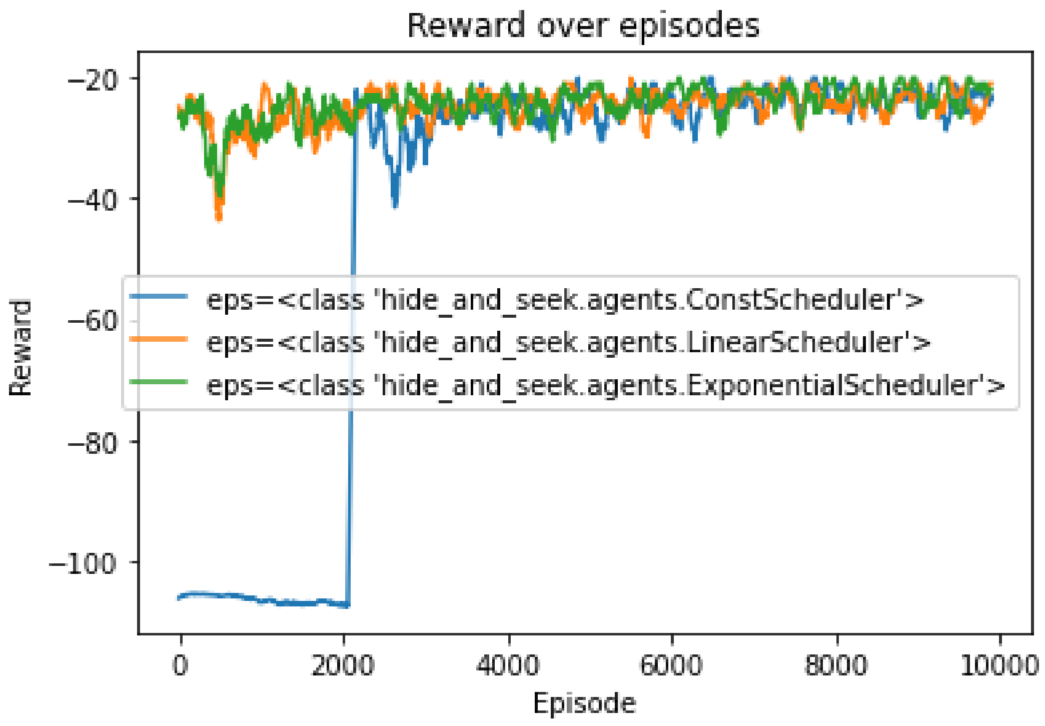
MAS HW1

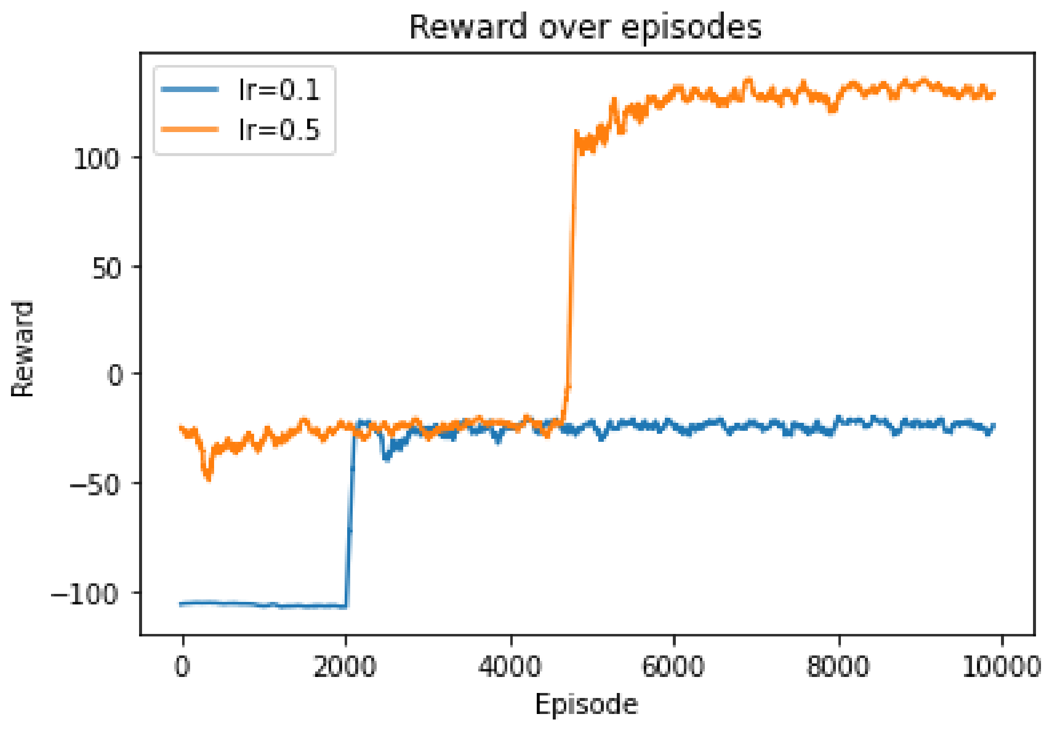
A Game of Hide and Seek

Q Learning

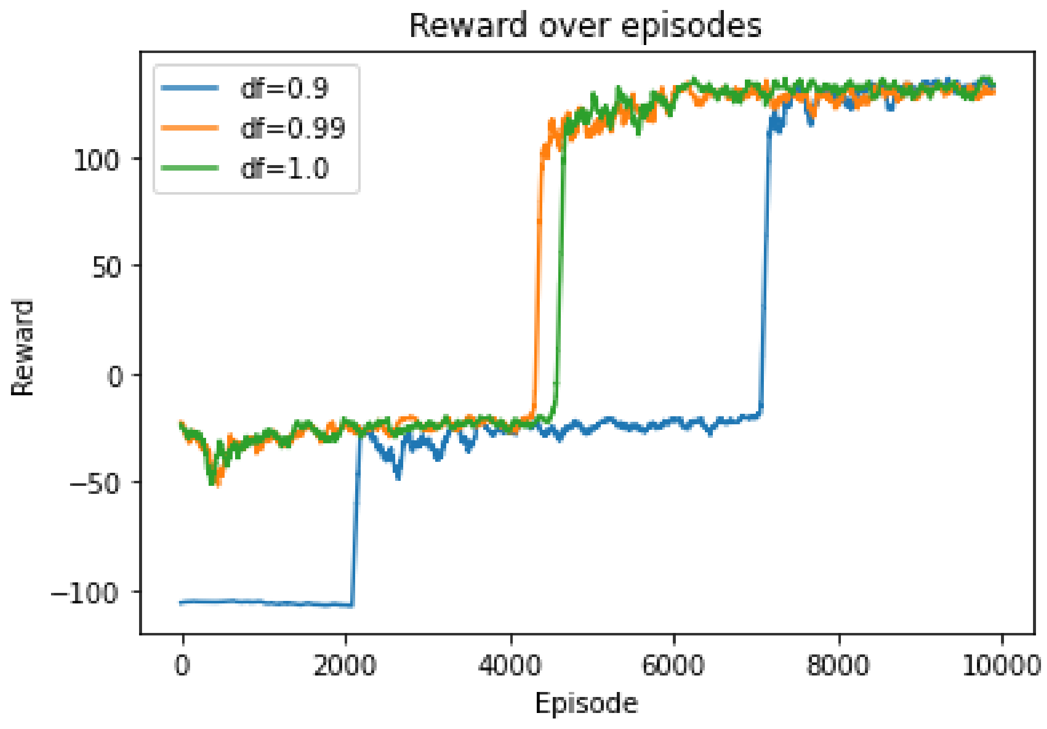
The reward of the Hider agents was better on average when using non constant scheduler for epsilon. The best value overall was obtained for the Exponential Scheduler.



In combination with the Exponential Scheduler, a higher learning rate (of 0.5) also resulted in better results.

