CS440 War Game

Dan Pang, Kevin Sullivan, and Alex Paul

To start, we implemented a basic minimax with the bare minimum game board. We imported a 5x5 game board into a 5x5 array of square objects and ran a recursive function to simulate the minimax tree, creating deep copies of the board every time (which slows it down a lot).

To make it simpler to think about, instead of having two separate categories of moves, we thought about each move as simply capturing a single point. Then, we checked to see who occupied the spaces around that point. If the captured point is adjacent to an allied point, all adjacent enemy points are also captured – a death blitz. Otherwise, nothing else happens – a para drop. This made it much easier to visualize the code, simply because we only had one type of move to worry about.

For the evaluation function, the easiest option (and the one we came up with first) was simply to use each player's current score. However, that meant we had a non-zero-sum game that would be slightly tricker to implement. So, instead, we used the difference between the two players' scores as our evaluation function. That way, when one player gained points, the other player lost points equivalently, turning this game into a zero-sum game with a very simple minimax function.

AB pruning was similarly simple to add – it simply checked the tree's minimax score against the current low to see if there needed to be any more points expanded.

In the end, we settled on a minimax depth of three, with or without AB pruning. Because the program is run on Python and because we rely on deep copies, the program runs very slowly and makes for a far less interactive experience for the player for any deeper a minimax tree. Even with AB optimization, the exponential nature of the minimax tree significantly increased calculation times for any higher of a tree depth.

To go above and beyond, we added not only the ability to import different-sized boards (ie, not just 5x5), but we added an interactive GUI that made it easy for a player to interact with the AI, on top of visualizing the game between the two different AI's.

There were a handful of optimizations we tried to add but could not for various reasons. We thought about using heuristics, to look at big play-making squares before other ones, ie, look at squares with large point values, or death-blitzable squares, to try to speed up the code a little. After all, that is how people usually play games like this – we judge what squares are important and focus on those.

To make debugging easier, we made a print function that prints the current state of the board as plain text. The format for each square is:

(Point value, Player)

Kalamazoo:

Minimax:

Green Player:

Score: 20

Nodes expanded: 51454

Blue Player:

Score: 15

Nodes expanded: 43796

Average nodes per move: 95250

Average time per move: 3.5552 sec

End board state:

(1, 'blue')	(1, 'green')	(1, 'green')	(1, 'green')	(1, 'green')
(1, 'blue')	(2, 'blue')	(2, 'green')	(2, 'green')	(1, 'green')
(1, 'blue')	(2, 'green')	(3, 'blue')	(2, 'green')	(1, 'green')
(1, 'green')	(2, 'green')	(2, 'blue')	(2, 'blue')	(1, 'green')
(1, 'green')	(1, 'blue')	(1, 'blue')	(1, 'blue')	(1, 'green')

AB-pruning:

Green Player:

Score: 20

Nodes expanded: 24662

Blue Player:

Score: 15

Nodes expanded: 23598

Average nodes per move: 1930.4

Average time per move: 1.7756 sec

End board state:

(1, 'blue')	(1, 'green')	(1, 'green')	(1, 'green')	(1, 'green')
(1, 'blue')	(2, 'blue')	(2, 'green')	(2, 'green')	(1, 'green')
(1, 'blue')	(2, 'green')	(3, 'blue')	(2, 'green')	(1, 'green')
(1, 'green')	(2, 'green')	(2, 'blue')	(2, 'blue')	(1, 'green')
(1, 'green')	(1, 'blue')	(1, 'blue')	(1, 'blue')	(1, 'green')

Peoria:

Minimax:

Green Player:

Score: 699

Nodes expanded: 51454

Blue Player

Score: 600

Nodes expanded: 43796

Average nodes per move: 3810

Average time per move: 3.5464 sec

End board state:

(99, 'blue')	(1, 'blue')	(99, 'green')	(1, 'green')	(99, 'green')
(1, 'blue')	(99, 'blue')	(1, 'blue')	(99, 'green')	(1, 'green')
(99, 'blue')	(1, 'blue')	(99, 'green')	(1, 'green')	(99, 'green')
(1, 'blue')	(99, 'blue')	(1, 'green')	(99, 'green')	(1, 'green')
(99, 'blue')	(1, 'blue')	(99, 'blue')	(1, 'green')	(99, 'green')

