ÇUKUROVA UNIVERSITY

ENGINEERING FACULTY

DEPARTMENT OF COMPUTER ENGINEERING

GRADUATION THESIS

IMPLEMENTATION OF REINFORCEMENT LEARNING ALGORITHMS FOR SOLVING MAZE PROBLEM AND COMPARING THE RESULTS

By

Zirveda AYTİMUR

Advisor

Dr. Barış ATA

June 2021

ADANA

ABSTRACT

Reinforcement learning enables autonomous robots to learn in various fields. This project is based on an autonomous robot finding the target point in the designed maze. It can be confusing which algorithm to choose in such complex problems. This thesis includes the solution of this maze using Random Actions algorithm, reinforcement learning algorithms Q-Learning and SARSA and the comparison of the results of the algorithms. The maze prepared with a matrix was visualized with PyGame and the algorithms were coded using Python. Algorithm comparisons were made with how many iterations the robot went to the target point, how many rewards it received, and the mean squared error in each episodes.

List of Figures

List of Tables

List of Symbols

α	Learning rate
γ	Discount factor
a	Action of an agent
s or S_t	State of an environment
s' or S_{t+1}	Next state
V	Value function
R	Reward of the selected action and state
P	Policy function
ε	Epsilon
π	Policy of an agent

CONTENTS

1. INTRODUCTION

Reinforcement learning; it is a sub-branch of machine learning that has become very popular in recent years. Today; reinforcement learning plays an important role in many areas such as self-moving robots [1], unmanned aerial and ground vehicles [2] [3], and the game world with self-learning virtual players [4].

Maze solving problems are used in various fields and technologies. The use of sensors and learning algorithms in various robotic problems such as an autonomous vehicle finding the shortest path without hitting obstacles [5], and a robot vacuum cleaner mapping the house [6] can be given as examples. It is used in path finding in strategy games [7] or can be used with image processing to solve a problem [8].

Maze learning problems have been conducted in early behaviorist psychology experiments. These have led to a theory of learning known as operant conditioning [9]. With the development of computer science, the solution of such problems was made possible by reinforcement learning. In reinforcement learning, it uses certain conditions for the learning of the program, and the rewards or penalties received from this return to the program as a feedback.

Q-Learning and SARSA algorithms were used for the learning of the maze, and besides these algorithms, it was observed what results would be obtained if the program moved randomly to test the learning. Algorithm comparisons were made with how many iterations the robot went to the target point, how many rewards it received, and the mean squared error in each episodes.

Two different mazes were designed for the implementation of the algorithms. Both mazes were designed in rooms based on the mapping system of robot vacuum cleaners. The first maze has three rooms and the second maze has four rooms. In both test areas, the starting point for the robot to start, the target point of the robot was determined, and the boundary zones with high penalties were drawn to represent the surroundings of the house, the walls defining the rooms and the obstacles in the rooms were defined with a lower penalty score. The prize is awarded only when the robot arrives at the designated target point. The algorithm was created with matrices and visualized later.

The number of steps in each episode, which is one of the algorithms comparison criteria, enables to find out at which step the robot’s learning begins, so which algorithm learned faster. The mean squared error rate indicates how much has been approached or moved away from the number of steps determined as the shortest distance in each episode. So we can see the approximation to the optimal point. The total reward points received in each section indicate whether the rate of hitting obstacles has decreased or not. With these three criteria, it is aimed to compare the difference between reinforcement learning algorithms from random acting and for which situations the algorithms may be advantageous.

This thesis is structured as follows: In the second chapter, there is an explanation of the reinforcement learning and algorithms used. In chapter three, the mazes used are described, and the results is listed. In chapter four, the results obtained after applying the algorithms to the mazes are shown. In chapter five, comparisons were made based on the results of the algorithm. Finally, there is the conclusion part.

2. REINFORCEMENT LEARNING

In reinforcement learning, train decision makers to take actions to maximize rewards in an uncertain environment. Software decision makers can be called as programs or agents. Actions are referring to the decisions that the agent has taken. The output of the program is a set of actions, rather than a set of predictions. The algorithm that determines these actions is called the policy. Those actions must be optimized to earn rewards and avoid punishments. Those rewards and punishments are externally imposed by the environment. The environment is complex, so the reward punishment for actions is usually not known in advance. The decision maker needs to be trained to explore that uncertain environment combining caution and courage. You can see this explanation exactly in the Figure 1.

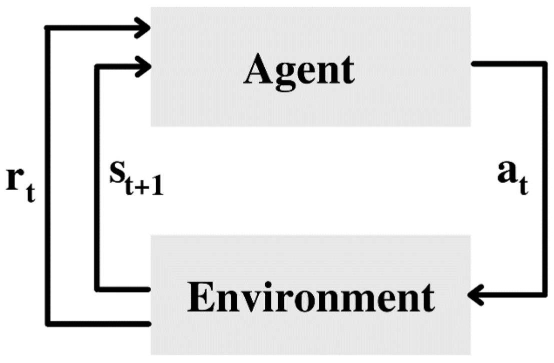


Figure 1 The reinforcement learning framework

At each time step, the agent observes the state of the environment $s_t \in S$ where S is a set of states. The agent chooses an action $a_t \in A$, where A is the set of actions available to the agent. The agent receives a reward signal $r_t \in R$ which is a measure for the consequence of its actions. The environment changes into a new state s_{t+1}. The agent's goal is to maximize the reward signal over time [10] [11].

2.1. Bellman Equation

The Bellman Equation was introduced in 1953 by the mathematician Richard Ernest Bellman and is therefore, called the Bellman Equation. It is associated with dynamic programming and is used to calculate the values of a decision problem at a particular point by including the values of previous states.

It is a way of computing dynamic programming or value functions in the environment that leads to modern reinforcement learning. The Bellman equation is given in The Bellman Equation formula.

$$V(s) = max_a(R(a,s) + \gamma V(s'))$$

The Bellman Equation formula

V(s) means value of the current state. R (a, s) or R (s, a) is reward to be received when action is taken. V(s') presents the value of the next state. The Bellman equation is defined as deterministic; we use the Markov Decision Process to make this equation stochastic [12] [11].

2.2. Markov Decision Process

A reinforcement learning problem that satisfies the Markov property is called a Markov Decision Process, or MDP. According to the MDP, the future is independent of the past, given the present. At each time step; environment is in some state (S_t). Decision maker can choose an action (a); this moves environment to new state (S_{t+1}). Decision maker receives reward (R_a (S_t, S_{t+1})). 'S_{t+1}' depends only 'a' and 'S_t'. Therefore, the reward is in the form of R_a (S_t, S_{t+1}).

All of the information about environment is encapsulated in present state; so how we got here does not matter. MDP greatly simplifies the exploration of environment. It allows use of dynamic programming. Best action, reward for state can be memoized(cached) [11].

To create a stochastic equation from The Bellman Equation we need to include the probabilities for the future states so the formula is given in Markov Decision Process formula.

$$V(s) = max_a(R(a,s) + \gamma \sum_{s'} P(s,a,s')V(s'))$$

Markov Decision Process formula

2.3. Policy Search

There are 3 main elements in reinforcement learning problems. Firstly, a positive result is given when we do the right actions which means reward. Secondly; there is software that competes for the award. We call it agent. Finally, an algorithm is also required to select the actions that will result in the reward. This is called policy. Policy determines our actions.

The working system of reinforcement learning is as follows; first, we discover the environment and receive rewards or punishments based on our actions. In line with these awards and penalties, we decide which actions we should take according to the situations we encounter, and this becomes our policy. While exploring, we choose actions in accordance with the current policy, so while constantly exploring; on the one hand, politics take shape. After the discovery process is over, the agent acts according to the specified policy.

The decision maker has to find the most suitable policy. The Policy function (P) takes the current state (S) and gives the action (a).

$$a = P(S)$$

Policy function

Policy P should maximize cumulative rewards. A reward or punishment is given for every change of situation but past experiences significantly contribute to our learning. For this reason, we use accumulated rewards, i.e. cumulative rewards [11] [12].

2.3.1. Cumulative Rewards

To calculate the cumulative reward between now and eternity, we need to add up all the expected rewards. However, in future rewards, we should reduce with a discount factor gamma because we are not entirely sure of future rewards. Gamma (γ) should be given between 0 and 1. 0 future rewards are insignificant; 1 means that future prizes are just as important as current prizes.

