Reinforcement Learning - Exercise 2

Muhammad Kamal Memon
600442

Question 1: What is the agent and the environment in this setup?

The agent in this environment is the sailor who wants to reach the harbour. The environment consists of the sea, the harbour, the rocks and the narrow passageway in between them. Also the heavy wind conditions and the calmness of the sea are part of the environment.

Task 1

```
# TODO: Compute value function and policy.
value_est, policy = np.zeros((env.w, env.h)), np.zeros((env.w, env.h)),

def action_values(state, V):
   A = []
   for transition in env.transitions[state[0], state[1]]:
       action_value = 0
       for next_state, reward, done, prob in transition:
           action_value += prob * (reward + (gamma * V[next_state] if not done else 0))
       A.append(action_value)
   return A

for lp in range(100):
   env.clear_text()
   for w in range(env.w):
       for h in range(env.h):
           A = action_values((w, h), value_est)
           value_est[w][h] = np.max(A)
           policy[w][h] = np.argmax(A)
   env.draw_values(value_est)
   env.draw_actions(policy)
   env.render()
```

The values get updated from the harbour state all the way down to bottom left. Figure 1 below shows how the values are updated after running the value iteration algorithm:

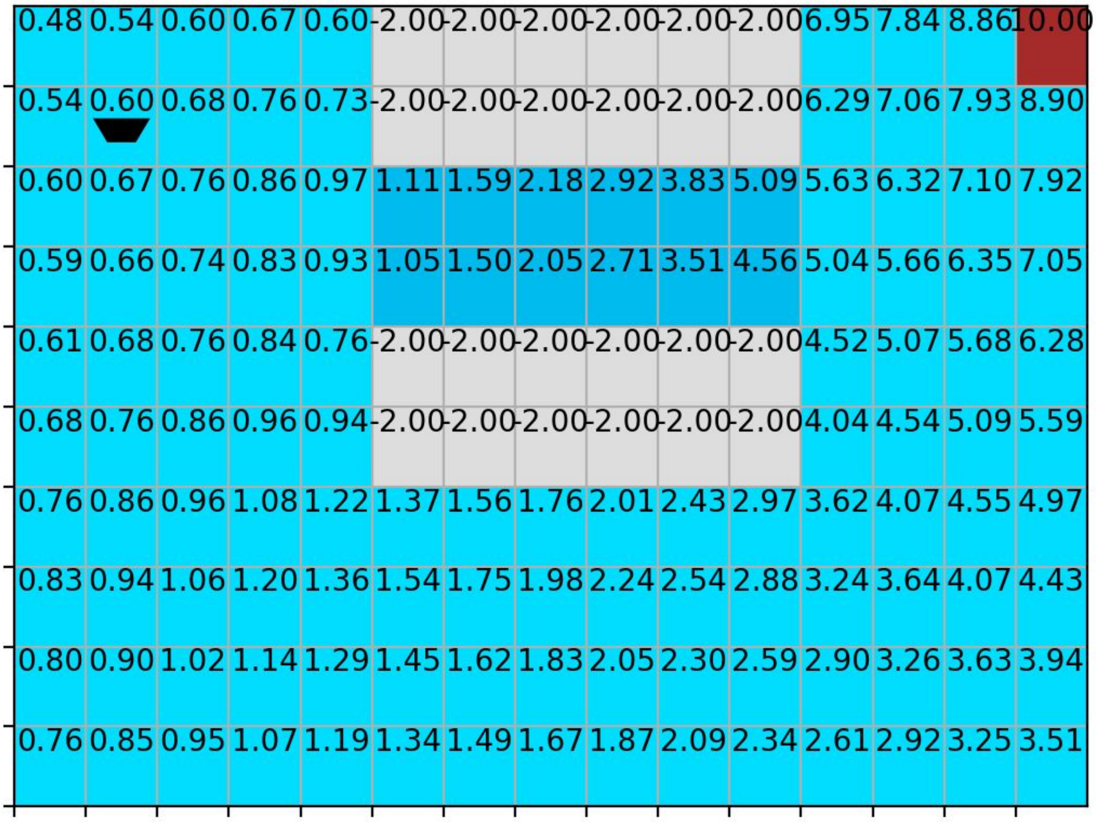

Figure1: Values in the grid state space after Value Iteration

Task 2

```
sleep(3)
done = False
while not done:
   # TODO: Use the computed policy here.
   action = policy[state]
   state, reward, done, _ = env.step(action)
   env.render()
   sleep(0.5)
```

The policy computed above in Task1 is implemented and the sailor manages to reach the goal state most of times the algorithm was run.

