CSCI 6660 Fall 2020

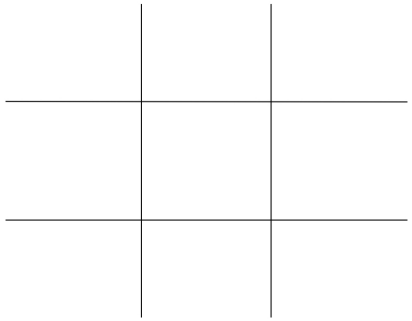
Tic-Tac-Toe Agents

By: Katrina Curro

Project Goals

Create agents to play on 3 different sized tic-tac-toe boards…

- random move agent
- basic strategy agent
- minimax agent
- minimax with alpha beta pruning agent



Data is observed for 10 runs:

- Number of nodes expanded in the game
- Number of times the agent won

Approach

4 different agents created

Project Controls:

- Opponent for all agents: Random Agent
- Agents always had the first move and played as X

Agent 1 – Random Agent

Game is played with the following:

Agent: Random Agent
Computer: Random Agent

Both choose a random location on the board

```
if depth == 9:
    x = choice([0, 1, 2])
    y = choice([0, 1, 2])
else:
    l = []

    for first in range(len(board)):
        for second in range(len(board)):
            if board[first][second] == 0:
                l.append((first, second))

    move = random.choice(l)
    x, y = move[0], move[1]
    expanded = 1

validMove(x, y, AGENT)
```

Agent 1- Findings

Small Board 3x3			
Round Number	Win/Loss/Tie	Number of Nodes Expanded	Execution Time (s)
1	W	4	6.03
2	W	6	7.48
3	L	5	6.48
4	W	6	7.38
5	L	5	6.4
6	W	6	7.49
7	W	6	7.45
8	T	8	9.24
9	L	5	6.46
10	W	8	9.45
Avg		5.9	7.386

Medium Board 4x4			
Round Number	Win/Loss/Tie	Number of Nodes Expanded	Execution Time (s)
1	W	10	11.69
2	W	10	11.57
3	L	13	14.56
4	T	15	16.57
5	L	15	16.83
6	W	12	14.5
7	T	15	16.47
8	L	15	16.49
9	W	12	13.52
10	W	10	11.47
Avg		12.7	14.367

Large Board 5x5			
Round Number	Win/Loss/Tie	Number of Nodes Expanded	Execution Time (s)
1	T	24	26.04
2	L	21	22.67
3	T	24	25.77
4	T	24	25.74
5	T	24	25.45
6	W	24	25.46
7	T	24	25.92
8	W	24	25.61
9	W	22	23.6
10	T	24	25.78
Avg		23.5	25.204

Number of Wins Based on Board Size:

Small Board: 6 -> 60%
Medium Board: 5 -> 50%
Large Board: 3 -> 30%

Agent 2 – Basic Game Play Agent

Game is played with the following:

Agent: Basic Game Play
Computer: Random Agent

Agent will play 4 corners first then will play a random locations

Computer chooses a random location on the board

```
if validMove(0, 0, AGENT):
    x, y = 0, 0
    expanded = 1
elif validMove(0, 2, AGENT):
    x, y = 0, 2
    expanded = 1
elif validMove(2, 0, AGENT):
    x, y = 2, 0
    expanded = 1
elif validMove(2, 2, AGENT):
    x, y = 2, 2
    expanded = 1

else:
    l = []

    for i in range(len(board)):
        for j in range(len(board)):

            if board[i][j] == 0:
                l.append((i, j))

    move = random.choice(l)
    x, y = move[0], move[1]
    expanded = 1
```

Agent 2- Findings

Small Board 3x3			
Round Number	Win/Loss/Tie	Number of Nodes Expanded	Execution Time (s)
1	L	6	6.51
2	L	8	8.58
3	W	7	7.28
4	L	8	8.92
5	W	9	9.56
6	W	9	9.37
7	L	6	6.36
8	W	7	7.31
9	W	7	7.35
10	W	7	7.48
Avg		7.4	7.872

Medium Board 4x4			
Round Number	Win/Loss/Tie	Number of Nodes Expanded	Execution Time (s)
1	W	15	15.45
2	T	16	16.47
3	T	16	16.44
4	W	15	15.36
5	W	11	11.3
6	W	15	15.26
7	T	16	16.29
8	T	16	16.25
9	W	13	13.4
10	W	13	13.4
Avg		14.6	14.962