The policy we found should have been to increase our total rewards, so the function is as shown in Cumulative Reward Function;

$$\sum_{t=0}^{\infty} \gamma^t R_{P(S)}(S_t, S_{t+1})$$

Cumulative Reward Function

2.3.2. Policy Search Algorithms

- **Brute Force Methods**

It evaluates every possible policy for every possible states. It chooses the best policy with the best expected rewards [12].

- **Policy Gradient Methods**

It explores the state space to find the best policy. Usually uses a neural network to learn policy. It does not need to model the environment with any method such as Markov Decision Process [12].

- **Value Function Methods**

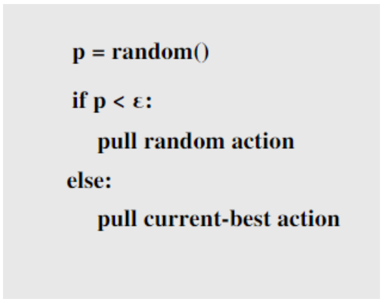
With the Markov Decision Process, the environment has to be clearly modeled. Algorithms belonging to this structure are Q-Learning, SARSA and Monte Carlo [12].

2.4. Dynamic Programming

Dynamic Programming uses caching or memoization to help reduce computational intensity. It is a standard technique in reinforcement learning. A method of solving a complex problem by decomposing it into simpler sub-problems, solving those simpler sub-problems just once; and caching the results. The Q-Table used in Q-Learning and SARSA algorithms is an example to Dynamic Programming [11].

2.5. Exploitation or Exploration with Epsilon-Greedy Algorithm

Both concepts are used when choosing the action to be taken. Whether the action will be selected by exploration or exploitation is determined by epsilon (ε). The epsilon value determines the exploration value. A random value is chosen, if this value is less than or equal to epsilon, a random action is chosen with exploration; if it is greater, the action is performed according to the maximum value in the Q-Table already learned by exploitation. The algorithm is given in Figure 6.



```
p = random()
if $p < \varepsilon$:
    pull random action
else:
    pull current-best action
```

Figure 2 Epsilon-Greedy algorithm

An important way to implement algorithms is to first select actions with exploration to explore the environment as much as possible, and then select actions with exploitation for these discovered states. For this, the epsilon value is gradually reduced after each action, thus reducing the rate of initial discoveries; in order not to reduce the discovery rate completely, the minimum epsilon rate is defined and the reduction process is stopped as soon as the epsilon value is equal to or less than this value. Thus, no matter how much the agent discovers the environment and the Q Table is shaped, the possibility of discovery is not completely finished, even if there is a small possibility [11].

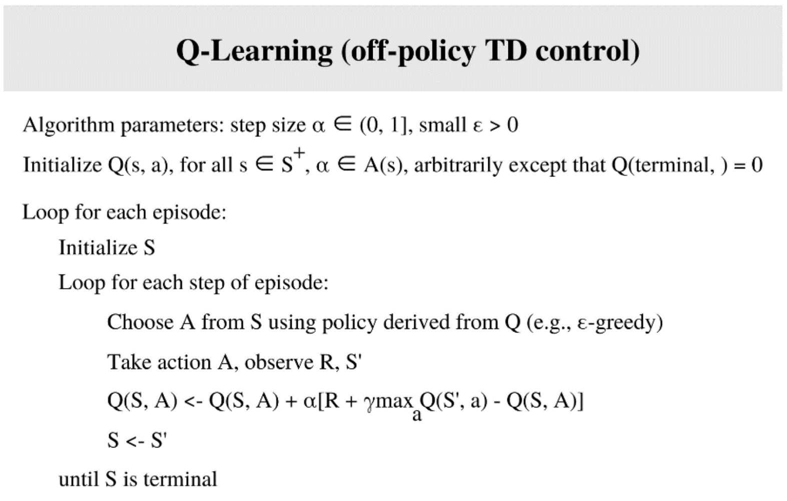
2.6. Temporal Difference

Temporal difference is an agent that learns from an environment through episodes without prior knowledge of the environment. This means that the temporal difference requires an independent or unsupervised learning approach from the model. Can think that it learns through trial and error [10] [11].

2.6.1. Q-Learning

Q learning is an RL algorithm that is used for temporal difference and is off-policy; that is, the learned action-value function converges to the best action-state function regardless of a policy. The policy is still active, but its sole purpose is to make sure that all state-action pairs continue to be updated.

Q (s, a) learns the value function, which shows how good it would be to choose the action 'a' in the 's' state. You can see the pseudocode of the Q-Learning algorithm in Figure 7.



Q-Learning (off-policy TD control)

```
Algorithm parameters: step size $\alpha \in (0, 1]$, small $\varepsilon > 0$
Initialize Q(s, a), for all $s \in S^+$, $\alpha \in A(s)$, arbitrarily except that Q(terminal, ) = 0
Loop for each episode:
    Initialize S
    Loop for each step of episode:
        Choose A from S using policy derived from Q (e.g., $\varepsilon$-greedy)
        Take action A, observe R, S'
        $Q(S, A) <- Q(S, A) + \alpha[R + \gamma \max_a Q(S', a) - Q(S, A)]$
        S <- S'
    until S is terminal
```

Figure 3 Q-Learning pseudocode

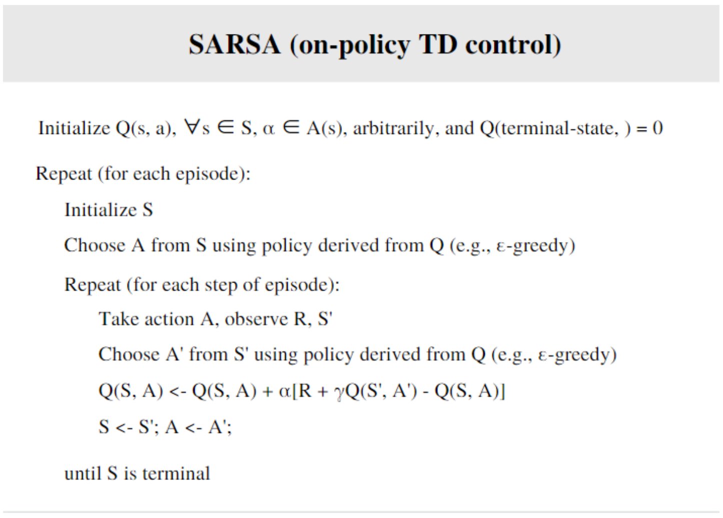
According to these steps, the formula for updating the Q-Table for Q-Learning can be seen in Q-Learning formula.

$$Q_{t+1}(s,a) = (1-\alpha)Q_t(s,a) + \alpha(R(s,a) + \gamma max_{a'}(Q(s',a')))$$

Q-Learning formula

2.6.2. SARSA

SARSA stands for State-Action-Reward-State-Action, which is a temporal difference learning method based on on-policy; that is, it chooses the action for each state while learning to use a particular policy. The purpose of SARSA is to calculate Q ‘(s, a) (next state and action) for selected current policy π and all (s-a) pairs. The main difference between Q-Learning and SARSA is that unlike Q-Learning, the maximum reward is not required for the next state to update the Q value in the table. At SARSA, new action and reward are selected using the same policy that determines the original action [12]. In Figure 9 you can see the pseudocode of the SARSA algorithm.



SARSA (on-policy TD control)

```
Initialize Q(s, a), $\forall s \in S, \alpha \in A(s)$, arbitrarily, and Q(terminal-state, ) = 0
Repeat (for each episode):
    Initialize S
    Choose A from S using policy derived from Q (e.g., $\varepsilon$-greedy)
    Repeat (for each step of episode):
        Take action A, observe R, S'
        Choose A' from S' using policy derived from Q (e.g., $\varepsilon$-greedy)
        $Q(S, A) \leftarrow Q(S, A) + \alpha[R + \gamma Q(S', A') - Q(S, A)]$
        S <- S'; A <- A';
    until S is terminal
```

Figure 4 SARSA pseudocode

The formula for the Q value to be updated in SARSA can be seen in SARSA formula.

$$Q_t(s,a) = (1 - \alpha)Q_t(s,a) + \alpha(R_{t+1}(s,a) + \gamma Q_{t+1}(s,a)$$

SARSA formula

3. MAZE

In the maze, the goal is to let the agent pass through the rooms and reach the target point. Two mazes were designed, one with three room and two goal points, the other with four rooms and one goal point. While creating the maze, rewards and penalties were specified on the matrix. States around the maze were given the highest penalty with '-4' so the agent learned to stay in the maze as much as possible. The problem was complicated by placing various obstacles inside the rooms; these obstacles were scored as '-1' penalty points. Rewards with '+10' points if the agent reaches the target point. When the agent hits an obstacle agent moves the starting state; if it reaches the target point, that episode ends and the new episode starts from the specified starting state.

