8
1/3
2/4
5/6
1/2
2/3
2/3
1/3

UCT(Upper Confidence with Tree-based Search)

$$\frac{U_i}{N_i} + C\sqrt{\frac{logN}{N_i}}$$

U_i : win number of simulations of child node

N_i : number of simulations of child node

N: total number of simulations

C : a constant, exploration-exploitation trade-off

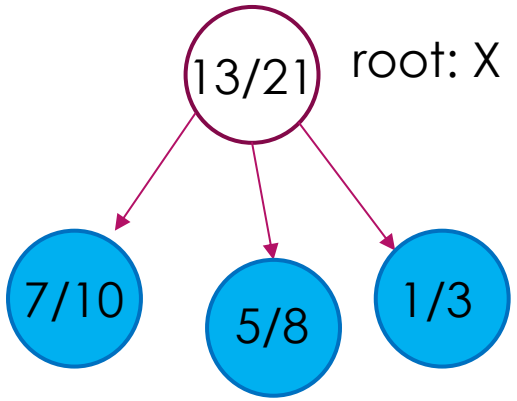
Own side UCB

Current parent node: N = 21

set C =10
child node1：
UCB = 7/10 +10 $*\sqrt{\log(21)/10}$ = 7.32

child node 2：
UCB = 5/8 +10 $*\sqrt{\log(21)/8}$ = 8.03

child node 3：
UCB = 1/3 +10 $*\sqrt{\log(21)/3}$ = 12.43



set C = 0.5
child node 1： UCB =1.03

child node 2： UCB =0.995

child node 3： UCB =0.938

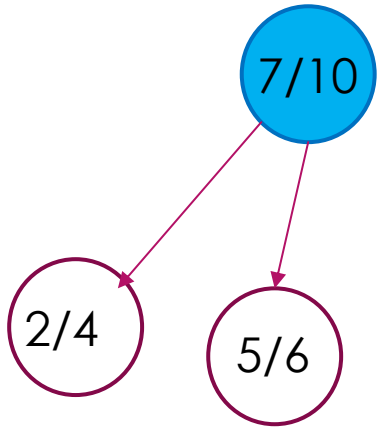
C
- prefer low N_i — **Exploration**
- prefer high $\frac{U_i}{N_i}$ — **Exploitation**

Current parent node: N = 10

set C =10

child node 1： UCB = 2/4 + 10*$\sqrt{\log(10)/4}$

child node 2： UCB = 5/6 + 10 ∗ $\sqrt{\log(10)/6}$



right? **No**

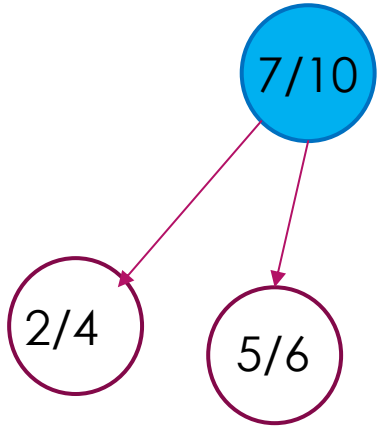
Opponent

Current parent node: N = 10

set C =10

child node 1： UCB = (1 – 2/4) + 10*$\sqrt{\log(10)/4}$ = 9.61

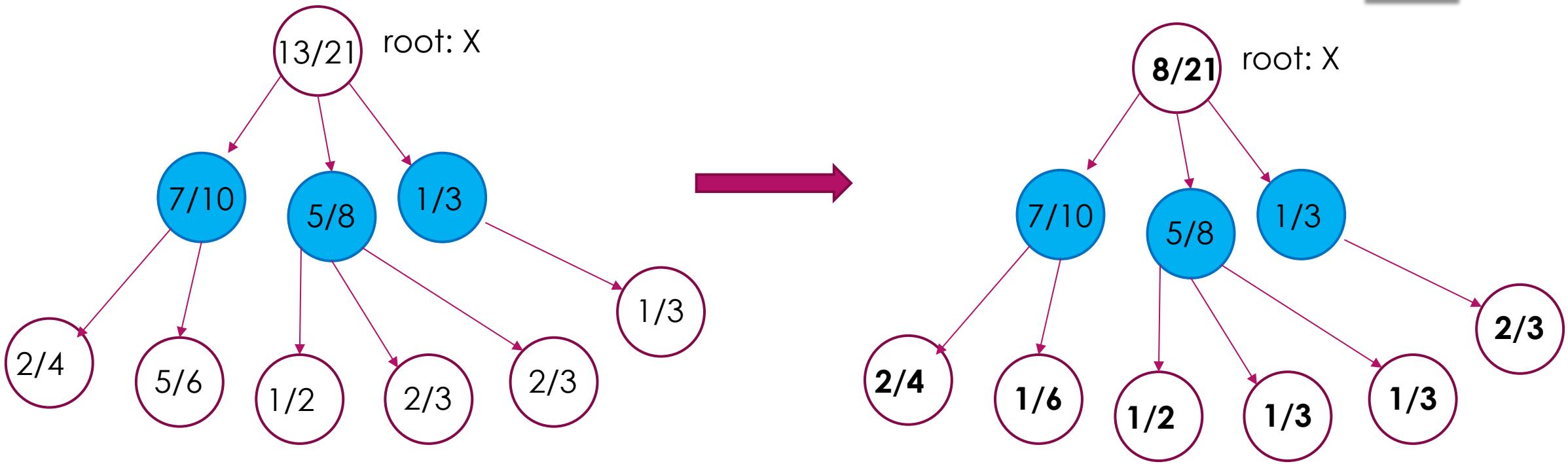
child node 2：UCB = (1-5/6) + 10 * $\sqrt{\log(10)/6}$ = 7.61



set C = 1

child node 1： UCB =1.41

child node 2：UCB =0.91


13/21
root: X
7/10
5/8
1/3
2/4
5/6
1/2
2/3
2/3
1/3
8/21
root: X
7/10
5/8
1/3
2/4
1/6
1/2
1/3
1/3
2/3

Monte Carlo simulation for Reversed-Reversi

- Note: Not only simulation, to get more accurate results, time is not enough, you can try to combined with evaluation function (or another way, uses it with neural networks)