TCG Project4 Report

0716120 林怡安

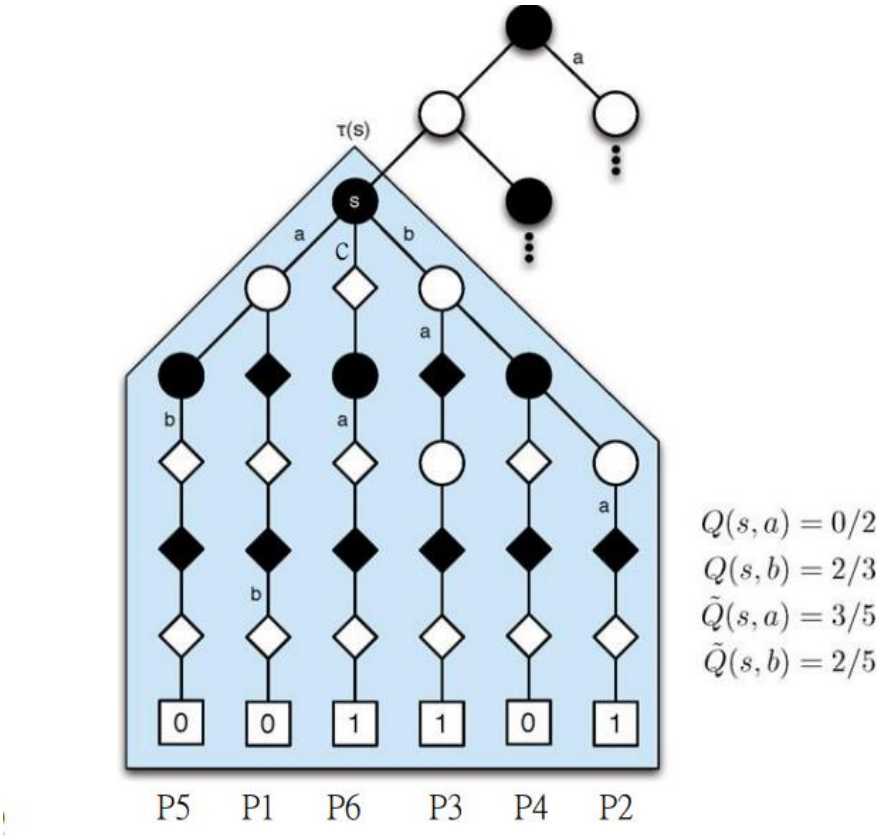
1. Implementation

   這次要做的是加強自己在 project3 的 mcts 程式，盡可能地讓自己能在競賽中勝出。我此次強化的方法為 implement 了 rave 以及做簡單的 time management。

2. Rave

   Rave 的原理是把同一個 state 下只要走過相同步的選項通通都加起來，不像一般的 mcts，只加那條路徑上真的有走過的次數。

   如下圖，a 的 rave 為 5(P5, P1, P6, P3, P2)，但一般 mcts 則只有 2(P1 跟 P5)。



   (1)

   要實作出 Rave，我們須先加 si_rave(rave 的總次數)及 wi_rave(rave 的勝利次數)到 node 的中，方便每個 node 做計算。

```
int wi_rave=0;
int si_rave=0;
```

   (2)

   接下來要更改原本計算每個 child 的 value 的算式，將其改為下圖：

$$Q_{\star}(s,a) = \big(1 - \beta(s,a)\big) Q(s,a) + \beta(s,a) \tilde{Q}(s,a)$$

UCT-RAVE: (Consider exploration bonus:)

$$Q_{\star}^{\oplus}(s,a) = Q_{\star}(s,a) + c\sqrt{\frac{\log N(s)}{N(s,a)}}.$$

$$\beta = \frac{\tilde{n}}{n + \tilde{n} + 4n\tilde{n}\tilde{b}^2}.$$

n: # of visits.

~n: # of RAVE visits.

~b: a selected constant.

Code Implement 如下圖：

```
double b=0.0015;
double beta=double(child->si_rave)/(double(child->si)+double(child->si_rave)+4*double(child->si*child->si_rave*b));
```

```
return (1-beta)*(double(child->wi)/double(child->si))+beta*double(child->wi_rave)/double(child->si_rave)
+exploration_c*sqrt(log(double(child->parent->si))/double(child->si));
```

(1- beta)*原本的 mcts 勝率+beta*rave 勝率+UCB 的 exploration 式子。

(3)

Backpropagation 的地方也要改，新增增加 rave 次數的 code。

```
std::vector<action::place> moves;
bool exist=false;
```

```
for(int i=0;i<moves.size();i++){
    if(current->placed_step==moves[i]){
        exist=true;
        break;
    }
}
if(!exist){
    moves.push_back(current->move_placed);
}
```

上兩張圖是用來儲存每一步，之後就可以拿 moves 裡的每一步去跟上面的 parent 做比對，如有相同的就可以增加他的 rave 次數以及勝利次數。如下圖：

```
if(current->parent!=nullptr&&current->parent->parent!=nullptr){
    for(int i=0;i<current->parent->parent->childrens.size();i++){
        for(int j=0;j<moves.size();j++){
            if(moves[j]==current->parent->parent->childrens[i]->move_placed){
                current->parent->parent->childrens[i]->si_rave++;
                current->parent->parent->childrens[i]->wi_rave+=score;
            }
        }
    }
}
```

最後也別忘了自己也要更新 rave。

```
current->si_rave++;
current->wi_rave+=score;
```

實作玩 rave 後，用之前 mcts-strong 來對打，勝率來到 8 成。以前普通 mcts 我大概只有 7 成。可見 rave 的確有增強的效果。

```
GoGui-TwoGTP Launcher V20221101
=========== PLAYERS ===========
P1B: ./nogo --shell --name="Hollow-Black" --black="mcts T=1000"
P1W: ./nogo --shell --name="Hollow-White" --white="mcts T=1000"
P2B: ./nogo-judge --shell --name="Judge-Weak-Black" --black="strong"
P2W: ./nogo-judge --shell --name="Judge-Weak-White" --white="strong"
============ GAMES ============
Storage: gogui-twogtp-20230109095232
Monitor: ./gogui-twogtp-20230109095232.mon
P1B vs P2W: ##### 4:1
P2B vs P1W: ##### 1:4
=========== RESULTS ===========
P1: (4+4)/10 = 80.0%
P2: (1+1)/10 = 20.0%
```

3. Time management

在聽同學報告的時候，有聽到同學報告 time management，大致是想要讓每一步計算時間都用到最有效率的方法去增進計算。像是可能前幾步以及盤面最後幾步都不太需要太久的時間去運算，就可以減少這兩部分的計算時間，將剩下的時間分配到覺得比較重要的部分，像是中間的步數。

我這次實作的 time management 相當的簡單，只是縮短前 4 步以及後大概 5 步的計算時間，將省下的時間以平分的方式給中間的步數。

```
if(steps<=4){
    if(double(end-start)/CLOCKS_PER_SEC>4.0){
        break;
    }
}
else if(steps<=26){
    if(double(end-start)/CLOCKS_PER_SEC>8.3){
        break;
    }
}
else{
    if(double(end-start)/CLOCKS_PER_SEC>4.0){
        break;
    }
}
```