3.1. Implementing Q-Learning Algorithm

While applying the Q-learning algorithm, it is checked whether the application is working at that moment for each episode and the steps are started to be implemented. Visualization is the first function applied, then it is followed by checking whether the agent has reached the target point. If the agent has reached one of the target points, an episode is finished and the necessary parameters for the graphics are added, the agent position is synchronized to the starting position. On the other hand, if the agent is not at the target point, a choice is made among the actions that the agent can take. Epsilon greedy method is used here. According to the action taken, the reward is received and the next state is determined. If the position of the agent is not the penalty sections shown as obstacles, the function is calculated only for Q-Learning function; if it is an obstacle, the position is updated as the starting position after the Q-learning function is calculated. The epsilon value is reduced by a determined rate. After the Q value is determined and the status is updated, the reward, number of iterations and mean squared error values in each episode are finally sent to the main data visualization function.

3.2. Implementing SARSA Algorithm

While applying the SARSA algorithm, as in Q-Learning, it is first checked whether the target point has been reached in the current state. Then, visualization is applied. If it has been reached, everything is reset and the data required for visualization is added to the lists. If it has not been reached, unlike Q-Learning, first the action to be taken according to the current state is selected and the agent is moved. After the reward is increased according to the current position, the new state becomes the current position and the new action is decided according to the current position. If the new state is not obstacles, the Q value for SARSA is calculated, the agent's state is synchronized to the new state, and the action is synchronized to the new action. If the new state is an obstacle, after Q value is calculated for SARSA, the position is reset, the current state is equalized to the starting position and the action is equal to the action to be taken in the initial state. The epsilon value is reduced at a determined rate. After the Q value is determined and the status is updated, the reward, number of iterations and mean squared error values in each episode are finally sent to the main data visualization function.

3.3. Implementing Random Actions

In the algorithm created for the random movement of the agent, it is checked whether the target position has been reached according to the current position after the visualization. If it has, everything is reset and the necessary parameters are added to lists for data visualization. Since there is no learning here, a random action is chosen according to the current position. If the new state is an obstacle, the position is set as the starting position. In order to get the results, the results are obtained whether the agent has reached the target point or when the number of iterations is 20000.

3.4. The Simpler Maze

The maze is 8 x 8 in size. There are three rooms and two exit points in this maze. There is a one-part passage between the first room and the second room, and a two-part passage between the second room and the third room. While there are no obstacles in the first and second rooms, there are two obstacles in the third room. The targeted exit points are in the third room. The starting point of the agent is given as the [1, 1] point. The shortest number of iterations from the starting point to the target point is 11. The view of the maze is given in Figure 11.

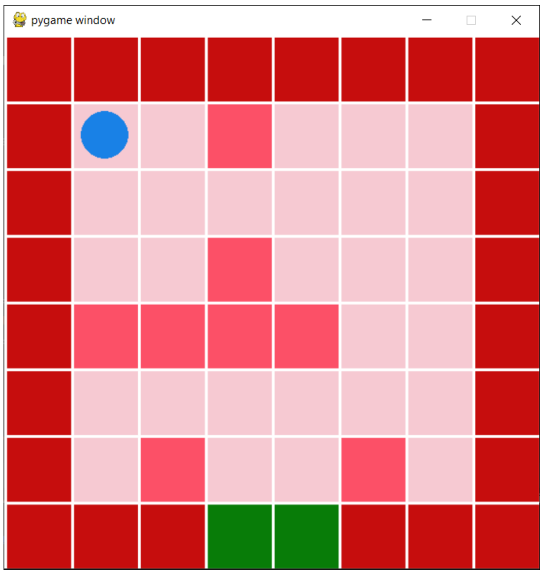
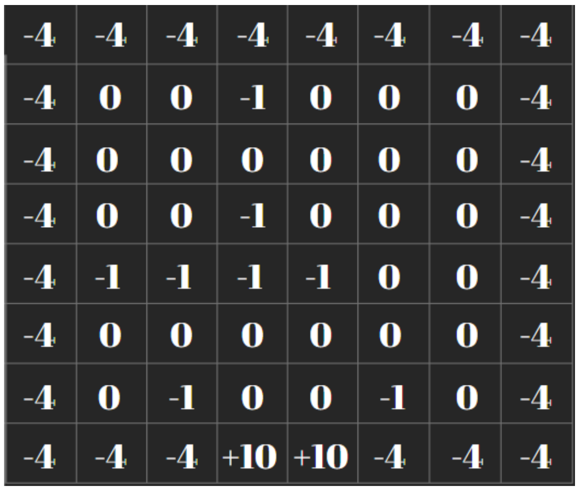


Figure 5 The Simpler Maze

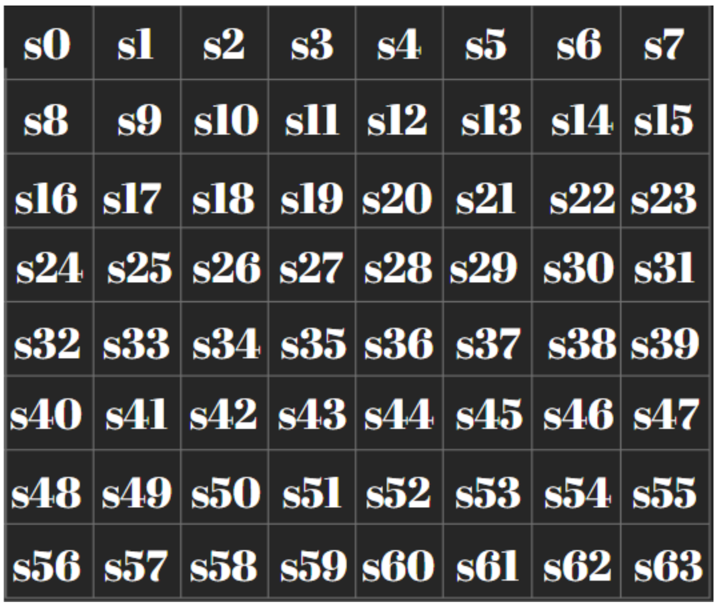
Dark red parts of the maze are scored with the highest penalty, '-4'. Dark pink areas are penalized with '-1', and green areas are rewarded with '+10'. In the light pink areas, no reward or penalty points are awarded. You can see the rewarding and punishment points in Figure 12.



-4	-4	-4	-4	-4	-4	-4	-4
-4	0	0	-1	0	0	0	-4
-4	0	0	0	0	0	0	-4
-4	0	0	-1	0	0	0	-4
-4	-1	-1	-1	-1	0	0	-4
-4	0	0	0	0	0	0	-4
-4	0	-1	0	0	-1	0	-4
-4	-4	-4	+10	+10	-4	-4	-4

Figure 6 The Simpler Maze rewards

In order to apply reinforcement learning algorithms, it is necessary to convert each point defined as matrix into states. For this reason, since this metric is an 8 x 8 matrix, there are 64 states. In Figure 13 you can see which state is at which point.



s0	s1	s2	s3	s4	s5	s6	s7
s8	s9	s10	s11	s12	s13	s14	s15
s16	s17	s18	s19	s20	s21	s22	s23
s24	s25	s26	s27	s28	s29	s30	s31
s32	s33	s34	s35	s36	s37	s38	s39
s40	s41	s42	s43	s44	s45	s46	s47
s48	s49	s50	s51	s52	s53	s54	s55
s56	s57	s58	s59	s60	s61	s62	s63

Figure 7 The Simpler Maze states

3.5. The Complex Maze

The maze is 10 x 10 in size. This maze is designed more complex than the other maze. There are four rooms and a target point in this maze, and every room has various obstacles except for the first room. The complexity was tried to be increased with the obstacles around the target point. The target point is located in the fourth room and the starting point of the agent is given as [1, 1]. The shortest number of iterations from the starting point to the target point is 20. The view of the maze is given in Figure 14.

