

オセロ×プログラミング

名前

1 人目: 2 人目:

使い方

Othello AIをつくらう AI作成 シミュレーション

A B C D E F

1

2

3

4

5

6

盤面

盤面コントロール

自分のAI 評価値表示

棋譜

棋譜保存

Othello AIのコード

```
1 // 盤面を初期化
2 let Factor_NumOfDiscs = 1; // 両方の石の数を数える (0以上の場合は多く通したがる)
3 let eval_board = [
4   // 盤面のそれぞれの位置の値を設定 (大きい数字なほど優先して打つようになる)
5   [0, 1, 1, 1, 1, 1],
6   [1, 1, 1, 1, 1, 1],
7   [1, 1, 1, 1, 1, 1],
8   [1, 1, 1, 1, 1, 1],
9   [1, 1, 1, 1, 1, 1],
10  [1, 1, 1, 1, 1, 1]
11 ];
12
13 // 両方の石の数を数える (両方の石の数を数える)
14 // [eval_board, moveArray] = evalBoard(moveArray) // 左の数を数える, moveArray
15 // [eval_board, moveArray] = evalBoard(moveArray) // 右の数を数える, moveArray
16 // Factor_NumOfDiscs = evalBoard(moveArray) // 左の数を数える, moveArray
17 // [eval_board, moveArray] = evalBoard(moveArray) // 右の数を数える, moveArray
18
19 // 評価値の計算
20 let moveList = MoveListFunc(board, turn);
21 let eval_list = [];
22 let eval_array = [];
23 for (let i = 0; i < moveList.length; i++) {
24   eval_list.push(
25     eval_board(moveList[i][0])(moveList[i][1]) +
26     Factor_NumOfDiscs * moveList[i][2] +
27     Math.random() * 0.1
28   );
29 }
30 let best_move_index = eval_list.indexOf(Math.max(...eval_list));
31 let best_move_array = [moveList[best_move_index][0], moveList[best_move_index][1]];
32 return_arr = [best_move_array, eval_list];
33 }
```

盤面数

黒 0回

白 0回

白 0回

対戦設定

黒番

白番

対戦設定

設定

AI登録

AIの名前

AI0

登録

● 盤面コントロール

盤面を進めたり戻したりすることができます。終わった後、戻しながら大事な場面を確認できます。

● 対戦設定

黒と白のプレイヤーを人間かコンピューターの選択をすることができます。

● 評価値表示

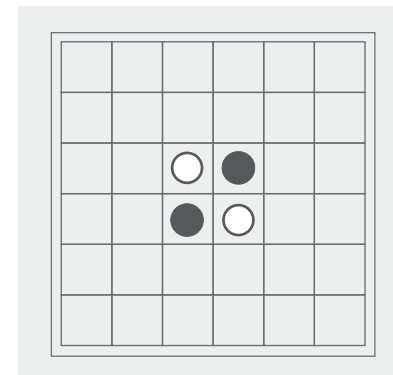
任意のAIで現在の盤面の点数を表示することができます。

● AIのアップロード

作ったAIをアップロードすることができます。

ペアで対戦してみよう！

勝負の分かれ目になった局面

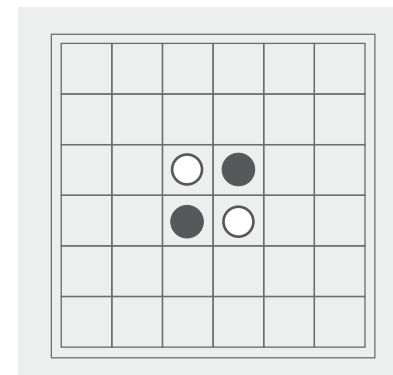


どうやれば勝てる？ どういう手がいい手？

どうなると負ける？ どういう手を避けるといい？

コンピューターと対戦してみよう！

勝負の分かれ目になった局面



どうやれば勝てる？ どういう手がいい手？

どうなると負ける？ どういう手を避けるといい？

オセロ×プログラミング

評価値の計算方法

以下の2つの要素を調整してAIを強くします。

eval_board : 盤面の場所の点数 (数字が大きいところに打ちたがる。)

Factor_NumOfDiscs: 返す石の数の大事さ (0以上では多く返す。0以下では少なく返す。)

この二つの要素を使って以下の計算をすることで、手に点数をつけます。

点数 = **eval_board**(置く場所) + **Factor_NumOfDiscs** × 返す石の数

この二つの要素を使って以下の計算をすることで、手に点数をつけます。

AIの調整の例

● 右下の角を優先的に打つ AI にする

eval_boardの右下角の数字を大きくすると、そこを優先的に打つようになります。

```
let Factor_NumOfDiscs = 0;  
let eval_board = [  
  [0, 0, 0, 0, 0, 0, 0],  
  [0, 0, 0, 0, 0, 0, 0],  
  [0, 0, 0, 0, 0, 0, 0],  
  [0, 0, 0, 0, 0, 0, 0],  
  [0, 0, 0, 0, 0, 0, 0],  
  [0, 0, 0, 0, 0, 0, 0],  
  [0, 0, 0, 0, 0, 0, 5],  
];
```



		0	0			
		○	○	0		
	●	○	○	0		
		●	○	0		
		●	●	○	5	

● できるだけ多く返す AI にする

Factor_NumOfDiscsの数字をおおきくするとできるだけ多く返すようになります。

```
let Factor_NumOfDiscs = 1;  
let eval_board = [  
  [0, 0, 0, 0, 0, 0, 0],  
  [0, 0, 0, 0, 0, 0, 0],  
  [0, 0, 0, 0, 0, 0, 0],  
  [0, 0, 0, 0, 0, 0, 0],  
  [0, 0, 0, 0, 0, 0, 0],  
  [0, 0, 0, 0, 0, 0, 0],  
  [0, 0, 0, 0, 0, 0, 0],  
];
```



		2	4			
		○	○	1		
	●	○	○	3		
		●	○	1		
		●	●	○	1	

● 右下の角を優先的に打つ AI にする

- **Factor_NumOfDiscs**の数字をマイナスにするとできるだけ少なく返すようになります。

- **eval_board**の右下角の数字を大きくすると、そこを優先的に打つようになります。

```
let Factor_NumOfDiscs = -1;  
let eval_board = [  
  [0, 0, 0, 0, 0, 0, 0],  
  [0, 0, 0, 0, 0, 0, 0],  
  [0, 0, 0, 0, 0, 0, 0],  
  [0, 0, 0, 0, 0, 0, 0],  
  [0, 0, 0, 0, 0, 0, 0],  
  [0, 0, 0, 0, 0, 0, 0],  
  [0, 0, 0, 0, 0, 0, 5],  
];
```



		-2	-4			
		○	○	-1		
	●	○	○	-3		
		●	○	-1		
		●	●	○	4	

AIをつよくしてみよう

AIの2つの要素をいくつにするといいか考えてみましょう。

eval_board

		○	●			
		●	○			

Factor_NumOfDiscs

Factor_NumOfDiscsと**eval_board**以外にどんな要素があるとより強くなる?