ECE 448

ARTIFICIAL INTELLIGENCE

SPRING 2023

MP2: Planning, Games Report

KE XU (3190110360) (Dropped)
JIARUN HU (3190110383)
ZHEYANG JIA (3190110096)

Prof. Hongwei Wang, Zhejiang University

Prof. Hasegawa-Johnson, University of Illinois Urbana-Champaign

Contents

1 Section I

1.1 Problem:pentomino tiling

The problem will give me a set of board with different shapes which has filled by 0s and 1s. And what need to do is try to flip and rotate pentominos to fill the area of the board filled by 1s with no overlaps. The constraints of the problem is that

- No pentominos overlap
- All pentominos lie completely on the board
- Fill all 1s squares.
- Cannot cover 0s squares

1.2 Algorithm

- Generate list of pentominos with different transformation method (rotate, flip).
- Select a location, Use pentominos in the pentominos list and try to fill the board.
- Recursively execute the 2 step, until cannot fill the board and remove the pentominos to return to the last location. If they can fill it, then return with success.

2 Section II

2.1 notes for playGamePredifinedAgen

def playGamePredifinedAgent(self, maxFirst, isMinimaxOffensive, isMinimaxDefensive):

maxFirst(bool): True for maxPlayer plays first, and False for minPlayer plays first.

isMinimaxOffensive(bool):True is minimax and False is alpha-beta. For offensive agent.

isMinimaxOffensive(bool):True is minimax and False is alpha-beta. For defensive agent.

2.2 offensive(minimax) vs defensive(minimax)

uttt.playGamePredifinedAgent(True, True, True)

```
O _ X O _ _ O X _
_ X _ _ _ _ _ _ _
X _ _ _ _ _ _ _ _

_ _ _ X _ O _ _ _
_ _ _ _ _ _ _ _ _
_ _ _ _ _ _ _ _ _

_ _ _ _ _ _ _ _ _
_ _ _ _ _ _ _ _ _
_ _ _ _ _ _ _ _ _

expandedNodes:  [704, 1391, 2000, 2656, 3175, 3749, 4308, 4895, 5246]
The winner is maxPlayer!!!
```

2.3 offensive(minimax) vs defensive(alpha-beta)

uttt.playGamePredifinedAgent(True, True, False)

```
O _ X _ O X O X O
_ X O _ X _ _ X _
_ _ O X O _ _ _ _

_ _ _ X O O _ _ _
O _ X X O _ O _ X
O _ X _ _ _ O _ X

X O _ _ _ _ X X _
_ _ _ X _ _ _ _ O
_ _ _ _ _ _ _ _ O

expandedNodes:  [704, 1000, 1609, 1894, 2413, 2670, 3229, 3471, 3994, 4203,
    4606, 4830, 5266, 5454, 5779, 6065, 6478, 6760, 7097, 7357, 7720, 7899,
    8267, 8433, 8745, 8877, 9091, 9245, 9461, 9570, 9840, 9950, 10168,
   10205, 10299]
The winner is maxPlayer!!!
```

2.4 offensive(alpha-beta) vs defensive(minimax)

uttt.playGamePredifinedAgent(False, False, True)

```
X O O X X O X O _
X O _ _ X _ X _ _
_ _ _ _ X _ _ _ _

O O _ O O X _ _ _
_ _ _ _ _ _ _ _ _
_ _ _ _ _ _ _ _ _

_ _ _ _ _ _ _ _ _
_ _ _ _ _ _ _ _ _
_ _ _ _ _ _ _ _ _

expandedNodes:  [704, 1000, 1609, 1894, 2413, 2700, 3133, 3389, 3929, 4178,
    4638, 4843, 5362, 5516, 5976, 6124, 6505, 6567]
The winner is maxPlayer!!!
```

2.5 offensive(alphabeta) vs defensive(alpha-beta)

uttt.playGamePredifinedAgent(False, False, False)

```
X O O X X O X O _
X O _ _ X _ X _ _
_ _ _ O _ _ _ O _
```

```
O O _ O _ X _ _ _
_ _ X O _ X O _ O
_ _ X _ _ _ X X _

_ _ X _ X _ _ _ _
_ _ O _ X _ _ O _
_ O _ O X O _ O _

expandedNodes:  [240, 536, 791, 1076, 1289, 1576, 1742, 1998, 2205, 2454,
   2679, 2884, 3156, 3310, 3544, 3692, 3891, 4111, 4364, 4545, 4772, 4950,
   5155, 5362, 5554, 5708, 5871, 6069, 6220, 6374, 6478, 6625, 6695, 6768,
   6853, 6953, 7004, 7035]
The winner is maxPlayer!!!
```

3 Section III

We play 20 times games of offensive agent vs your agent, our defensive agent win 17 out of 20 games.

Percentage of winning time: 85% Expanded nodes for games: [126183, 116619, 12047, 12482, 102934, 118362, 124923, 128374, 108363, 103783, 103848, 113748, 113802, 113294, 117492, 117392, 128373, 119742, 118374, 103729]

3 representative final game boards:

First type: Defend vertically

```
_ X _ 0 X _ X _ 0
0 X X _ 0 X _ _ _
0 _ _ _ _ 0 0 _ _

_ X _ 0 _ _ _ _ _
_ 0 _ _ _ _ X _ _
X 0 X _ X _ X _ _
```

The winner is minPlayer!!!