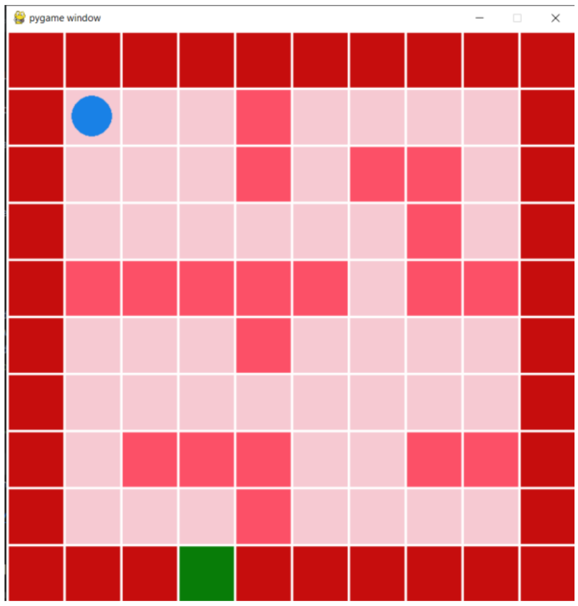
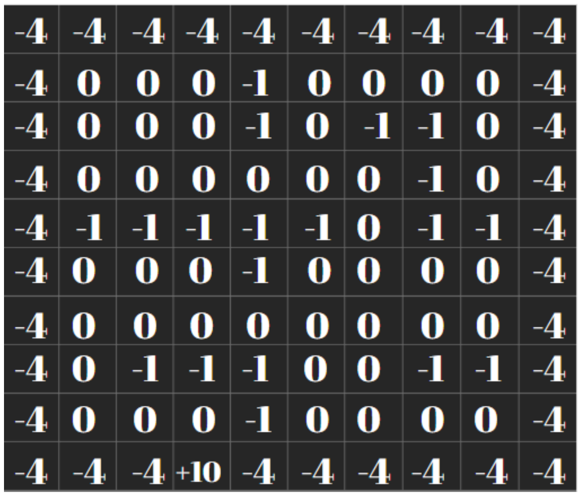


Figure 8 The Complex Maze

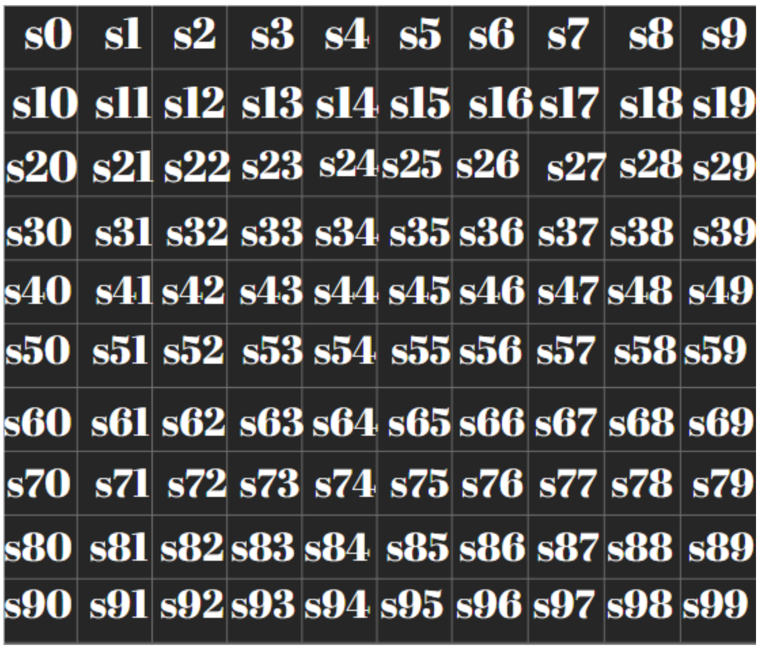
Dark red parts of the maze are scored with the highest penalty, '-4'. Dark pink areas are penalized with '-1', and green areas are rewarded with '+10'. In the light pink areas, no reward or penalty points are awarded. You can see the rewarding and punishment points in Figure 15.



-4	-4	-4	-4	-4	-4	-4	-4	-4	-4
-4	0	0	0	-1	0	0	0	0	-4
-4	0	0	0	-1	0	-1	-1	0	-4
-4	0	0	0	0	0	0	-1	0	-4
-4	-1	-1	-1	-1	-1	0	-1	-1	-4
-4	0	0	0	-1	0	0	0	0	-4
-4	0	0	0	0	0	0	0	0	-4
-4	0	-1	-1	-1	0	0	-1	-1	-4
-4	0	0	0	-1	0	0	0	0	-4
-4	-4	-4	+10	-4	-4	-4	-4	-4	-4

Figure 9 The Complex Maze rewards

To apply reinforcement learning algorithms, it is necessary to convert each point defined as matrix into states. For this reason, since this metric is a 10 x 10 matrix, there are 100 states. In Figure 16 you can see which state is at which point.



s0	s1	s2	s3	s4	s5	s6	s7	s8	s9
s10	s11	s12	s13	s14	s15	s16	s17	s18	s19
s20	s21	s22	s23	s24	s25	s26	s27	s28	s29
s30	s31	s32	s33	s34	s35	s36	s37	s38	s39
s40	s41	s42	s43	s44	s45	s46	s47	s48	s49
s50	s51	s52	s53	s54	s55	s56	s57	s58	s59
s60	s61	s62	s63	s64	s65	s66	s67	s68	s69
s70	s71	s72	s73	s74	s75	s76	s77	s78	s79
s80	s81	s82	s83	s84	s85	s86	s87	s88	s89
s90	s91	s92	s93	s94	s95	s96	s97	s98	s99

Figure 10 The Complex Maze states

4. ALGORITHM RESULTS

Three algorithms are applied to both mazes and the results will be compared. These comparisons are Q-Learning, SARSA and random actions algorithms, respectively. Then the results of the two mazes will be looked at and the comparison of the two mazes will be made.

4.1. Q-Learning Results

In Q-Learning there is a Q-Table with the Q values of states and actions, which is initially filled with 0s. In the Q-learning algorithm, the Q value is calculated for each state in the maze and for every action that can be taken in that state, and the Q table is updated. According to this value, the policy to be made by the agent is determined. The agent decides which action to take while in that state by looking at the maximum value from the Q-table. Different Q tables are formed in both mazes. Rows in these tables show states and columns show actions.

4.1.1. Result for The Simpler Maze

This maze has 64 states (Figure 13) and 4 actions. These actions are to go up, down, left and right respectively. Table 1 is the updated table as a result of the Q-Learning algorithm. After applying the Q-Learning algorithm to this maze, the way the agent learns to go to the target point is as follows in 13 iterations. This path can be seen by looking at the Q table created for which action will be selected in each case.

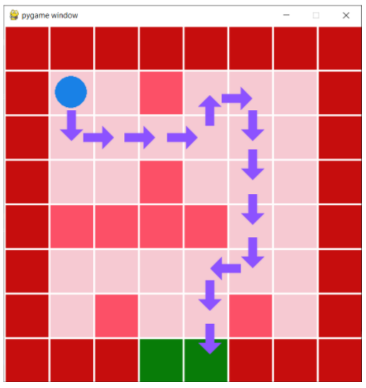


Figure 11 Learning path for The Simpler Maze with Q-Learning

4.1.2. Result for The Complex Maze

This maze has 100 states (Figure 16) and 4 actions. These actions are to go up, down, left and right respectively. Table 2 is the updated table as a result of the Q-Learning algorithm. After applying the Q-Learning algorithm to this maze, the way the agent learns to go to the target point is as follows in 20 iterations. This path can be seen by looking at the Q table created for which action will be selected in each case.

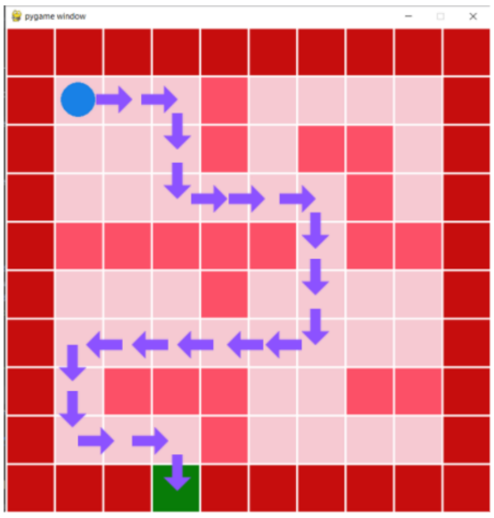


Figure 12 Learning path for The Complex Maze with Q-Learning

4.2. SARSA Results

The SARSA algorithm also has a Q table as in Q-learning. As the agent moves, the Q value is calculated according to the current state and the selected action and the table is updated. Different Q tables are formed in both mazes. Rows represent states, columns represent actions.