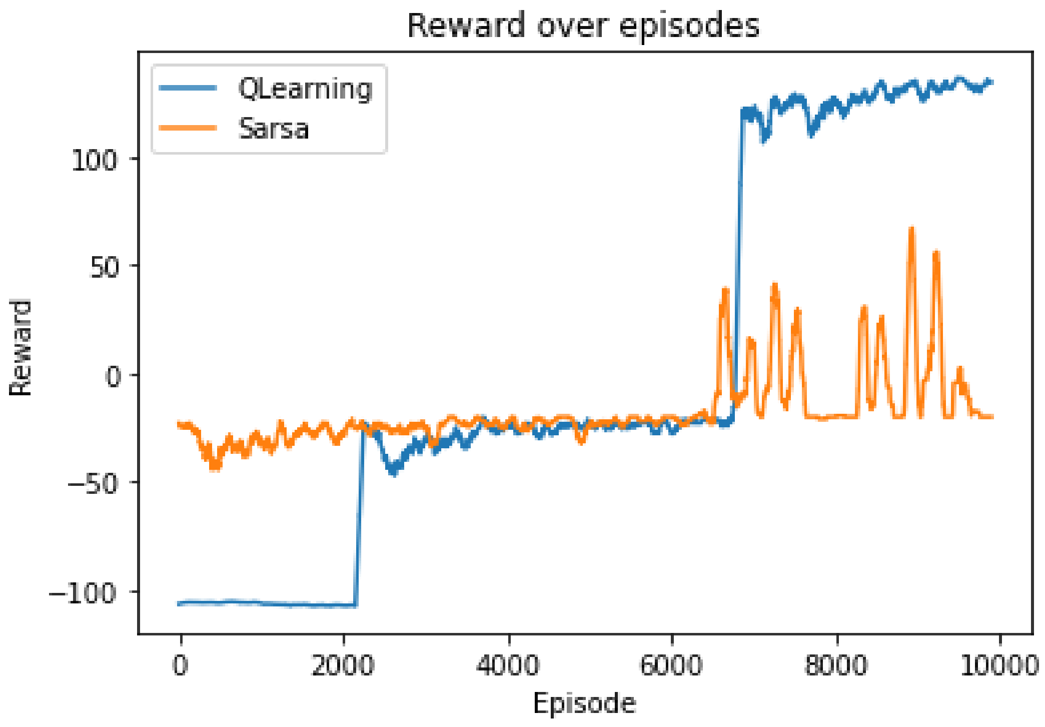


Finally, a higher discount factor is preferred for this type of environment. This makes the agent focus more on winning the game rather than on the immediate reward. In this case, hiding successfully gives 100 points, but each move gives -0.5.



Sarsa

For the Sarsa algorithm, the results are very similar, and the best parameters are the same. However, only the Q Learning agent manages to win the game and successfully hide from the seeker agents consistently.



The Observant Maze Runner

POMDP Model

Prob. (LEFT)	SL	SL1	G	SR1	SR
SL	1	0	0	0	0
SL1	1	0	0	0	0
G	0	1	0	0	0
SR1	0	0	1	0	0
SR	0	0	0	1	0

