**Assignment-1 Report**

Name: Satwik Nallapati

Ubmail: [satwikna@buffalo.edu](mailto:satwikna@buffalo.edu)

Ubit number: 50408410

**Abstract:**

In this report we describe two reinforcement learning environments. A stochastic environment and a deterministic environment with 25 states and 5 actions. We also analyze and compare two tabular methods to solve these environments, Q-Learning and SARSA.

**Part 1: Defining RL Environments**

For this project, I created the grid environment in the

following way:

Environment features:

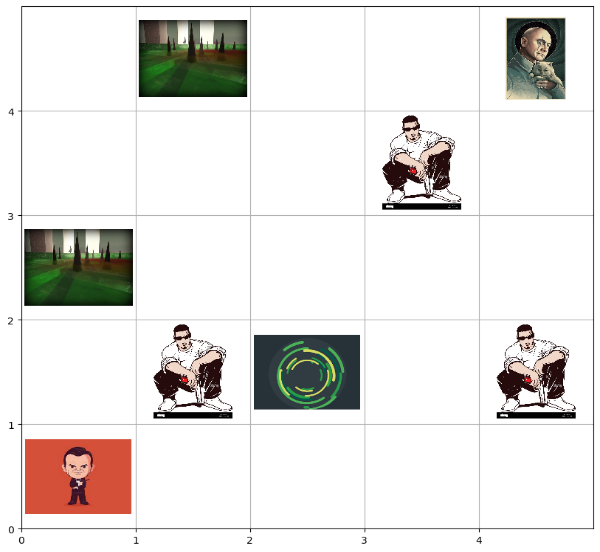
* Number of states :25 (5x5 grid environment)
* Min number of states: 25
* Min number of actions: 4 (up, down, left, right)
* Min number of rewards: 4(+100, +50, -10, -10)

This is the environment that I have built:

Here my agent is James Bond (1,1), and his destination is the villain (final stop at (4,4)) where he will get a reward of 50 points. At each timestep a reward of -1 is received. In between there are 2 kinds of obstacles which is a poisonous pit and a gangster if our agent gets to any of these positions, we will get a reward of -20 points. In between there is a portal if our agent reaches the portal, he gets teleported from [(2,1) to (2,3)] in this case we get a reward of +20.

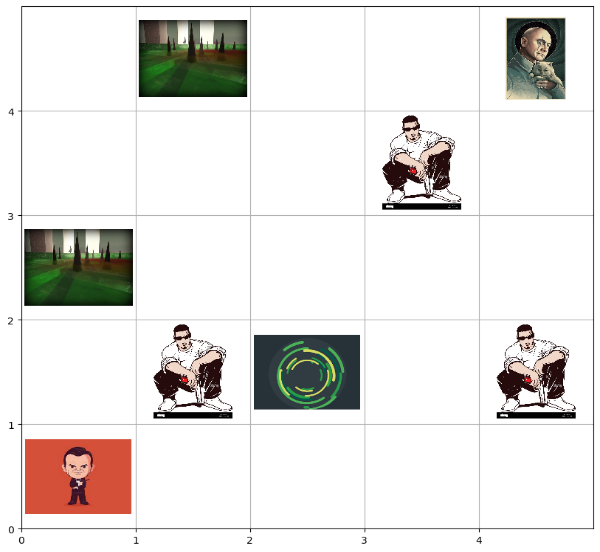
**1.2 Deterministic Environment:**

In the deterministic environment, the agent moves based on the action selected deterministically. If there is no cell in that direction, then it stays on the same cell.



**1.3 Stochastic environment:**

In the stochastic environment, the agent stays in the same cell with probability 0.05, and moves in the direction based on the action selected with probability 0.95.



**Part 2: Applying Tabular Methods**

**2.1 Q-Learning**

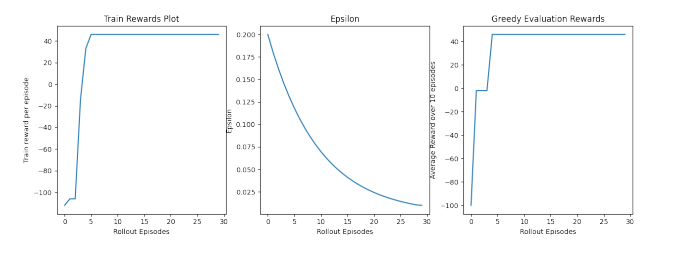
The Q (s, a) update rule used for Q-learning is as follows

Q (s, a) ← Q (s, a) + α · (r + γ · max a′ Q (s ′, a′) – Q (s, a))

This is an off-policy method.