4.2.1. Result for The Simpler Maze

As can be seen in Figure 13, this maze consists of 64 states and there are 4 actions that can be taken in each state. In Table 3, after the SARSA algorithm is applied and the agent finishes learning, it can be see the final state of the Q value table. In Figure 19 is the shortest path found by the SARSA algorithm. The agent learns to go to the target point is as follows in 11 iterations.

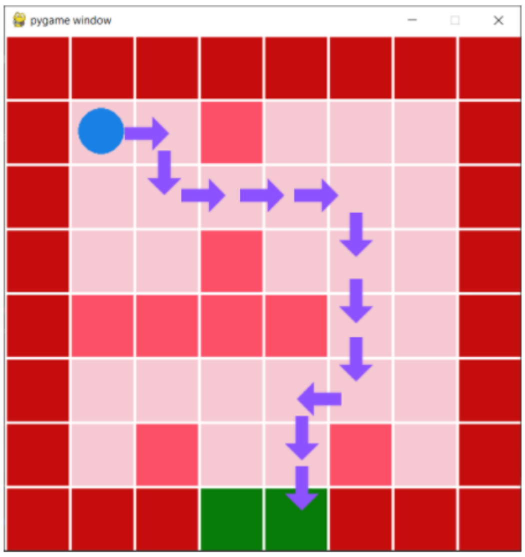


Figure 13 Learning path for The Simpler Maze with SARSA

4.2.2. Result for The Complex Maze

As can be seen in Figure 16, this maze consists of 100 states and there are 4 actions that can be taken in each state. In Table 4, after the SARSA algorithm is applied and the agent finishes learning, it can be see the final state of the Q value table. In Figure 20 is the shortest path found by the SARSA algorithm. The agent learns to go to the target point is as follows in 20 iterations.

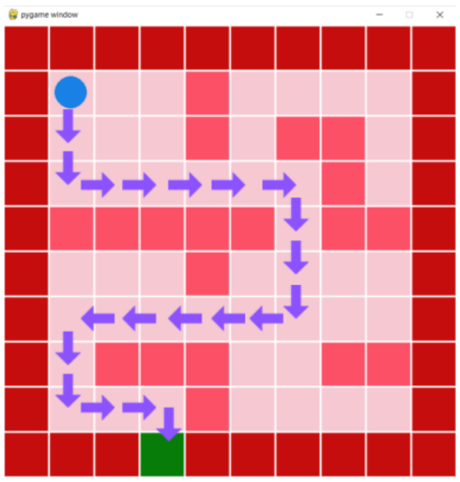


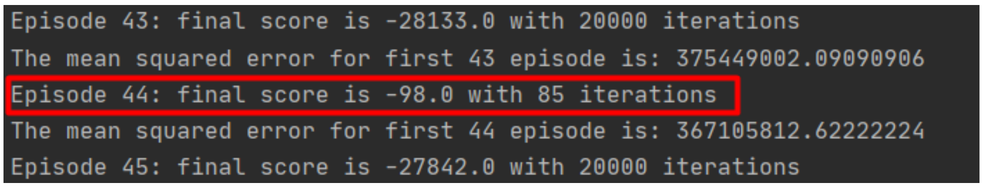
Figure 14 Learning path for The Complex Maze with SARSA

4.3. Random Actions Result

Q Table does not occur in Random Actions where there is no learning. In this algorithm, since the agent acts completely randomly, it is not expected to find any short paths. If the agent does not reach any result, after 20000 iterations the movements are stopped and the next episode starts.

4.3.1. Result for The Simple Maze

In The Simple Maze, the agent reached the target points in random moves 14 times in 100 episodes. The agent is not expected to find the short path in random moves; the agent went to the target points in 85 iterations in the 44th episode as the shortest.



```
Episode 43: final score is -28133.0 with 20000 iterations
The mean squared error for first 43 episode is: 375449002.09090906
Episode 44: final score is -98.0 with 85 iterations
The mean squared error for first 44 episode is: 367105812.62222224
Episode 45: final score is -27842.0 with 20000 iterations
```

Figure 15 The shortest iteration number agent went with random actions

4.3.2. Result for The Complex Maze

In The Complex Maze, the agent could not reach the target point in limiting 20000 iterations by acting randomly.

5. COMPARISONS

Two algorithms, three algorithms were applied and the results were seen. Three different comparisons are made in this section. First, reinforcement learning algorithms for each maze will be compared, then the performances of these algorithms will be compared with random actions. For the performance of the algorithms, three tables were created; first table shows the cumulative reward for one episode, second one shows the number of iterations in one episode, and third table shows the mean squared error changes. Finally, there are two maze comparisons.

5.1. Comparisons for The Simpler Maze

For The Simpler Maze, three algorithms have been implemented, these are Q-Learning, SARSA and Random Actions; Q-Learning and SARSA are reinforcement learning algorithms. Random Actions is based on completely random movements. The shortest path in this maze is 11 iterations. Possible shortcuts in Figure 16 can be seen with different colored arrow icons.

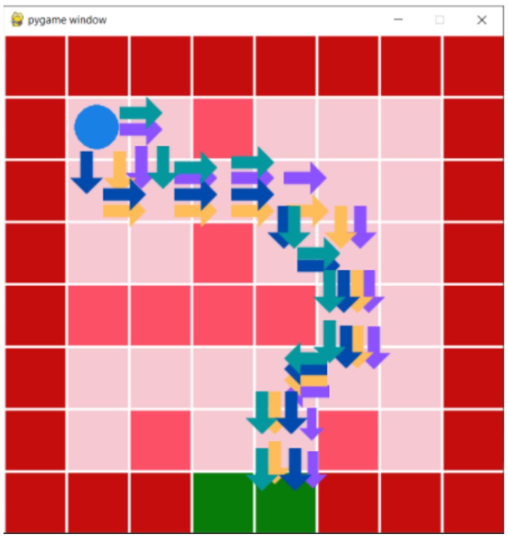


Figure 16 The Simpler Maze short paths

5.1.1. Reinforcement Learning Algorithms

When creating the first table, it was shown how the rewards won by the agent change in each episode according to the rewards shown in Figure 6. In the second table, it was observed how many iterations it took to reach the target point in each section, and the mean squared error rate of the number of iterations made by the agent to the targeted points according to the shortest path was shown in the last table.

As seen in Figure 11, Q-Learning reaches the target point in 13 iterations. This situation does not comply with any path described in Figure 16 and reached the target point with 2 more iterations than 11 iterations. On the other hand, as seen in Figure 13, the agent reached the target point in 11 iterations with the SARSA algorithm. This is one of the paths shown in Figure 16.

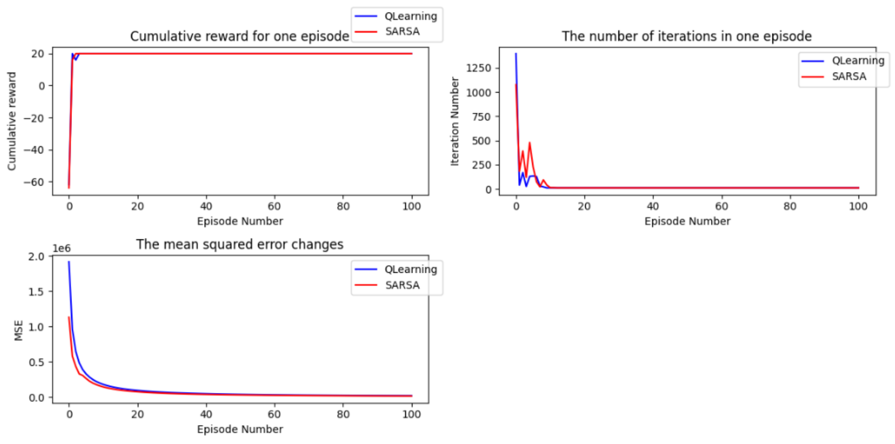


Figure 17 The Simpler Maze first results

As can be seen in Figure 17, there is no particular difference in the reward received in each episode. In the other two tables, there is no difference in the points where the learning ends. It is seen that training seems to be over about in 25th episode and there is a policy to be implemented. Considering the mean squared error and the number of iterations in the episodes, it is seen that the main difference is at the beginning of the learning. At the beginning of the training, it is seen that the SARSA algorithm learns the policy that should be implemented more quickly. This is because, as explained in Chapter 2.6.2., SARSA is an on-policy learning, so when learning to use a particular policy, choosing the action for each episode, it is sooner to find the policy that needs to be applied.