Prob. (RIGHT)	SL	SL1	G	SR1	SR
SL	0	1	0	0	0
SL1	0	0	1	0	0
G	0	0	0	1	0
SR1	0	0	0	0	1
SR	0	0	0	0	1

The Transition Matrix will contain the probabilities to go from one state to another using the action. In this case the probabilities will be 1, because the environment is deterministic.

Prob. (LEFT)	O: 3	O: 2
SL	1	0
SL1	1	0
G	0	1
SR1	0	1
SR	0	1

Prob. (RIGHT)	O: 3	O: 2
SL	0	1
SL1	0	1
G	0	1
SR1	1	0
SR	1	0

The Observation Matrix will contain the probabilities to observe either 2 wall or 3 walls when taking an action from a state.

Reward (LEFT)	
SL	-1
SL1	-1
G	-1
SR1	0
SR	-1

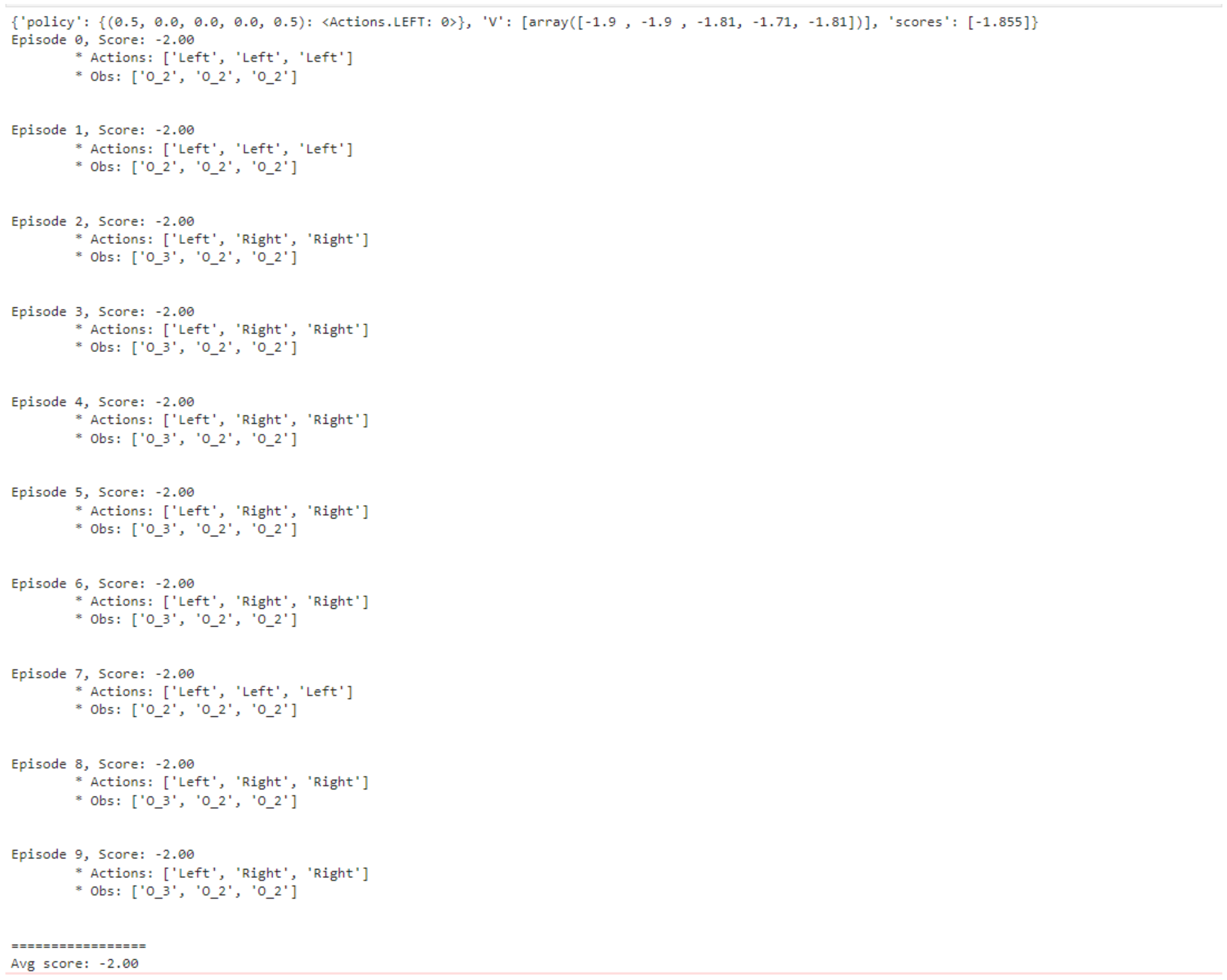
Reward (RIGHT)	
SL	-1
SL1	0
G	-1
SR1	-1
SR	-1

