Artificial Intelligence Assignment

CHECKERS GAME

Ayush Agrawal (B14CS010)

Varun Kumar (B14CS039)

The rules for the game can be found in the following link -

http://www.wcdf.net/rules/rules_of_checkers_english.pdf

Representation of board:-

This is the initial state of the board.

```
Board:
  A B C D E F G H
8|  |w|  |w|  |w|  |w|
7|w|  |w|  |w|  |w|  |
6|  |w|  |w|  |w|  |w|
5|.|  |.|  |.|  |.|  |
4|  |.|  |.|  |.|  |.|
3|b|  |b|  |b|  |b|  |
2|  |b|  |b|  |b|  |b|
1|b|  |b|  |b|  |b|  |

White: 0
Black: 0
```

w -> Computer's pieces (white pieces){King will be denoted by W}

b -> User's pieces (black pieces){King will be denoted by B}

These are the utility of the board with respect to the colours.

Choice of Heuristic :

Utility value for a player is defined as the difference between the sum of that player's pieces value and opponent's pieces value where man's piece value is 1 and king's piece value is 2.

For example, in above board, Utility value for White is ->

Utility = (1*12 (12 white men) + 2*0 (0 white kings)) - (12 (12 black men) + 2*0 (0 black kings)) = 0

This Utility value will be the heuristic for the leaf node.

This is chosen because **the goal is to eliminate opponent pieces and retain our pieces.**

Varying depth :

When we increase depth, the move taken by computer would be more optimistic. This can be seen in Example 1 and 2 below.

Example 1 :

When depth is 2, computer moves (W) 4f -> 3e though it can be attacked by opponent piece (B). But when we increase depth to 7, computer moves (W) 4f->5e preventing it from beaten by the opponent(B).

When depth is 2

Computer Moves : 4f -> 3e

```
  A B C D E F G H
8|  |.|  |.|  |.|  |.|
7|.|  |.|  |.|  |.|  |
6|  |.|  |.|  |.|  |.|
5|.|  |.|  |.|  |.|  |
4|  |.|  |.|  |W|  |.|
3|.|  |.|  |.|  |.|  |
2|  |.|  |.|  |B|  |.|
1|.|  |.|  |.|  |.|  |

White: 0
Black: 0
  A B C D E F G H
8|  |.|  |.|  |.|  |.|
7|.|  |.|  |.|  |.|  |
6|  |.|  |.|  |.|  |.|
5|.|  |.|  |.|  |.|  |
4|  |.|  |.|  |.|  |.|
3|.|  |.|  |W|  |.|  |
2|  |.|  |.|  |B|  |.|
1|.|  |.|  |.|  |.|  |

Passed Time = 0 secs


White: 0
Black: 0
```

When depth is 7

Computer Moves : 4f -> 5e

```
  A B C D E F G H
8|  |.|  |.|  |.|  |.|
7|.|  |.|  |.|  |.|  |
6|  |.|  |.|  |.|  |.|
5|.|  |.|  |.|  |.|  |
4|  |.|  |.|  |W|  |.|
3|.|  |.|  |.|  |.|  |
2|  |.|  |.|  |B|  |.|
1|.|  |.|  |.|  |.|  |

White: 0
Black: 0
  A B C D E F G H
8|  |.|  |.|  |.|  |.|
7|.|  |.|  |.|  |.|  |
6|  |.|  |.|  |.|  |.|
5|.|  |.|  |W|  |.|  |
4|  |.|  |.|  |.|  |.|
3|.|  |.|  |.|  |.|  |
2|  |.|  |.|  |B|  |.|
1|.|  |.|  |.|  |.|  |

Passed Time = 0.068 secs


White: 0
Black: 0
```

Example 2 :

When depth is 3 computer moves (W) 6d->5e. In turn, when user moves (B) 2b->3c, computer moves 6h->5g. But when depth is 7 the second move of computer is 5c->3a. It can be seen that depth 7 moves are more optimistic which make the opponent (B) lose its pieces as shown by red circles.