SARSA, one of the reinforcement learning algorithms, gave a better result for this maze.

5.1.2. Random Actions and Reinforcement Algorithms

Random Actions did not show any learning. Therefore, it reached the target in the shortest 85 iterations, as can be seen in Chapter 4.3.1, without approaching the possible short paths.

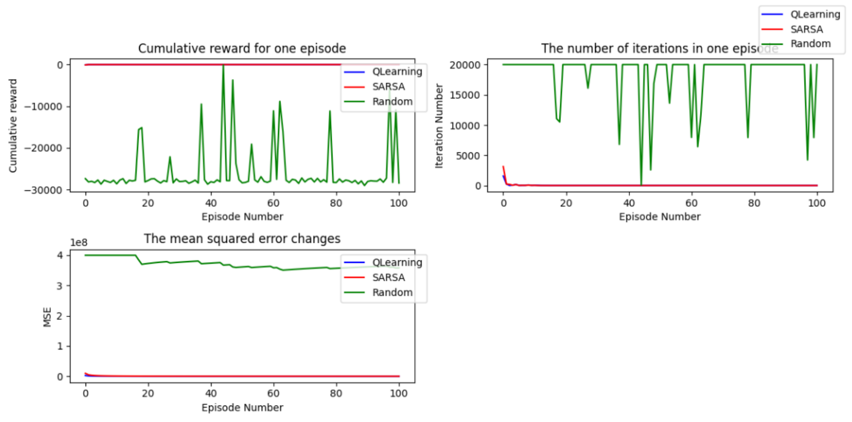


Figure 18 The Simpler Maze second results

As can be seen in all 3 tables, Random Actions worked with poor performance against Reinforcement Learning algorithms. Even if it reaches the target in some places, it often fails to reach the target in 20000 iterations. Since the mean squared error rate is quite high, Reinforcement Learning algorithms can be preferred over Random Actions.

5.2. Comparisons for The Complex Maze

For The Simpler Maze, three algorithms have been implemented, these are Q-Learning, SARSA and Random Actions; Q-Learning and SARSA are reinforcement learning algorithms. The shortest path in this maze is 20 iterations. Possible four shortcuts in Figure 19 can be seen with different colored arrow icons.

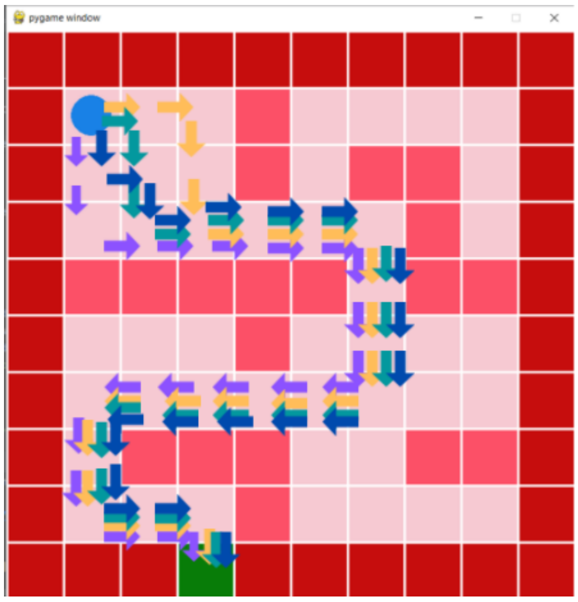


Figure 19 The Complex Maze short paths

5.2.1. Reinforcement Learning Algorithms

Reward calculations for the tables are made according to Figure 9. Based on 20 iterations in the shortest path, can be seen four of them in Figure 19, the mean square error rate was calculated accordingly.

As can be seen in Figure 12 and Figure 14, Q-Learning and SARSA found the shortest iteration number, albeit differently.

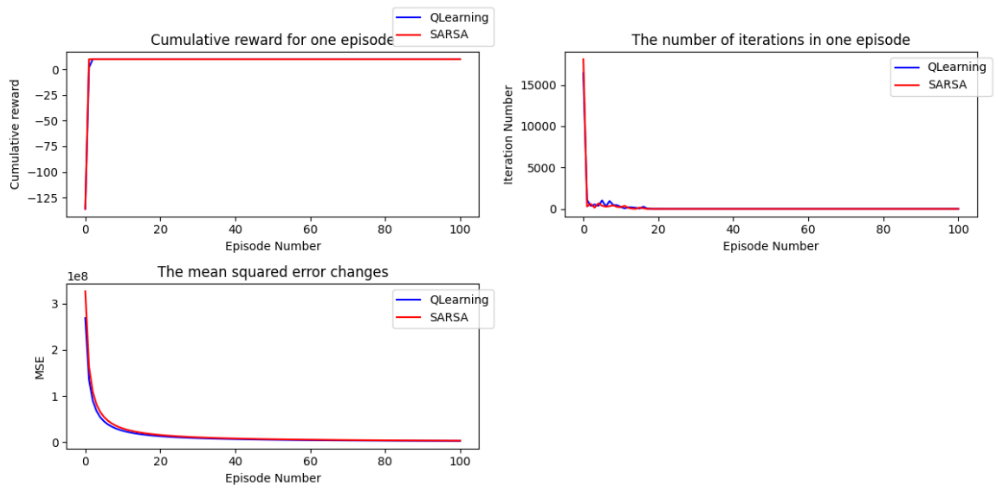


Figure 20 The Complex Maze first results

As can be seen in Figure 20, there is no difference in the points where the learning ends. It is seen that training seems to be over about in 18th episode and there is a policy to be implemented. While no difference is observed in the rewards obtained in each episode, there is a small difference in the number of iterations in each episode. In the number of iterations in each episode, the SARSA algorithm has consistently reduced the number of iterations except for the first two episodes, but the Q-Learning algorithm is unstable in some steps until it enables learning. Considering the mean square error rate, the Q-Learning algorithm kept the error rate smaller than the SARSA algorithm at first. The reason why the error rate was low for Q-Learning in the beginning is that SARSA wants to act according to the optimal policy while learning is taking place, but it takes a long time to search for a policy in very handicapped areas such as this complex maze.

As a result of these comparisons, if a faster learning is desired, the SARSA algorithm should be preferred, but if the error rate in the first episodes is important, the Q-Learning algorithm should be preferred.

5.2.2. Random Actions and Reinforcement Learning Algorithms

Random Actions did not show any learning. As stated in Chapter 4.3.2 in this maze, the agent could not reach the target point in 100 episodes.

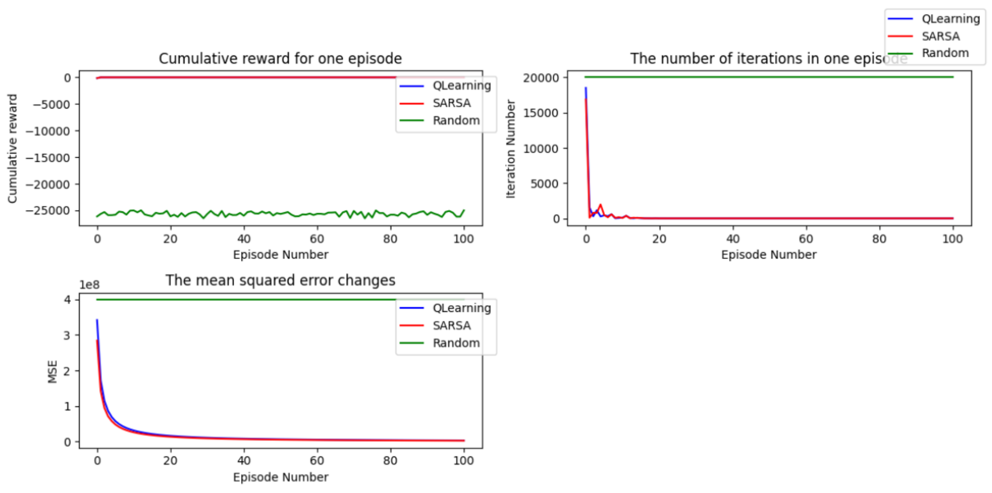


Figure 21 The Complex Maze second results

As seen in Figure 21, Random Actions worked with poor performance against Reinforcement Learning algorithms. It never reaches the target points. As a result, Reinforcement Learning algorithms can be preferred over Random Actions.