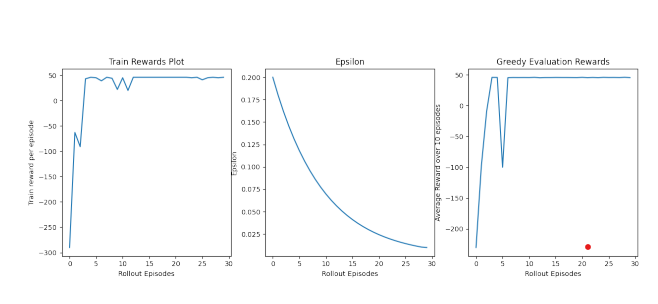
**2.1.1Deterministic environment**

In this section we observe the performance of Q-Learning on deterministic environment. In the figure, the first images show reward per episode during training, the middle plot shows the epsilon and how it decays, and the last plot is greedy evaluation result after each training episode.

****

**2.1.2 Stochastic environment**

In this section we observe the performance of Q-Learning on stochastic environment. In the figure, the first images show reward per episode during training, the middle plot shows the epsilon and how it decays, and the last plot is greedy evaluation result after each training episode.



**2.2 SARSA**

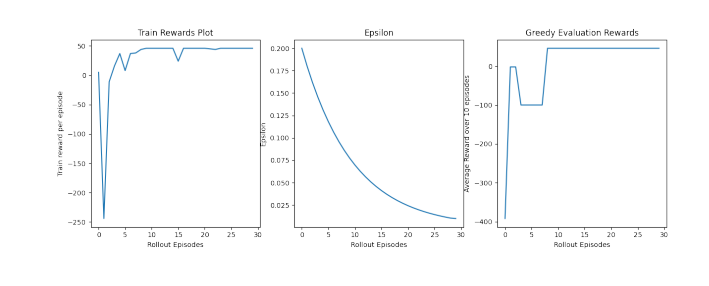
The Q (s, a) update rule used for SARSA is as follows

Q(s, a) ← Q(s, a) + α · (r + γ · Q(s ′ , a′ ) − Q(s, a)).

Here a ′ is sampled from the policy used to train. This is an on-policy method.

**2.2.1 Deterministic environment**

In this section we observe the performance of SARSA on deterministic environment. In the figure, the first images show reward per episode during training, the middle plot shows the epsilon and how it decays, and the last plot is greedy evaluation result after each training episode.



**2.2.2 Stochastic environment**

In this section we observe the performance of SARSA on stochastic environment. In the figure, the first images show reward per episode during training, the middle plot shows the epsilon and how it decays, and the last plot is greedy evaluation result after each training episode.

**2.3 Comparison between Q-Learning and SARSA**

**2.3.1 Deterministic Environment**

Here we compare the evaluation performance of Q-learning and SARSA averaged over 10 episodes after every train episode. We observe that Q-learning converges faster. We can see the results in

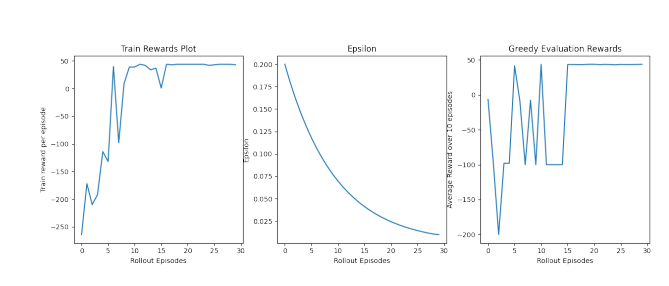


Figure: Stochastic environment initial state visualization



Figure 7: Stochastic environment initial state visualization

**4.Safety of the environment:**

In the environment defined the agent will never go beyond the defined grid world (5x5). The 'clip' function in Python was used to do this.