AI Final project report – 0756604 曹育榕

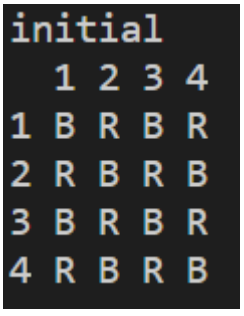
- **Search strategy**

在本次專案中使用 MCTS(monte carlo tree search)來進行搜尋，找出 AI 要移動的格子為何。整個搜尋的過程分為 Tree policy,Default policy,Backpropagation 三個階段 ：

- **Tree policy** (詳見 code)
  在此階段我們會選出一個要進行 Default policy 的 node。選擇的方式為，若目前的 node 沒有 children(沒有 Expand 過)，那就 Expand 一個 child 來當作 target node(一次只 expand 一個 node)；若某 node 已經有 children 了，則會有一半的機率使用 UCB function 或是 Expand 一個新的 node 來當作 target node。
- **Default policy** (詳見 code)
  隨機選擇合法的格子進行移動，直到達到終止條件，之後回傳計算好的 reward。
  終止條件 ： 當能分出輸贏或是平局時。
  Reward: 若結果為贏，reward = 3；若結果為輸，reward = -3，若結果為平手，reward = -2
- **Backpropagation**
  將 Reward 向上更新

將上述步驟執行 K 個(K = 1000) iteration 後，再利用 UCB function 選出一個最佳的格子來移動。(K 在 AILoop()函數裡設定)

- 操作步驟
  - 遊戲架構的部分在程式的最下方，執行程式後會看到 initial board，上方跟左方的數字輔助 User 輸入欲移除的格子編號。



- 選擇是否讓 User 先攻，若是的話，User 可以選擇一個顏色。
  接下來輸入 row 跟 col 來移除格子，會先顯示移除該格的棋盤，隨後顯示 Side effect 後的結果。

```
user go first or not(Y/N):Y
Enter your color B/R:B
User => enter a row : 1
User => enter a col : 1
user move (1,1) : B

  1 2 3 4
1 X R B R
2 R B R B
3 B R B R
4 R B R B


side effect

  1 2 3 4
1 X R B R
2 R B R B
3 B R B R
4 R B R B
```

- AI 會緊接著做出回應，User 可以選下一步要移除的格子。

```
AI move (1,4) : R

  1 2 3 4
1 X R B X
2 R B R B
3 B R B R
4 R B R B


side effect

  1 2 3 4
1 X R B X
2 R B R B
3 B R B R
4 R B R B


User => enter a row :
```

- 不斷巡迴直到遊戲結束。

```
AI move (4,1) : R

  1 2 3 4
1 X X X X
2 X X X B
3 X X X X
4 X X X X


side effect

  1 2 3 4
1 X X X X
2 X X X B
3 X X X X
4 X X X X


AI win
```

- 若有需要的話，調整 AILoop()裡的 K 可以調整 MCTS 的 iteration 次數。

```
def AILoop(pre_state):
    #AI turn
    K = 1000  # MCTS training iteration
    ply_node = Node(pre_state)
    AI_state = UCTSEARCH(K,ply_node).state
```

- 調整 fileName 就可以更換棋盤。

```
282    # read input file
283    fileName = 'test_n4.txt'
284    board = [line.strip() for line in open(fileName,'r')]
285    size = int(board[0])
```