5.3. Comparisons of The Simpler Maze and The Complex Maze

In The Complex Maze, the number of iterations on the short paths to the target point is higher and due to various obstacles in the rooms, large iteration numbers are observed at first while learning takes place.

In The Simpler Maze, the agent could reach the target point by moving randomly, while in The Complex Maze, it could not achieve this.

In both mazes, Reinforcement Learning algorithms worked with high performance against Random actions.

In The Simpler Maze, the SARSA algorithm learned the policy learning faster and showed a preferable performance compared to the Q-Learning algorithm. This is because the SARSA algorithm is on-policy so policy discovery is faster. While SARSA performed faster in The Complex Maze, the error rate was lower in the Q-Learning algorithm at the beginning of learning.

When look at the both maze’s results, SARSA algorithm can be preferred for learning. Q-Learning assumes that an optimal policy is already being followed [11]. Therefore, it may show learning in an unstable structure at first. Since Q-Learning algorithm assumes the policy it determined at that moment as the most optimal policy, it may not be able to find the short path policy as in The Simpler Maze (Figure 11).

6. CONCLUSION AND FUTURE WORK

In this thesis, three algorithms are applied for two mazes. First of all, information about Reinforcement Learning is given. Then, the structure of mazes is explained. The results of the algorithms are shown for both mazes.

Algorithm comparisons and maze comparisons and the most suitable algorithm among Q-Learning, SARSA and Random Actions were examined. When the maze is simple, it is seen that the result is achieved even with random movements, while in a complex environment, the agent could not reach the result by acting randomly. Therefore, Reinforcement Learning algorithms were found to be better than random acting.

In Reinforcement Learning algorithms, it has been observed that SARSA learns the policy to be applied faster than Q-Learning and the agent acts more stable.

When we look at the effect of obstacles on learning, the high number of obstacles especially for Q-Learning contributed to the learning of the policy of finding the short paths.

For future works, it is aimed to implement this study in a real environment and to develop it using different algorithms.

REFERENCES

[1] P. K. Donepudi, "Reinforcement Learning for Robotic Grasping and Manipulation: A Review," *Asia Pac. j. energy environ.,* vol. 7, no. 2, pp. 66-78, 2020.

[2] H. Lu, Y. Li, S. Mu, D. Wang, H. Kim and S. Serikawa, "Motor Anomaly Detection for Unmanned Aerial Vehicles Using Reinforcement Learning," *IEEE Internet of Things Journals,* vol. 8, no. 4, pp. 2315-2322, 2018.

[3] H. Wang, S. Yuan, M. Guo, C.-Y. Chan, X. Li and W. Lan, "Tactical Driving Decisions of Unmanned Ground Vehicles in Complex Highway Environments: A Deep Reinforcement Approach," *Journal of Automobile Engineering,* vol. 235, no. 4, pp. 1113-1127, 2020.

[4] K. G. Vamvoudakis, H. Modares, B. Kiumarsi and F. L. Lewis, "Game Theory-Based Control System Algorithms with Real-Time Reinforcement Learning: How to Solve Multiplayer Games Online," *IEEE Control Systems Magazine,* vol. 37, no. 1, pp. 33-52, 2017.

[5] R. Kumar, P. Jitoko, S. Kumar, K. Pillay, P. Prakash, A. Sagar, R. Singh and U. Mehta, "Maze Solving Robot with Automated Obstacle Avoidance," *Procedia Computer Science,* vol. 105, pp. 57-61, 2017.

[6] H. Silaghi, E. Ecsedi, E. Mihok and V. Spoiala, "The Development of an Autonomomus Maze Robot," in *15th International Conference on Engineering of Modern Electric Systems*, Oradea, Romania, 13-14 June 2019.

[7] N. H. Barnouti, S. S. M. Al-Dabbagh and M. A. S. N. Naser, "Pathfinding in Strategy Games and Maze Solving Using A* Search Algorithm," *Journal of Computer and Comminications,* vol. 04, no. 11, 2016.

[8] O. Kathe, V. Turkar, A. Jagtap and G. Gidaye, "Maze Solving Robot Using Image Processing," in *IEEE Bombay Section Symposium*, Mumbai, India, 2015.

[9] B. F. Skinner, Science and Human Behavior, The Free Press; New Impression, 1953.

[10] S. R. Sutton and G. A. Barto, Reinforcement Learning: An Introduction, The MIT Press, 1998.

[11] J. Ravi, "Pluralsight," Pluralsight, Inc., [Online]. Available: https://app.pluralsight.com/library/courses/understanding-algorithms-reinforcement-learning/table-of-contents. [Accessed 25 April 2021].

[12] "Javatpoint," [Online]. Available: https://www.javatpoint.com/reinforcement-learning. [Accessed 5 May 2021].

ACKNOWLEDGEMENT

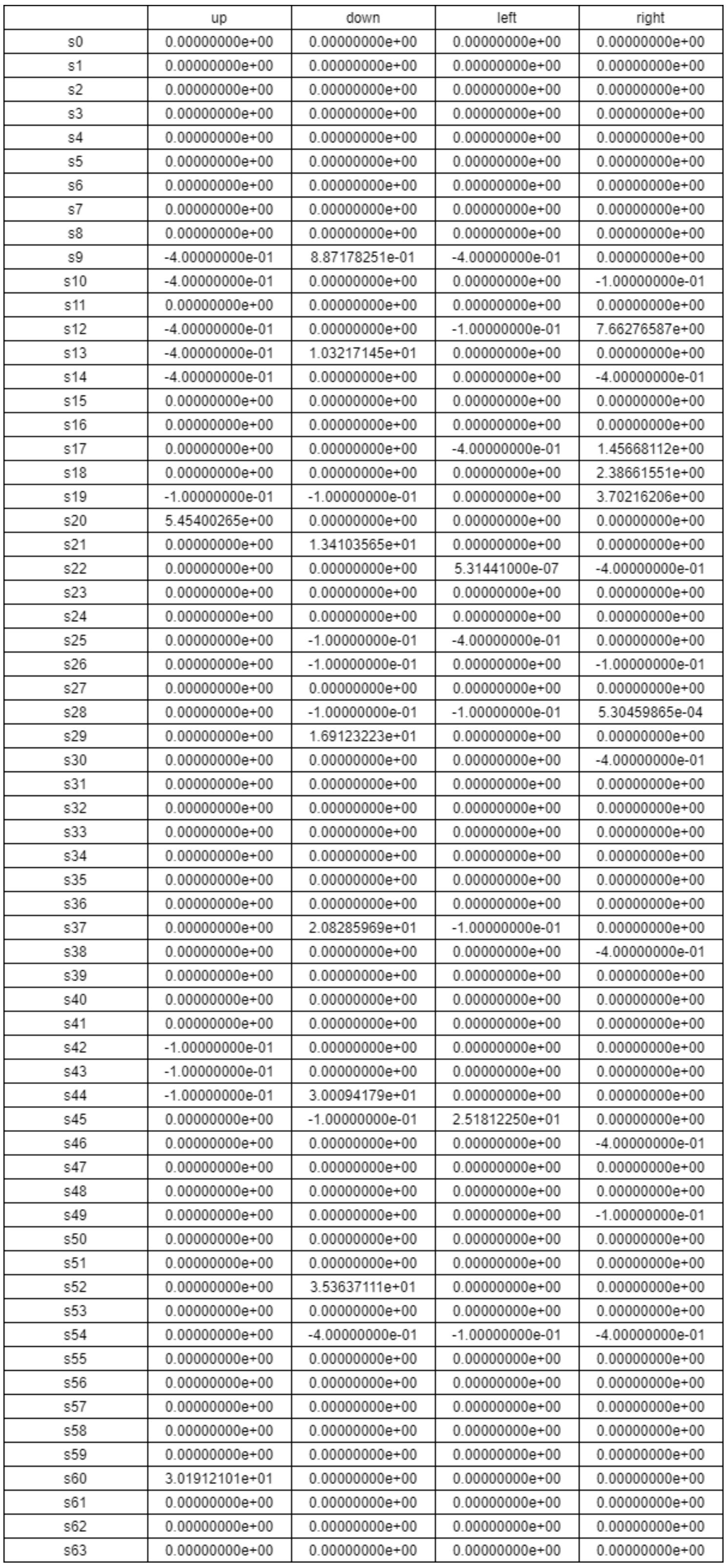
The study presented in this thesis was carried out under the supervision of Dr. Barış ATA. I would like to express my sincere gratitude to him for his patience, motivation, advice and time for this work.