When Depth is 3 -

Computer Moves : 6d -> 5e

```
  A B C D E F G H
8| |w| |w| |w| |w|
7|w| |w| |w| |w| |
6| |.| |w| |w| |w|
5|.| |.| |.| |.| |
4| |b| |b| |.| |.|
3|.| |.| |b| |b| |
2| |b| |.| |b| |b|
1|b| |b| |b| |b| |

White: 0
Black: 0
  A B C D E F G H
8| |w| |w| |w| |w|
7|w| |w| |w| |w| |
6| |.| |.| |w| |w|
5|.| |.| |w| |.| |
4| |b| |b| |.| |.|
3|.| |.| |b| |b| |
2| |b| |.| |b| |b|
1|b| |b| |b| |b| |

Passed Time = 0.015 secs


White: 0
Black: 0
```

User moves : 2b->3c

```
  A B C D E F G H
8| |w| |w| |w| |w|
7|w| |w| |w| |w| |
6| |.| |.| |w| |w|
5|.| |.| |w| |.| |
4| |b| |b| |.| |.|
3|.| |b| |b| |b| |
2| |.| |.| |b| |b|
1|b| |b| |b| |b| |

White: 0
Black: 0
  A B C D E F G H
8| |w| |w| |w| |w|
7|w| |w| |w| |w| |
6| |.| |.| |w| |.|
5|.| |.| |w| |w| |
4| |b| |b| |.| |.|
3|.| |b| |b| |b| |
2| |.| |.| |b| |b|
1|b| |b| |b| |b| |

Passed Time = 0.014 secs


White: 0
Black: 0
```

Computer Moves : 6h -> 5g

When Depth is 7 -

Computer Moves : 6d -> 5c

```
  A B C D E F G H
8| |w| |w| |w| |w|
7|w| |w| |w| |w| |
6| |.| |w| |w| |w|
5|.| |.| |.| |.| |
4| |b| |b| |.| |.|
3|.| |.| |b| |b| |
2| |b| |.| |b| |b|
1|b| |b| |b| |b| |

White: 0
Black: 0
  A B C D E F G H
8| |w| |w| |w| |w|
7|w| |w| |w| |w| |
6| |.| |.| |w| |w|
5|.| |w| |.| |.| |
4| |b| |b| |.| |.|
3|.| |.| |b| |b| |
2| |b| |.| |b| |b|
1|b| |b| |b| |b| |

Passed Time = 1.801 secs


White: 0
Black: 0
```

User moves : 2b->3c

```
  A B C D E F G H
8| |w| |w| |w| |w|
7|w| |w| |w| |w| |
6| |.| |.| |w| |w|
5|.| |w| |.| |.| |
4| |b| |b| |.| |.|
3|.| |b| |b| |b| |
2| |.| |.| |b| |b|
1|b| |b| |b| |b| |

White: 0
Black: 0
  A B C D E F G H
8| |w| |w| |w| |w|
7|w| |w| |w| |w| |
6| |.| |.| |w| |w|
5|.| |.| |.| |.| |
4| |.| |b| |.| |.|
3|w| |b| |b| |b| |
2| |.| |.| |b| |b|
1|b| |b| |b| |b| |

Passed Time = 2.881 secs


White: 1
Black: -1
```

Computer Moves : 5c -> 3a

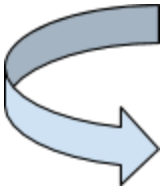
Comparison with and without Alpha Beta Pruning (with constant depth of 5)

MinMax algorithm with alpha beta pruning discards a set of nodes in the search tree, thereby taking less time to compute the next state than MinMax algorithm without alpha beta pruning.