AB-pruning:

Green Player:

Score: 699

Nodes expanded: 13299

Blue Player:

Score: 600

Nodes expanded: 14223

Average nodes per move: 1100.88

Average time per move: 1.0228 sec

End board state:

(99, 'blue')	(1, 'blue')	(99, 'green')	(1, 'green')	(99, 'green')
(1, 'blue')	(99, 'blue')	(1, 'blue')	(99, 'green')	(1, 'green')
(99, 'blue')	(1, 'blue')	(99, 'green')	(1, 'green')	(99, 'green')
(1, 'blue')	(99, 'blue')	(1, 'green')	(99, 'green')	(1, 'green')
(99, 'blue')	(1, 'blue')	(99, 'blue')	(1, 'green')	(99, 'green')

Piqua:

Minimax:

Green Player:

Score: 14

Nodes expanded: 51454

Blue Player

Score: 11

Nodes expanded: 43796

Average nodes per move: 3810

Average time per move: 3.5852 sec

End board state:

(1, 'blue')	(1, 'green')	(1, 'green')	(1, 'green')	(1, 'green')
(1, 'blue')	(1, 'blue')	(1, 'green')	(1, 'blue')	(1, 'green')
(1, 'blue')	(1, 'green')	(1, 'blue')	(1, 'green')	(1, 'green')
(1, 'blue')	(1, 'blue')	(1, 'green')	(1, 'blue')	(1, 'green')
(1, 'blue')	(1, 'green')	(1, 'blue')	(1, 'green')	(1, 'green')

AB-pruning:

Green Player:

Score: 14

Nodes expanded: 10656

Blue Player:

Score: 11

Nodes expanded: 13063

Average nodes per move: 948.76

Average time per move: 0.8764 sec

End board state:

(1, 'blue')	(1, 'green')	(1, 'green')	(1, 'green')	(1, 'green')
(1, 'blue')	(1, 'blue')	(1, 'green')	(1, 'blue')	(1, 'green')
(1, 'blue')	(1, 'green')	(1, 'blue')	(1, 'green')	(1, 'green')
(1, 'blue')	(1, 'blue')	(1, 'green')	(1, 'blue')	(1, 'green')
(1, 'blue')	(1, 'green')	(1, 'blue')	(1, 'green')	(1, 'green')

Punxsutawney:

Minimax:

Green Player:

Score: 53

Nodes expanded: 51454

Blue Player

Score: 42

Nodes expanded: 43796

Average nodes per move: 3810

Average time per move: 3.53 sec

End board state:

(10, 'green')	(5, 'green')	(2, 'green')	(1, 'green')	(1, 'green')
(10, 'blue')	(5, 'green')	(2, 'blue')	(1, 'green')	(1, 'blue')
(10, 'green')	(5, 'blue')	(2, 'green')	(1, 'blue')	(1, 'blue')
(10, 'blue')	(5, 'green')	(2, 'blue')	(1, 'green')	(1, 'blue')
(10, 'green')	(5, 'blue')	(2, 'blue')	(1, 'blue')	(1, 'blue')

AB-pruning:

Green Player:

Score: 53

Nodes expanded: 10858

Blue Player:

Score: 42

Nodes expanded: 13074

Average nodes per move: 957.28

Average time per move: 0.8828 sec

End board state:

(10, 'green')	(5, 'green')	(2, 'green')	(1, 'green')	(1, 'green')
(10, 'blue')	(5, 'green')	(2, 'blue')	(1, 'green')	(1, 'blue')
(10, 'green')	(5, 'blue')	(2, 'green')	(1, 'blue')	(1, 'blue')
(10, 'blue')	(5, 'green')	(2, 'blue')	(1, 'green')	(1, 'blue')
(10, 'green')	(5, 'blue')	(2, 'blue')	(1, 'blue')	(1, 'blue')

Wallawalla:

Minimax:

Green Player:

Score: 38

Nodes expanded: 51454

Blue Player

Score: 38

Nodes expanded: 43796

Average nodes per move: 3810

Average time per move: 3.5996 sec

End board state:

