03/12/2024

Forward Chaining

Q. As per the law, it is a crime for an American to sell weapons to hostile nations. Country A, an enemy of America, has some missiles, and all the missiles were sold to it by Robert, who is an American citizen

Prove that Robert is a Criminal.

→ Proof:

Crime for an American to sell weapons to hostile nations

$Weapon(y) \wedge American(x) \wedge Sells(x, y, z) \wedge Enemy(America, z) \Rightarrow Criminal(x)$

Country A is an enemy of America

Enemy (America, A)

Robert sold missiles to A

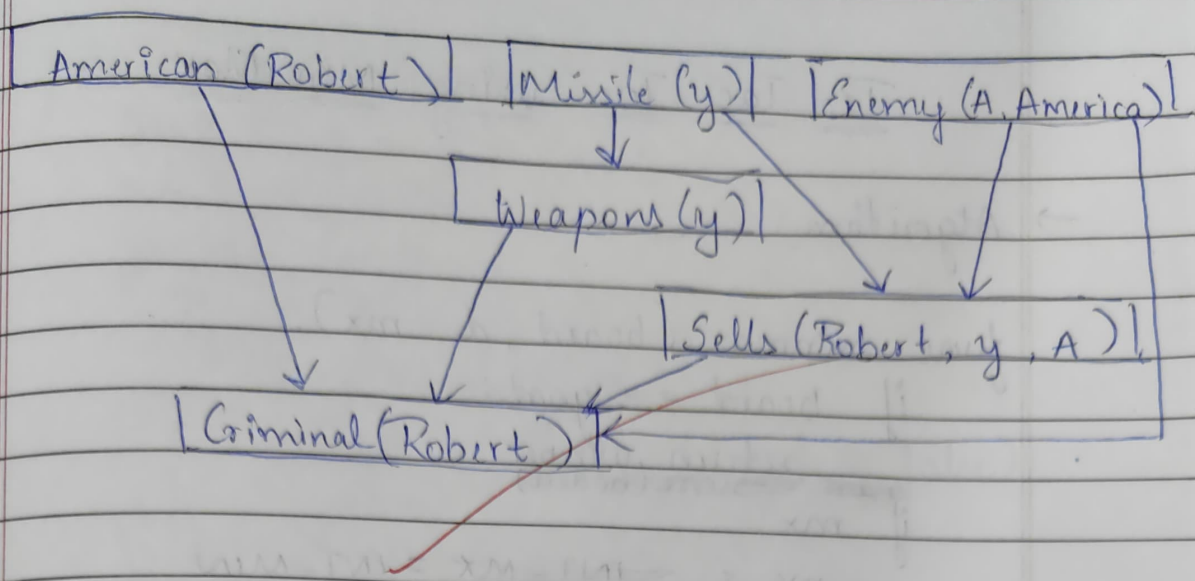
Sells (Robert, y, A) ∧ Missiles (y)

All missiles are weapons

Missile (y) ⇒ Weapon (y)

Robert is American

American (Robert)


American (Robert)
Missile (y)
Enemy (A, America)
Weapons (y)
Sells (Robert, y, A)
Criminal (Robert)

Tic Tac Toe using Min Max

→ Algorithm:

```
func minmax (board, d, mx):
    if board == goal:
        return winner
    if mx:
        ms = ~~-INT_MX~~ = INT_MIN
        for cell in board:
            if cell is empty:
                move(board, cell, 'x')
                score = minmax(board, d+1, fal
                undo()
                ms = max(ms, score)
        return ms
    else:
        ms = INT_MAX
        for cell in board:
            if cell is empty:
                move(board, cell, 'o')
                score = minmax(board, d+1, true)
                undo()
                ms = min(ms, score)
        return ms
```

```
func makeMove(board):
    best = -1
    ms = INT_MIN
    for cell in board:
        if cell is empty:
            move(board, cell, 'x')
            score = minmax(board, 0, false)
            undo()
        if score > ms:
            ms = score
            best = cell
    return cell
```

→ Output

Row & Col : 1, 1

X		
	O	

Row & col : 1, 0

X		
O	O	X

Row & col : 2, 0

X		X
O	O	X
O		

Row & col : 2, 2

X	X	X
O	O	X
O		O

AI wins!

8 Queens using Alpha - Beta Pruning

→ Algorithm:

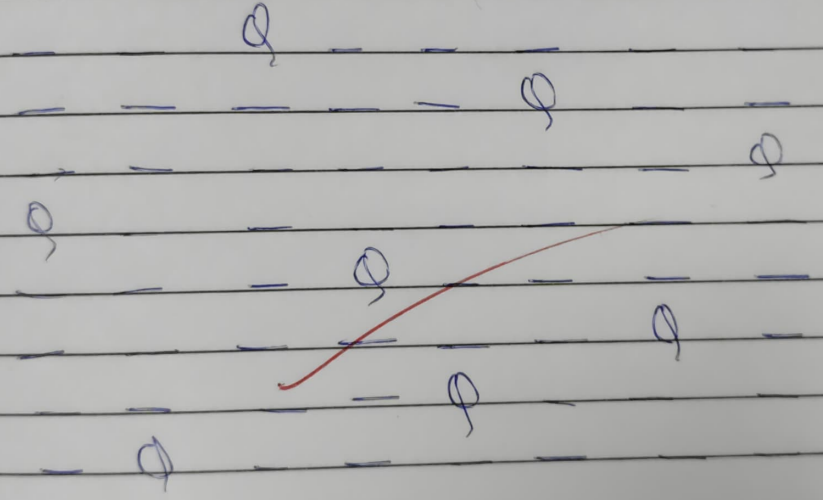
```
func is_valid (board, r, c):
    for i in range(r):
        if board[i] == c or abs(board[i] - c) == abs(i
            return 0
    ← return 1

def alpha_beta (board, r, alpha, beta, mx):
    if (r == len(board)):
        return 1
    if mx:
        ms = 0
        for c in range (len (board)):
            if is_valid (board, r, c):
                board[r] = c
                ms += alpha_beta(board, r+1, alpha,
                                 beta, 0)
                board [r] = -1
                alpha = max (alpha, ms)
                if beta <= alpha:
                    break
        return ms
    else:
        ms = INT_MAX
        for col in range (len (board))
            if is_valid (board, r, c):
                board[r] = c
                ms = min (ms, alpha_beta (board, r+1,
                                          alpha, beta, 1))
                board [r] = -1
```

```
            beta = min(beta, ms)
            if beta <= alpha:
                break
        return ms

func queens()
    board = [-1] * 8
    alpha = -INT_MAX
    beta = INT_MAX
    return alpha_beta(board, 0, alpha, beta, True)
```

Output:



3/12/24