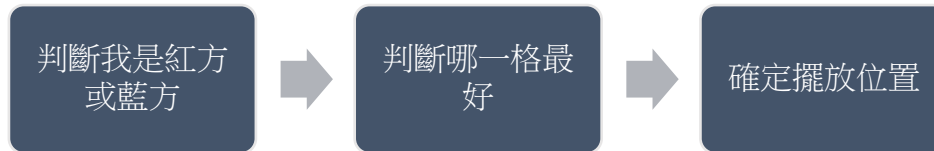


1) Project Description

1-1) Program Flow Chart



1-2) Detailed Description

1. 判斷我是紅方或藍方
 - ◆ 利用 `player.get_color()` 知道我是哪一方，以此來判斷接下來位置的好壞
2. 判斷哪一格最好
 - ◆ 用 `minimax+alpha_beta` 方法
 - ◆ 自己定義一個 `value function` 用來判斷目前棋盤的分數
 - ◆ 把所有我能下的位置都用 `value function` 算分數，最後取最大的那格下
 - ◆ 總共往下做 4 層，5 層有時候會大於 1 秒
 - ◆ 如果有遇到更好的位置，更新 `minimax_row`、`minimax_col` 的值
3. `value function` 定義方式
 - ◆ 依序判斷板子上的每一顆棋子
 - ◆ 以剩餘棋數判斷是否結束，輸了得 -10000 分，贏了得 10000 分
 - ◆ 如果還沒有結束，板子上我的棋子越多越好
 - ◆ 要注意旁邊是否有對方快爆掉(剩餘格數=1)的棋子，如果有的話 `value` 減少，減少得多寡以我這格的 `capacity` 判斷，`capacity` 越少減越多，因為角落、邊邊是比較好的位置
 - ◆ 如果我自己快爆了，附近也有快爆的我的棋子，`value` 增加，因為可以造成 `chain reaction`
 - ◆ 如果附近都沒有對方快爆掉的棋子，那在角落、邊邊就更好了
4. 確定擺放位置
 - ◆ 把 `minimax_row`、`minimax_col` 分別記錄到 `index[0]`、`index[1]` 上

2) Screen Shots

2-1) Partial Implemented Code

1. Value function

```

int val_func(Board board, Player player) {
    int val=0, orb=0, op_orb=0;
    int dir[8][2]={{1,0},{-1,0},{0,1},{0,-1},{1,1},{1,-1},{-1,1},{-1,-1}};

    for(int i=0; i<5; i++){
        for(int j=0; j<6; j++){
            if(board.get_cell_color(i,j)==color){
                orb+=board.get_orbs_num(i,j);
                bool safe=true;
                int cnt=0;
                int left=board.get_capacity(i,j)-board.get_orbs_num(i,j);
                for(int k=0; k<8; k++){
                    int near_i=i+dir[k][0], near_j=j+dir[k][1];
                    if((0<=near_i&&near_i<ROW) && (0<=near_j&&near_j<COL)){
                        if(board.get_cell_color(near_i,near_j)==op_color){
                            int op_left=board.get_capacity(near_i,near_j)-board.get_orbs_num(near_i,near_j);
                            if(op_left<=1){
                                safe=false;
                                val-=8-board.get_capacity(i,j);
                            }
                        }
                        else if((board.get_cell_color(near_i,near_j)==color)&&(left<=1)){
                            if((board.get_capacity(near_i,near_j)-board.get_orbs_num(near_i,near_j)<=1)){
                                val+=2;
                            }
                        }
                    }
                }
            }
            if(safe){
                if(board.get_capacity(i,j)==3)
                    val+=5;
                else if(board.get_capacity(i,j)==5)
                    val+=3;
                if(left==1)
                    val+=3;
            }
            else if(board.get_cell_color(i,j)==op_color) op_orb+=board.get_orbs_num(i,j);
        }
    }
    val+=orb;
    if(orb>0&&op_orb==0)
        val=10000;
    else if(orb==0&&op_orb>0)
        val=-10000;
    return val;
}