Large Board 5x5			
Round Number	Win/Loss/Tie	Number of Nodes Expanded	Execution Time (s)
1	T	25	25.61
2	T	25	25.52
3	T	25	25.41
4	W	25	25.38
5	T	25	25.31
6	L	20	20.23
7	L	24	24.25
8	L	20	20.3
9	T	25	25.22
10	L	24	24.38
Avg		23.8	24.161

Number of Wins Based on Board Size:

Small Board: 6 -> 60%
Medium Board: 6 -> 60%
Large Board: 1 -> 10%

Agent 3 – Minimax Agent

Game is played with the following:

Agent: Minimax Agent
Computer: Random Agent

Agent will play using the minimax algorithm

Computer chooses a random location on the board

```
if player == AGENT:
    best = [-1, -1, -infinity, expanded]
    for cell in availablePositions(state):
        x, y = cell[0], cell[1]
        state[x][y] = player
        score = minimax(state, depth - 1,
changePlayer(player), expanded)
        state[x][y] = 0
        score[0], score[1] = x, y
        if score[2] > best[2]:
            temp = max(score, best)
            best = temp
        expanded += 1
        best[3] = expanded
    return best
```

Agent 3- Findings

Small Board 3x3			
Round Number	Win/Loss/Tie	Number of Nodes Expanded	Execution Time (s)
1	L	38	8.72
2	W	54	9.32
3	W	54	9.28
4	T	54	10.42
5	W	21	5.49
6	L	22	6.31
7	W	21	5.4
8	W	37	7.32
9	T	54	9.29
10	W	54	9.31
Avg		40.9	8.086

Large Board 5x5	max depth 5		
Round Number	Win/Loss/tie	Number of Nodes Expanded	Execution Time (s)
1	L	946	108.68
2	L	805	103.33
3	T	1234	109.21
4	L	1090	109.94
5	T	1234	110.68
6	T	1234	114.04
7	L	946	109.4
8	L	1090	109.43
9	W	1234	111.67
10	T	1234	111.53
Avg		1104.7	109.791

Medium Board 4x4	max depth 7		
Round Number	Win/Loss/tie	Number of Nodes Expanded	Execution Time (s)
1	L	231	195.03
2	W	287	189.68
3	L	231	194.13
4	W	287	217.23
5	L	288	195.86
6	T	288	194.99
7	L	288	189.71
8	L	288	190.66
9	L	288	194.18
10	L	288	192.93
Avg		276.4	195.44

Number of Wins Based on Board Size:

Small Board: 6 -> 60%
Medium Board: 2 -> 20%
Large Board: 1 -> 10%

Agent 4 – Minimax Agent with Alpha Beta Pruning

Game is played with the following:

Agent: Minimax Agent utilizing alpha beta pruning

Computer: Random Agent

Agent will play using the minimax algorithm with alpha beta pruning

Computer chooses a random location on the board

```
if player == AGENT:
    best = [-1, -1, -infinity, expanded]
    for cell in availablePositions(state):
        x, y = cell[0], cell[1]
        state[x][y] = player
        score = alphaBeta(state, depth - 1,
changePlayer(player), alpha, beta, expanded)
        state[x][y] = 0
        score[0], score[1] = x, y

        if score[2] > best[2]:
            score[2] = max(score[2], best[2])
            temp = [score[0], score[1], score[2], score[3]]
            best = temp

        expanded += 1
        best[3] = expanded

        if score[2] > beta:
            return best

        alpha = max(alpha, best[2])
    return best
```

Agent 4- Findings

Small Board 3x3			
Round Number	Win/Loss/tie	Number of Nodes Expanded	Execution Time (s)
1	L	19	8.41
2	W	14	5.29
3	L	19	8.3
4	W	20	9.27
5	W	18	7.43
6	W	18	7.22
7	W	18	7.19
8	W	18	7.23
9	L	19	8.23
10	W	14	5.17
Avg		17.7	7.374

Large Board 5x5	max depth 5		
Round Number	Win/Loss/tie	Number of Nodes Expanded	Execution Time (s)
1	L	145	109.77
2	T	156	118.96
3	L	155	117.31
4	T	156	118.36
5	T	156	117.66
6	L	145	111.37
7	W	150	113.12
8	T	156	117.14
9	W	150	113.15
10	L	145	114.75
Avg		151.4	115.159

