MORRIS GAME VARIANT

CS 6364: Artificial Intelligence Project

Abstract

Creation of a variant of the 9-man Morris Game using MiniMax and Alpha/Beta Pruning

Bhattacharya, Abhirup

AXB200027,

abhirup.bhattacharya@utdallas.edu

Examples of Moves On Different Positions

Example 1:

```
py ABOpening.py output/board1.txt output/board1AB0.txt 5
Time Elapsed :  0.061714887619018555 seconds
```

Input Board: XXXXXXXXXXXXXXXXXX

Alpha Beta Output:

```
BoardPosition:                        WXXXXXXXXXXXXXXXXX
Positions Evaluated:   9273
MINIMAX estimate:             1
```

Example 2:

```
py MiniMaxOpening.py output/board1.txt output/board1MM0.txt 5
Time Elapsed :  2.613321304321289 seconds
```

Input Board: XXXXXXXXXXXXXXXXXX

MiniMax Output:

```
BoardPosition:                        WXXXXXXXXXXXXXXXXX
Positions Evaluated:   984480
MINIMAX estimate:             1
```

Example 3:

```
 py MiniMaxOpeningBlack.py output/board2.txt output/board1MM0_B.txt 5
```

Input Board : WBWBWWBWXXXXXXXXXX

MiniMax Black Output:

```
BoardPosition:                        WBWBWWBWBXXXXXXXXX
Positions Evaluated:   10
MINIMAX estimate:             1
```

Example 4:

```
py ABGame.py output/board3.txt output/board3ABG.txt 5
Time Elapsed :  0.02837085723876953 seconds
```

Input Board: WBWBWWBWBBXWBBBXWW

Alpha Beta Game Output:

BoardPosition: XXWBWWBWBBXWBBBWW

Positions Evaluated: 1386

MINIMAX estimate: 2993

Example 5:

py MiniMaxGame.py output/board3.txt output/board3MMG.txt 5

Time Elapsed : 0.2711019515991211 seconds

Input Board : WBWBWWBWBBXWBBBXWW

MiniMax Game Output:

BoardPosition: XXWBWWBWBBXWBBBWW

Positions Evaluated: 9488

MINIMAX estimate: 2993

Example 6:

py MiniMaxGameBlack.py output/board4.txt output/board4MMG_B.txt 5

Time Elapsed : 1.378211259841919 seconds

Input Board: XXWBWWBWBBXWBBBWW

MiniMax Black Game Output:

BoardPosition: XBWBWWBWXBXWBBBWW

Positions Evaluated: 34195

MINIMAX estimate: -5

Example 7:

py ABGame.py output/board5.txt output/board5ABG.txt 5

Time Elapsed : 0.32007265090942383 seconds

Input Board: BWWXWXWXXBXWBBXWWW

Alpha Beta Game Output:

BoardPosition: BWWXWXWXXBXWBBWWW

Positions Evaluated: 11540

MINIMAX estimate: 9999999999999999999999

Example 8:

```
py MiniMaxGame.py output/board5.txt output/board5MMG.txt 5
Time Elapsed :  21.065046310424805 seconds
```

Input Board: BWWXWXWXXBXWBBXWWW

MiniMax Game Output:

```
BoardPosition:                          BWWXWXWXXBXWBBWWW
Positions Evaluated:   391399
MINIMAX estimate:             9999999999999999999999
```

Example 9:

```
py MiniMaxGameImproved.py output/board3.txt output/board3MMG_improved.txt 5
Time Elapsed :  0.29692673683166504 seconds
```

Input Board : WBWBWWBWBBXWBBBXWW

MiniMax Game Improved Output:

```
BoardPosition:                          XXWBWWBWBBXWBBBWW
Positions Evaluated:   9488
MINIMAX estimate:             2988
```

Example 10:

```
py MiniMaxopeningImproved.py output/board6.txt output/board6MMO_improved.txt 5
Time Elapsed :  5.006737470626831 seconds
```

Input Board: WBXXXXXXXXXXXXXXX

MiniMax Opening Improved Output:

```
BoardPosition:                          WBWXXXXXXXXXXXXX
Positions Evaluated:   546802
MINIMAX estimate:             6
```

Alpha Beta Savings Over MiniMax:

Initial Opening Position:

Input Board: XXXXXXXXXXXXXXXXX

Depth Used : 3

MiniMax Output:

BoardPosition: *WXXXXXXXXXXXXXXXX*

Positions Evaluated: ***4624***

MINIMAX estimate: *1*

Alpha Beta Pruning Output:

BoardPosition: *WXXXXXXXXXXXXXXXX*

Positions Evaluated: ***288***

MINIMAX estimate: *1*

Percentage Reduction: 93.77%

Initial Position:

Input Board: WBWBWWBWBBXWBBBXWW

Depth Used : 5

MiniMax Output:

BoardPosition: *XXWBWWBWBBXWBBBWW*

Positions Evaluated: ***9488***

MINIMAX estimate: *2993*

Alpha Beta Pruning Output:

BoardPosition: *XXWBWWBWBBXWBBBWW*

Positions Evaluated: ***1386***

MINIMAX estimate: *2993*

Percentage Reduction : 85.39%

Improved Static Estimation Function v/s Provided Static Estimation Function

Input Board Position:

WBWBXXXXXXXXXXXXXX

Output with Original Static Estimation Function:

```
BoardPosition:                    WXWBWXXXXXXXXXXX
Positions Evaluated:  442831
MINIMAX estimate:           2
```

Output with Improved Static Estimation Function:

```
BoardPosition:                    WBWBXXXXXWXXXXXX
Positions Evaluated:  442831
MINIMAX estimate:           5
```

Input Board Position:

WBWBWWBWBWBWBXBXXW

Output with Original Static Estimation Function:

```
BoardPosition:                    XBWBWWBWBWBWBXBWX
Positions Evaluated:  9711
MINIMAX estimate:           -33
```

Output with Improved Static Estimation Function:

```
BoardPosition:                    XBWBWWBWBWBWBXBWX
Positions Evaluated:  9711
MINIMAX estimate:           -38
```

Improved Static Estimation Function:

Opening: numWhitePieces + possibleMillCounts - numBlackPieces - curr_depth

Midgame/EndGame : 1000*(numWhitePieces - numBlackPieces + possibleMillCounts) - blackMovesCount - curr_depth

Where,

numWhitePieces : Number of white pieces in the position

numBlackPieces : Number of black pieces in the position

possibleMillCounts : Number of potential mills that can be formed in the current position with different possible combinations

blackMovesCount : Number of possible black moves

curr_depth : the depth of the node being evaluated

I have considered 2 more variables while improving the static estimation function i.e., possible Mill Counts, which evaluates all the number of mills counts that can be formed by taking by W in the current position and the current depth of the node that is being evaluated. The current depth is subtracted from the static estimation value, thus giving more incentive to the static estimation value at a lower depth than that at a higher depth. Hence, if there are 2 goal nodes, one at depth 3 and one at depth 5, the original static estimation function can choose any one of them based on the depth first traversal order, but in the improved static estimation function will give a higher value to the one present at depth 3 over the one present at depth 5, thus improving the chances of winning the game in fewer moves.