Example 1 :

Without Alpha Beta Pruning

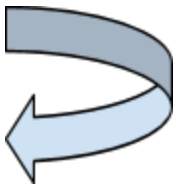
Computer Moves : 6b -> 5a



```
  A B C D E F G H
8| |w| |w| |w| |w|
7|w| |w| |w| |w| |
6| |w| |w| |w| |w|
5|.| |.| |.| |.| |
4| |.| |b| |.| |.|
3|b| |.| |b| |b| |
2| |b| |b| |b| |b|
1|b| |b| |b| |b| |

White: 0
Black: 0
  A B C D E F G H
8| |w| |w| |w| |w|
7|w| |w| |w| |w| |
6| |.| |w| |w| |w|
5|w| |.| |.| |.| |
4| |.| |b| |.| |.|
3|b| |.| |b| |b| |
2| |b| |b| |b| |b|
1|b| |b| |b| |b| |

Passed Time = 0.896 secs

White: 0
Black: 0
```

With Alpha Beta Pruning

Moves Computer : 6b -> 5a



```
  A B C D E F G H
8| |w| |w| |w| |w|
7|w| |w| |w| |w| |
6| |w| |w| |w| |w|
5|.| |.| |.| |.| |
4| |.| |b| |.| |.|
3|b| |.| |b| |b| |
2| |b| |b| |b| |b|
1|b| |b| |b| |b| |

White: 0
Black: 0
  A B C D E F G H
8| |w| |w| |w| |w|
7|w| |w| |w| |w| |
6| |.| |w| |w| |w|
5|w| |.| |.| |.| |
4| |.| |b| |.| |.|
3|b| |.| |b| |b| |
2| |b| |b| |b| |b|
1|b| |b| |b| |b| |

Passed Time = 0.179 secs

White: 0
Black: 0
```

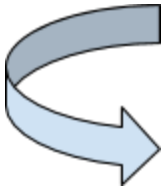
As we can see here, there is not much branching in the initial stage of the game. That's why without Alpha Beta Pruning it doesn't take a **significantly** large time to process (0.896 sec > 0.179 sec). But still, with pruning it takes less time to process the branches.

Example 2 :

Without Alpha Beta Pruning

With Alpha Beta Pruning

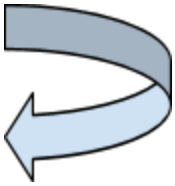
Computer Moved : 6h -> 5g



```
  A B C D E F G H
8| |w| |w| |w| |w|
7|w| |w| |w| |w| |
6| |.| |.| |.| |w|
5|b| |.| |.| |.| |
4| |.| |b| |.| |.|
3|b| |.| |.| |b| |
2| |w| |.| |.| |b|
1|b| |b| |b| |b| |

White: 1
Black: -1
  A B C D E F G H
8| |w| |w| |w| |w|
7|w| |w| |w| |w| |
6| |.| |.| |.| |.|
5|b| |.| |.| |w| |
4| |.| |b| |.| |.|
3|b| |.| |.| |b| |
2| |w| |.| |.| |b|
1|b| |b| |b| |b| |

Passed Time = 10.886 secs

White: 1
Black: -1
```

```
  A B C D E F G H
8| |w| |w| |w| |w|
7|w| |w| |w| |w| |
6| |.| |.| |.| |w|
5|b| |.| |.| |.| |
4| |.| |b| |.| |.|
3|b| |.| |.| |b| |
2| |w| |.| |.| |b|
1|b| |b| |b| |b| |

White: 1
Black: -1
  A B C D E F G H
8| |w| |w| |w| |w|
7|w| |w| |w| |w| |
6| |.| |.| |.| |.|
5|b| |.| |.| |w| |
4| |.| |b| |.| |.|
3|b| |.| |.| |b| |
2| |w| |.| |.| |b|
1|b| |b| |b| |b| |

Passed Time = 0.11 secs

White: 1
Black: -1
```

Computer Moved : 6h -> 5g



As we can see here, there is much more branching option in the later stage of the game. That's why without Alpha Beta Pruning it takes a **significantly** large time to process (10.886 sec >>> 0.11 sec). This is evident from the example above.