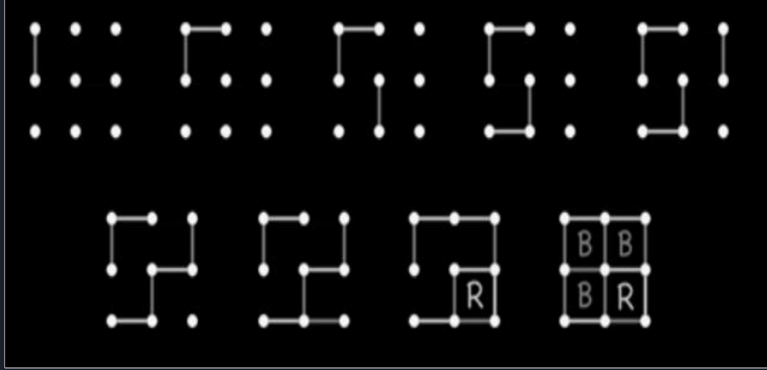
Machine Learning
BATCH 8

Deepthishree G S 18Z312
Harshini S 18Z320
Iswaryaa GP 18Z323
Janani R 18Z324
Swetha M 18Z360

Topic: Reinforcement Learning

Problem Statement



- ➔ Topic: **Dots and Boxes**
- ➔ To train an agent to play a simple pen and pencil game **Dots and Boxes** using a reinforcement learning (Q-learning Algorithm)
- ➔ To make the agent learn the game from the rewards and penalties
- ➔ To check its performance by playing against human and random agents
- ➔ To deepen the understanding of Q Learning algorithm by building it from scratch

Why RL for this problem?

- ➔ There is no training data available beforehand
- ➔ The agent improves by playing more and more games
- ➔ The more the games, the better the agent learns from the rewards and penalties from environment
- ➔ Since, the agent learns the best move to from boxes based on the rewards, reinforcement learning will be best for the problem

Problem Design

Task: Play Dots and Boxes

Performance: Win-rate against itself, humans and a random agent

Training Experience: Play 10000 Games against itself

Target Function: Policy : State -> Action

Target Function representation:

Q Table Q(s,a)=Immediate Reward + DR * V(Next State)

Policy is to choose the action with maximum Q value for the given state

(Reference for Equations: Tom Mitchell Ch 13)

Assumptions and representations

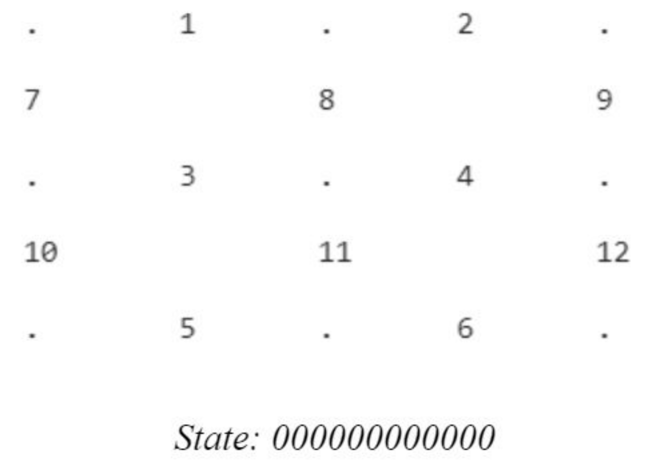
- 3x3 grid with 4 boxes. 2 player game
- Player with greater number of boxes wins. There is a possibility of tie (unlike a few variations of the game)
- Assign Line numbers from 1 to 12 to accept input and represent the board state.
- Board state is represented as bit array.

Eg: 000000000000 is the initial state and 010101010101 is an intermediary state with 6 lines drawn

- Action is a number from 1 to 12 indicating legal moves from a state.

Eg: in the above mentioned state 1, 12 possible actions {1, 2, 3, … 12 } and state 2, only six possible actions {1, 3, 5, 7, 9, 11}

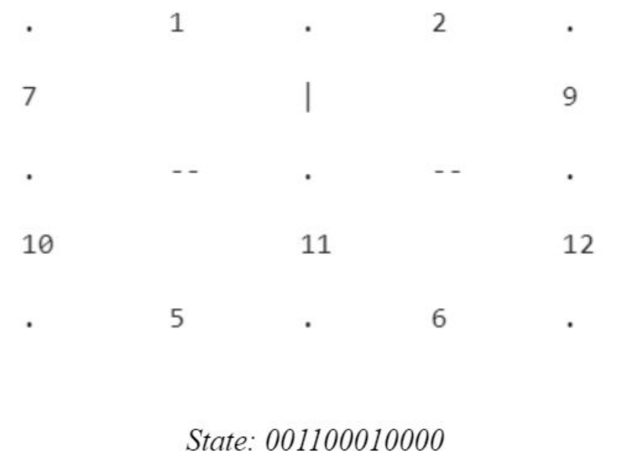
Board States



```
.       1       .       2       .

7               8               9

.       3       .       4       .

10              11              12

.       5       .       6       .
```

State: 000000000000

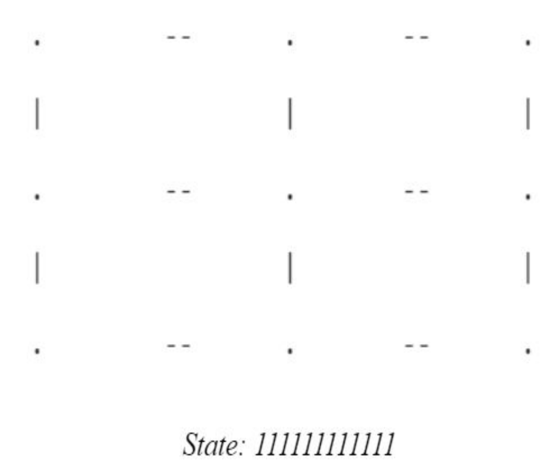
Initial board state



```
.       1       .       2       .

7               |               9

.       --      .       --      .

10              11              12

.       5       .       6       .
```

State: 001100010000

Intermediate board state



```
.       --      .       --      .

|               |               |

.       --      .       --      .

|               |               |

.       --      .       --      .
```

State: 111111111111

Final board state

Design Choices

- ➔ **Learning Rate**=0.2 [Each update of QTable affects learning slowly]
- ➔ **Discount Rate**=0.8 [Importance of future rewards compared to immediate rewards]
- ➔ **Reward:** 200 for winner, 100 for each box filled (only 4 boxes need to be filled), 50 for tie (more common in this game)
- ➔ **Penalty**: 200 for loser, 100 for each box opponent fills
- ➔ **Score:** 2 for each box completed
- ➔ **Exploration** (50 %) using random moves and visiting states with least visit count and then **Exploitation** using Q Table

Results and Inference

➔ Against Random agent:

(1000 games)

- Win-rate = 0.921
- Tie-rate = 0.064
- Lose-rate = 0.015

➔ Against Human:

(10 games)

- Win-rate = 0.3
- Tie-rate = 0.5
- Lose-rate = 0.2

THANK YOU!