Medium Board 4x4	max depth 7		
Round Number	Win/Loss/tie	Number of Nodes Expanded	Execution Time (s)
1	L	64	201.25
2	L	49	201.11
3	L	64	203.14
4	L	61	202.15
5	L	64	209.39
6	L	61	210.12
7	L	64	206.26
8	L	64	209.55
9	T	64	202.37
10	L	61	206.4
Avg		61.6	205.174

Number of Wins Based on Board Size:

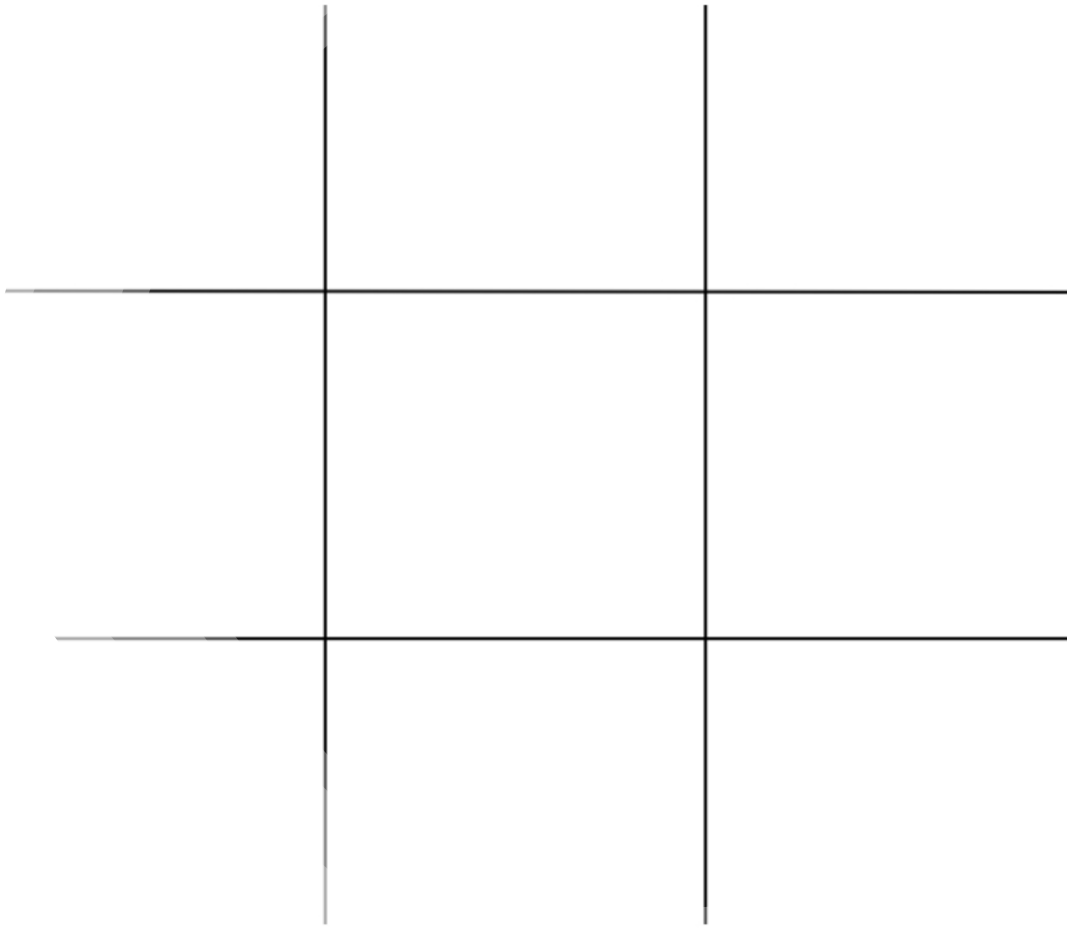
Small Board: 7 -> 70%
Medium Board: 0 -> 0%
Large Board: 2 -> 20%

Notable Comparisons

- Who expands the least number of nodes?
  - Random Agent

- Agent 3 vs Agent 4
  - Agent 4 expands significantly less nodes than Agent 3
  - Ex. Large board 5x5
    - Agent 3 expands: 1104.7 nodes
    - Agent 4 expands: 151.4 nodes

- Best win percentages:
  - Small board: Agent 4
  - Medium board: Agent 2
  - Large board: Agent 1



Insight:

When designing agents, do not always assume the opponent plays optimally

The larger the state space (in this case board size) becomes we have exponentially more possible combinations

In cases of large state spaces, we need to limit the depth we explore or else we will be computing for what feels like forever