Second type: Defend Horizontally

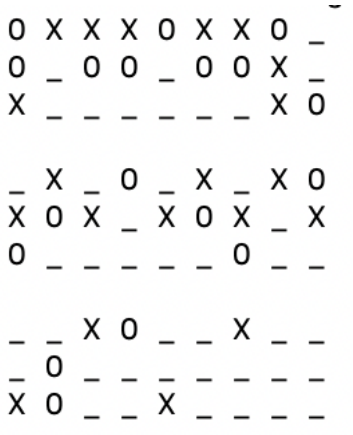
```
0 X X X 0 X _ 0 0
0 _ X X 0 0 X X _
X _ _ 0 _ _ _ _ _

X X 0 0 _ _ 0 _ _
0 0 X 0 X 0 X _ X
X _ _ _ X X 0 0 _

0 X _ 0 0 _ _ X _
_ 0 _ _ _ _ _ X _
X 0 0 _ X X 0 _ 0
```

The winner is minPlayer!!!

Third type: Defend Diagonally



The winner is maxPlayer!!!

I think the strategy we used to design our evaluation function works quite well by simply assigning much more penalty score to prevent offensive agent from winning with three-in-row cases. It will extend the game times so that the defensive agent at least will not lose the game at current cycle and that all full three-in-row but with different players are clear example of successful defense.

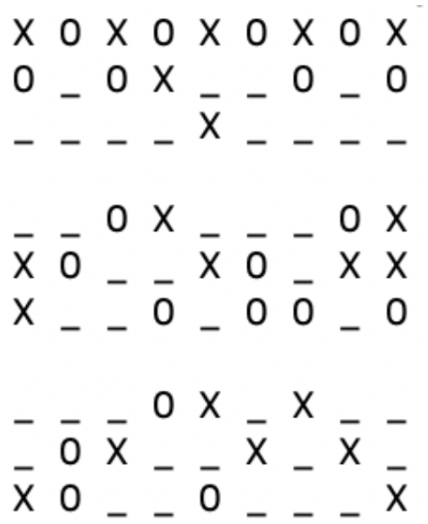
4 Section IV

We played 20 times games of human vs your agent but human only win for 6 times.

Percentage of winning time: 30%

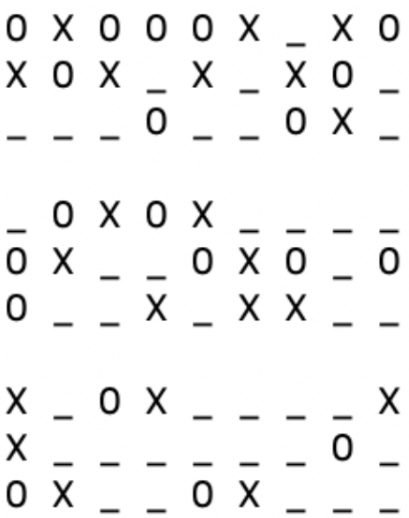
3 representative final game boards: who (maxPlayer/minPlayer) goes first, human is how (defensive/offensive), our agent uses which (minimax/alphabeta).

First type: maxPlayer goes first, human is defensive, our agent uses minimax



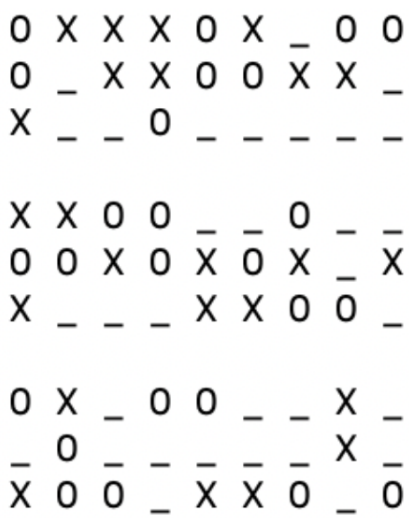
The winner is maxPlayer!!!

Second type: maxPlayer goes first, human is offensive, our agent uses alpha-beta



The winner is minPlayer!!!

Third type: maxPlayer goes first, human is defensive, our agent uses alpha-beta



The winner is maxPlayer!!!

I think the disadvantages of our evaluation function is that the agent we designed always prevent our human player from winning, with this strategy it will always make wise choise while human player has to take much more time on thinking about the next step. But we acknowledge that the testing result might be baised, since we only test it 20 times (already with a long time) and all of our team members haven't touched this game before so that we're unfamiliar and very likely to make stupid choice, letting agent to win.

5 Statement of Contribution:

Zheyang Jia:
1. evaluatePredifined()
2. checkMovesLeft()
3. checkWinner()
4. minimax()
5. playGamePredifinedAgent()

Ke Xu (Dropped):
1. evaluateDesigned()
2. playGameHuman()
3. playGameYourAgent()
4. alphabeta()