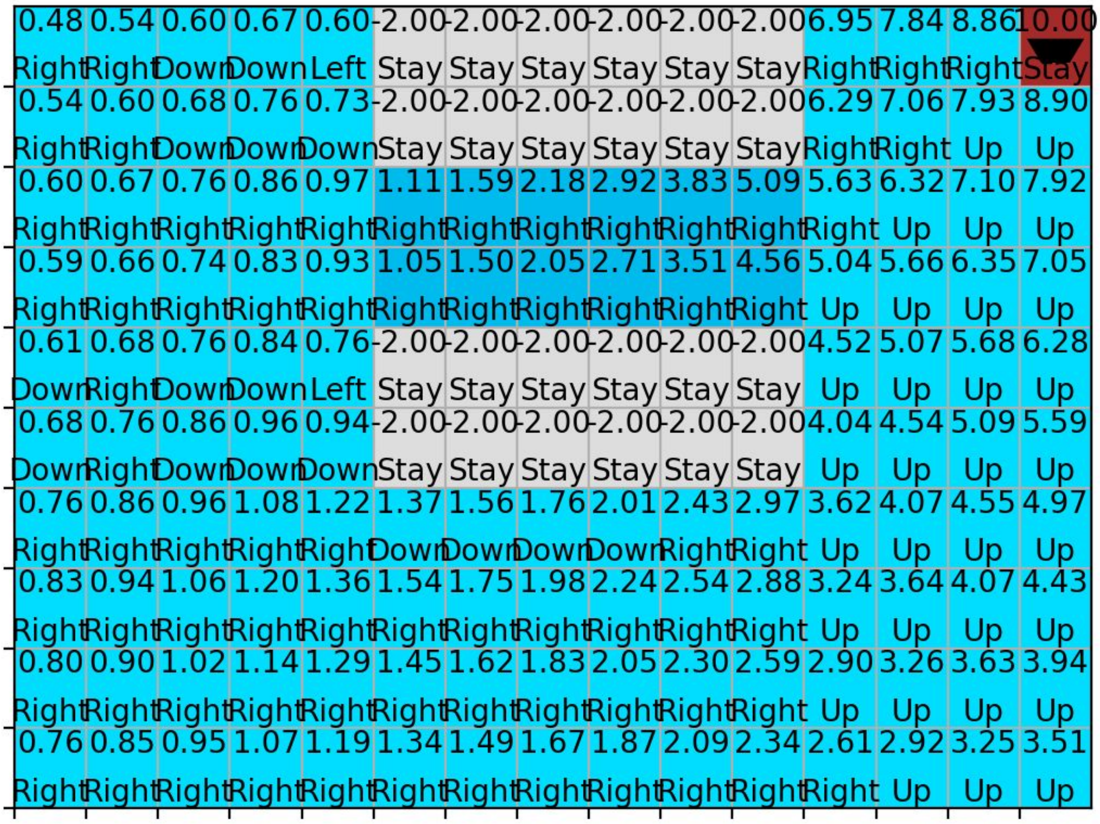

Figure 2: Value function, policy and sailor end

Question 2

The sailor always chooses the path through the narrow passage when the reward to hit the rocks is -2. However when the reward is set to -10 it decides to not take the path from the narrow passage as now the risk of getting negative reward is high and so it goes the other way around to reach the goal.

Question 3

We ran the algorithm for 30 and 15 iterations instead of 100. For 30 iterations the value function and policy still converges and the sailor is able to find its way to the harbour. On the other hand for 15 iterations they don't converge. In more experiments it resulted so that the policy gets converged but the value function take more iterations to do so. The reason being that the value function need the whole state space to take into account while policy gets converged in less iterations. See the Figure below for 15 iterations:

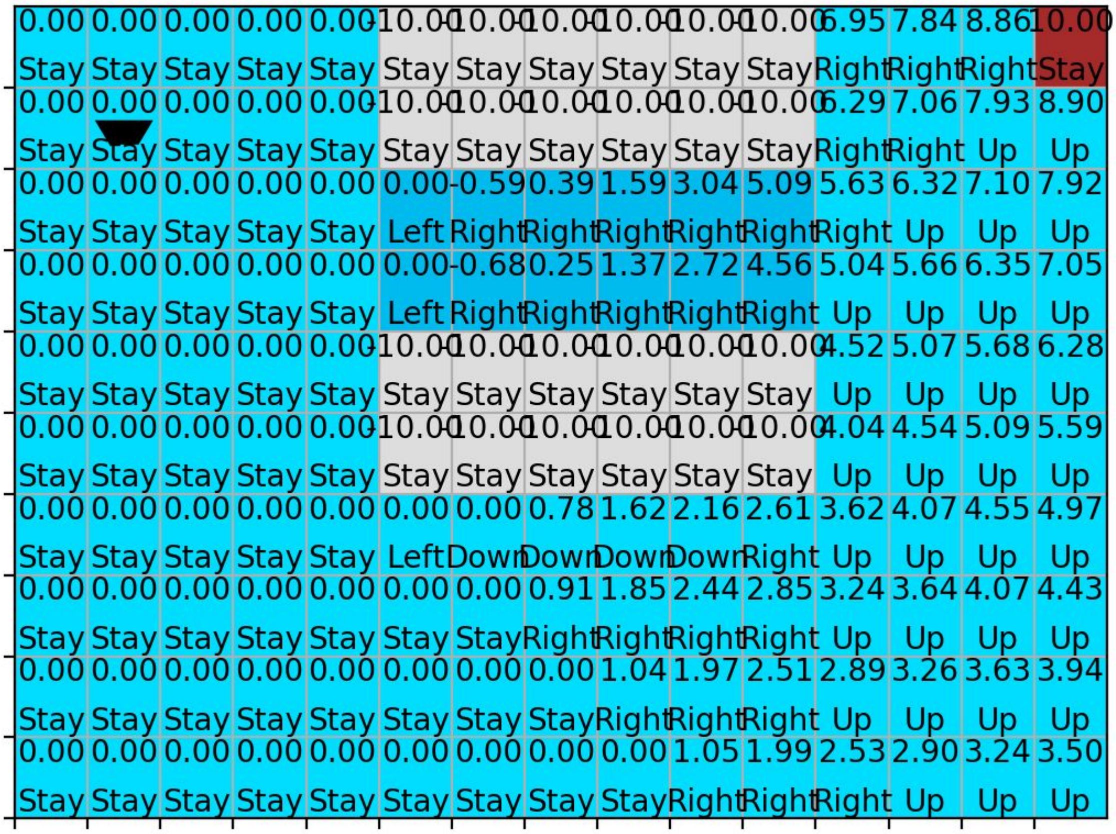

Figure3: Incomplete value function and policy did not converge

Task 3

```
def action_values(state, V):
   A = []
   for transition in env.transitions[state[0], state[1]]:
       action_value = 0
       for next_state, reward, done, prob in transition:
           action_value += prob * (reward + (gamma *
                                   V[next_state] if not done else 0))
       A.append(action_value)
   return A

old_values = value_est.copy()
eps = 0.0001

for lp in range(100):
   env.clear_text()
   for w in range(env.w):
       for h in range(env.h):
           A = action_values((w, h), value_est)
           value_est[w][h] = np.max(A)
           policy[w][h] = np.argmax(A)
   if ((value_est - old_values) < eps).all():
```

```
        print('Stopping')
        break
    else:
        old_values = np.copy(value_est)
    env.draw_values(value_est)
    env.draw_actions(policy)
    env.render()
```

Task 4

```
discounted_reward = 0
i = 0
done = False
while not done:
    # TODO: Use the computed policy here.
    action = policy[state]
    state, reward, done, _ = env.step(action)
    discounted_reward += (gamma ** i) * reward
    i += 1
    env.render()
    sleep(0.5)
print(discounted_reward)
```

Discounted return	Sailor status
1.0941	Managed to reach the harbour
-0.4117	Hit the rocks
1.8530	Managed to reach the harbour
-0.9565	Hit the rocks
1.5009	Managed to reach the harbour
1.6677	Managed to reach the harbour

Question 4

For any policy π and any state s, the following holds:

$$v_\pi(s) \doteq \mathbb{E}_\pi[G_t \mid S_t = s]$$

That is, the value function is expectation (following a particular policy) of the discounted return G_t for any state s [Chapter 3 Sutton & Barto]

Question 5

The value iteration approach implemented here wouldn't fit too well to a real world scenario. Mainly because here we have finite state space with a deterministic set of actions. Moreover, in a real world setting computing the value function the way we did can also get very computationally expensive as we are maximizing over action values. Also the model of the problem would not be this straight forward.

References:

The exercise involved discussion with following students:

- Gadidjah Ogmundsdottir
- Hector Laria Matecon