The Reward Matrix will contain the rewards for taking an action from a state. The rewards are -1 reward for moves and 0 reward for winning. For example, going to the left from SR1 results in the agent moving to the goal state G.

Best Policy

The initial state of the agent is to spawn in either the left most cell or the right most cell with equal probabilities.

The best policy learned using Point-Based Value Iteration is to start by taking the action LEFT. Because the agent cannot go out of bounds there are two cases, it will observe two walls or three walls. If the agent observes two walls, then it knows that it was spawned in the right most cell and will continue to the LEFT, reaching the goal state in two steps. If the agent observes three walls, then it knows that it was spawned in the left most cell and will move to the RIGHT two times, reaching the goal state in three moves.



```
{'policy': {(0.5, 0.0, 0.0, 0.0, 0.5): <Actions.LEFT: 0>}, 'V': [array([-1.9 , -1.9 , -1.81, -1.71, -1.81])], 'scores': [-1.855]}
Episode 0, Score: -2.00
        * Actions: ['Left', 'Left', 'Left']
        * Obs: ['O_2', 'O_2', 'O_2']


Episode 1, Score: -2.00
        * Actions: ['Left', 'Left', 'Left']
        * Obs: ['O_2', 'O_2', 'O_2']


Episode 2, Score: -2.00
        * Actions: ['Left', 'Right', 'Right']
        * Obs: ['O_3', 'O_2', 'O_2']


Episode 3, Score: -2.00
        * Actions: ['Left', 'Right', 'Right']
        * Obs: ['O_3', 'O_2', 'O_2']


Episode 4, Score: -2.00
        * Actions: ['Left', 'Right', 'Right']
        * Obs: ['O_3', 'O_2', 'O_2']


Episode 5, Score: -2.00
        * Actions: ['Left', 'Right', 'Right']
        * Obs: ['O_3', 'O_2', 'O_2']


Episode 6, Score: -2.00
        * Actions: ['Left', 'Right', 'Right']
        * Obs: ['O_3', 'O_2', 'O_2']


Episode 7, Score: -2.00
        * Actions: ['Left', 'Left', 'Left']
        * Obs: ['O_2', 'O_2', 'O_2']


Episode 8, Score: -2.00
        * Actions: ['Left', 'Right', 'Right']
        * Obs: ['O_3', 'O_2', 'O_2']


Episode 9, Score: -2.00
        * Actions: ['Left', 'Right', 'Right']
        * Obs: ['O_3', 'O_2', 'O_2']


=================
Avg score: -2.00
```

Q Learning

Since the mapping from observations to real world states is not 1 to 1, the Q Learning agent would not be able to solve the problem. For example, if the agent observes two walls, then it cannot take an action with 100% confidence that it will lead to the goal state and not the edge. The Q table will have the shape (2, 2) where the first axis will represent the observations (two walls and three walls) and the second axis will represent the actions that the agent can take (LEFT, RIGHT). With this information alone it would be impossible to model the entire environment. Something that might happen is that the model will always predict one of the two actions (for example always going left) and winning half of the games.