```

2. Minimax with alpha beta pruning

```

int minimax_alpha_beta(Board board, int depth, int a, int b, Player player) { //init a=-infinite, b=+infinite
    if(depth==0 || board.win_the_game(player))
        return val_func(board, player);
    if(player.get_color()==color){ //max
        Player op(op_color);
        //int best_val=INT_MIN;
        for(int i=0; i<5; i++){
            for(int j=0; j<6; j++){
                if(board.get_cell_color(i,j)==color || board.get_cell_color(i,j)=='w'){
                    Board temp_board=board;
                    temp_board.place_orb(i,j,player);
                    int temp_val=minimax_alpha_beta(temp_board, depth-1, a, b, op);
                    if(temp_val>a){
                        a=temp_val;
                        if(depth==4){
                            minimax_row=i;
                            minimax_col=j;
                        }
                    }
                    if(b<=a) break;
                }
            }
            if(b<=a) break;
        }
    }
    return a;
}



```

```

} else if (player.get_color() == op_color) { //min
    Player this_player(color);
    //int best_val = INT_MAX;
    for (int i = 0; i < 5; i++) {
        for (int j = 0; j < 6; j++) {
            if (board.get_cell_color(i, j) == op_color || board.get_cell_color(i, j) == 'w') {
                Board temp_board = board;
                temp_board.place_orb(i, j, &player);
                int temp_val = minimax_alpha_beta(temp_board, depth - 1, a, b, this_player);
                if (temp_val < b)
                    b = temp_val;
                if (b <= a) break;
            }
        }
        if (b <= a) break;
    }
    return b;
}
}
}

```

2-2) GitHub Control History

	wangyupin Create algorithm_st.cpp	ce08723 12 hours ago	🕒 1 commit
	algorithm_st.cpp	Create algorithm_st.cpp	12 hours ago

2-3) Compare with TA's AI Code (*randomMove*) for 7 results. (7 pictures)

Round: 93
Place orb on (0, 0)

00	0	0000	0000	0000	00
000	0000000	000000			0
0000	000	000	0000000	0000000	0000
0	0000	00	0000000	0000000	
	0	0	00	00	00

Red Player won the game !!!

Round: 89
Place orb on (3, 4)

00	0000	0000	0000	0	00
0000	00		0000000	0000	000
	0000	0000	00000	00	00
0000	0000000	000000	0	00	000
00	0	000		0000	0

Red Player won the game !!!

Round: 101
Place orb on (0, 0)

00	0	0000	000	000	00
0	0000000	000	0000000	0	0
	000000	0		00000	000
00000	00000	0000	000000	00000	00
00	00	0		0000	00

Red Player won the game !!!

```

Round: 103
Place orb on (4, 5)
=====
|00    | |00    | |0    | |0000  | |00    | |00    |
|0000  | |    | |0000  | |0000  | |000  | |0    |
|0      | |0000000| |00000  | |0000  | |00    | |00    |
|0000  | |0000000| |0000000| |0000000| |0000  | |0000  |
|00     | |0000  | |0000  | |00000  | |    | |0    |
=====

Red Player won the game !!!

```

```

Round: 81
Place orb on (4, 5)
=====
|00    | |    | |0000  | |0    | |0000  | |00    |
|00    | |0000000| |0000000| |00    | |    | |0000  |
|00    | |000000  | |00    | |00    | |00    | |00    |
|00    | |0000  | |0000000| |000000  | |000  | |000  |
|0      | |    | |0    | |00    | |    | |0    |
=====

Red Player won the game !!!

```

```

Round: 121
Place orb on (3, 3)
=====
|00    | |0000  | |0000  | |0000  | |0000  | |00    |
|0000  | |000  | |0000000| |0000000| |00    | |00    |
|0      | |0000000| |000000  | |0000000| |000000  | |00    |
|0      | |0000000| |0000000| |00    | |00000  | |0    |
|00     | |00000  | |0    | |    | |00000  | |00    |
=====

Red Player won the game !!!

```

```

Round: 87
Place orb on (2, 2)
=====
|00    | |0000  | |000000  | |0000  | |0    | |00    |
|000    | |0000  | |00    | |    | |0000000| |0    |
|0000  | |    | |0000  | |0000  | |000000  | |00    |
|0000  | |0000  | |00000  | |    | |000  | |0    |
|    | |00    | |00    | |00    | |00    | |00    |
=====

Red Player won the game !!!

```

2-4) Describe the reason why you win TA's AI Code or why you can't win

我的棋子會先以下在角落和邊邊為主，這樣 TA 的就比較多是下在中間，中間要集滿八顆才能爆炸，但角落和邊邊比較容易，而且我的棋子盡量連續分布且非必要不會讓它爆炸，這樣之後比較容易產生 chain reaction 也比較不會被 TA 的棋子變色。