(1, 'blue')	(10, 'blue')	(1, 'green')	(1, 'green')	(2, 'green')
(10, 'blue')	(1, 'blue')	(2, 'green')	(5, 'green')	(10, 'green')
(5, 'blue')	(1, 'blue')	(1, 'green')	(3, 'green')	(7, 'green')
(2, 'blue')	(1, 'green')	(1, 'green')	(1, 'blue')	(3, 'green')
(1, 'green')	(2, 'blue')	(2, 'blue')	(1, 'blue')	(2, 'blue')

AB-pruning:

Green Player:

Score: 38

Nodes expanded: 13602

Blue Player:

Score: 38

Nodes expanded: 13556

Average nodes per move: 1086.32

Average time per move: 1.0204 sec

End board state:

(1, 'blue')	(10, 'blue')	(1, 'green')	(1, 'green')	(2, 'green')
(10, 'blue')	(1, 'blue')	(2, 'green')	(5, 'green')	(10, 'green')
(5, 'blue')	(1, 'blue')	(1, 'green')	(3, 'green')	(7, 'green')
(2, 'blue')	(1, 'green')	(1, 'green')	(1, 'blue')	(3, 'green')
(1, 'green')	(2, 'blue')	(2, 'blue')	(1, 'blue')	(2, 'blue')

4 by 4 custom:

Minimax:

Green Player:

Score: 14

Nodes expanded: 8888

Blue Player

Score: 6

Nodes expanded: 6904

Average nodes per move: 987

Average time per move: 0.61875 sec

End board state:

(1, 'green')	(0, 'blue')	(1, 'green')	(0, 'green')
(0, 'blue')	(2, 'blue')	(0, 'blue')	(2, 'green')
(3, 'green')	(0, 'blue')	(3, 'green')	(0, 'green')
(0, 'blue')	(4, 'green')	(0, 'green')	(4, 'blue')

AB-pruning:

Green Player:

Score: 14

Nodes expanded: 6462

Blue Player:

Score: 6

Nodes expanded: 5203

Average nodes per move: 729.0625

Average time per move: 0.456875 sec

End board state:

(1, 'green')	(0, 'blue')	(1, 'green')	(0, 'green')
(0, 'blue')	(2, 'blue')	(0, 'blue')	(2, 'green')
(3, 'green')	(0, 'blue')	(3, 'green')	(0, 'green')
(0, 'blue')	(4, 'green')	(0, 'green')	(4, 'blue')

6 by 6 custom:

Minimax:

Green Player:

Score: 94

Nodes expanded: 217758

Blue Player

Score: 71

Nodes expanded: 194754

Average nodes per move: 11458.67

Average time per move: 14.990278 sec

End board state:

(1, 'blue')	(9, 'blue')	(2, 'blue')	(1, 'blue')	(7, 'blue')	(3, 'green')
(3, 'blue')	(4, 'blue')	(2, 'blue')	(0, 'blue')	(6, 'blue')	(8, 'blue')
(3, 'green')	(8, 'blue')	(9, 'blue')	(5, 'blue')	(6, 'blue')	(7, 'green')
(5, 'green')	(2, 'green')	(9, 'green')	(4, 'green')	(5, 'green')	(4, 'green')
(2, 'green')	(3, 'green')	(9, 'green')	(0, 'blue')	(1, 'green')	(9, 'green')
(3, 'green')	(9, 'green')	(4, 'green')	(3, 'green')	(1, 'green')	(8, 'green')

AB-pruning:

Green Player:

Score: 94

Nodes expanded: 80001

Blue Player:

Score: 71

Nodes expanded: 82476

Average nodes per move: 4513.25

Average time per move: 5.85583 sec

End board state:

(1, 'blue')	(9, 'blue')	(2, 'blue')	(1, 'blue')	(7, 'blue')	(3, 'green')
(3, 'blue')	(4, 'blue')	(2, 'blue')	(0, 'blue')	(6, 'blue')	(8, 'blue')
(3, 'green')	(8, 'blue')	(9, 'blue')	(5, 'blue')	(6, 'blue')	(7, 'green')
(5, 'green')	(2, 'green')	(9, 'green')	(4, 'green')	(5, 'green')	(4, 'green')
(2, 'green')	(3, 'green')	(9, 'green')	(0, 'blue')	(1, 'green')	(9, 'green')
(3, 'green')	(9, 'green')	(4, 'green')	(3, 'green')	(1, 'green')	(8, 'green')