I would like to thank to my family for their support. Finally, thanks to my Master Yoda İbrahim for being patient with me, making me feel like he is with me at every moment and always supporting me.

CURRICULUM VITAE

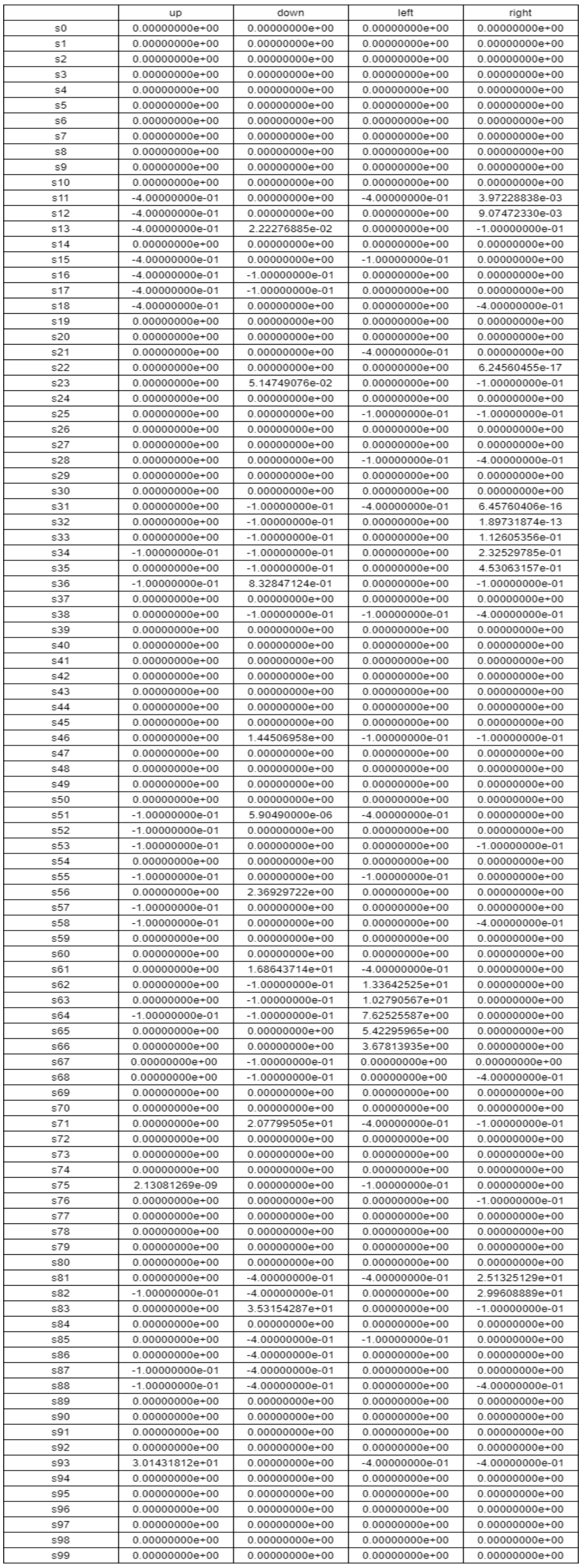
Zirveda AYTIMUR was born in Mersin, in 1997. She is a senior student at Çukurova University Computer Engineering department. Her research interests include machine learning, natural language processing and image processing.

APPENDIX A: TABLES



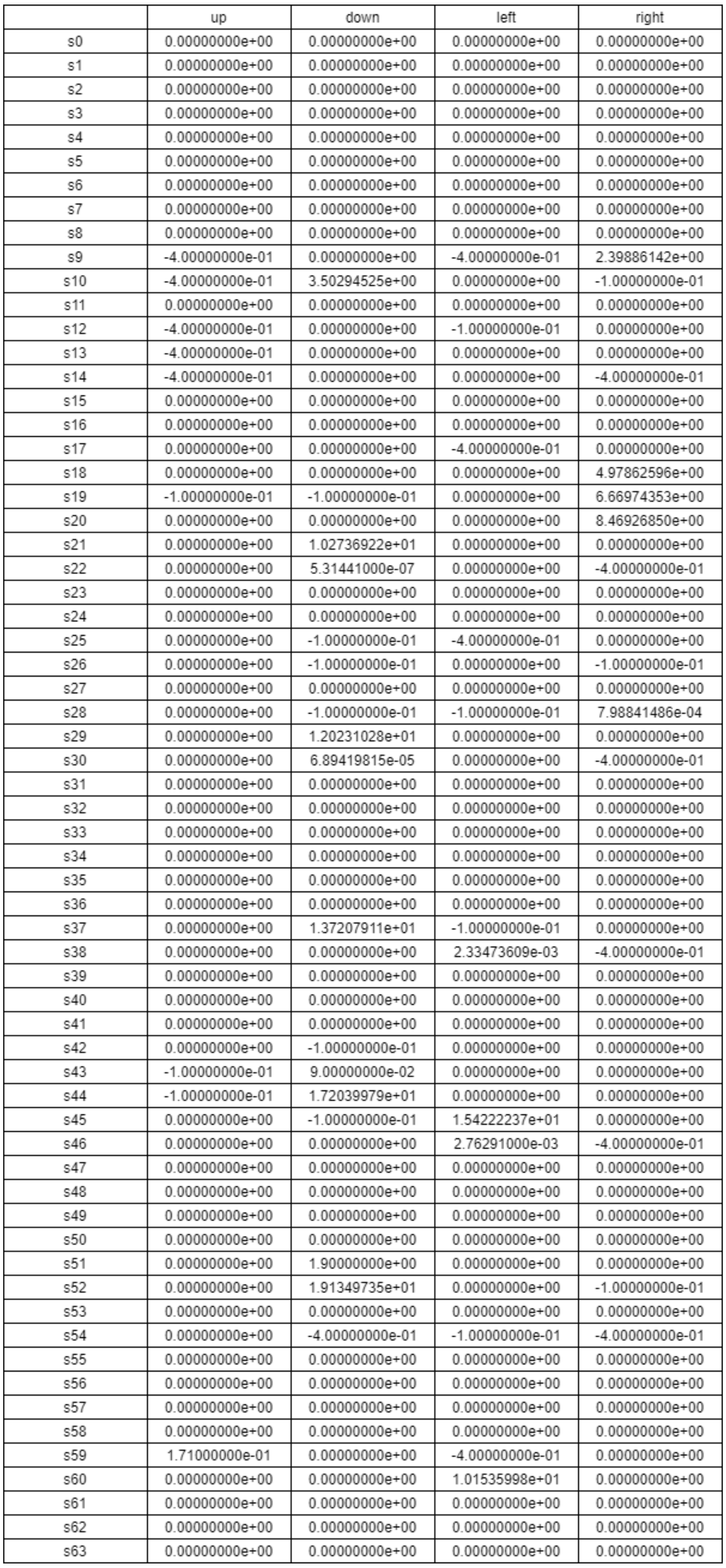
	up	down	left	right
s0	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s1	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s2	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s3	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s4	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s5	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s6	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s7	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s8	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s9	-4.00000000e-01	8.87178251e-01	-4.00000000e-01	0.00000000e+00
s10	-4.00000000e-01	0.00000000e+00	0.00000000e+00	-1.00000000e-01
s11	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s12	-4.00000000e-01	0.00000000e+00	-1.00000000e-01	7.66276587e+00
s13	-4.00000000e-01	1.03217145e+01	0.00000000e+00	0.00000000e+00
s14	-4.00000000e-01	0.00000000e+00	0.00000000e+00	-4.00000000e-01
s15	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s16	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s17	0.00000000e+00	0.00000000e+00	-4.00000000e-01	1.45668112e+00
s18	0.00000000e+00	0.00000000e+00	0.00000000e+00	2.38661551e+00
s19	-1.00000000e-01	-1.00000000e-01	0.00000000e+00	3.70216206e+00
s20	5.45400265e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s21	0.00000000e+00	1.34103565e+01	0.00000000e+00	0.00000000e+00
s22	0.00000000e+00	0.00000000e+00	5.31441000e-07	-4.00000000e-01
s23	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s24	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s25	0.00000000e+00	-1.00000000e-01	-4.00000000e-01	0.00000000e+00
s26	0.00000000e+00	-1.00000000e-01	0.00000000e+00	-1.00000000e-01
s27	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s28	0.00000000e+00	-1.00000000e-01	-1.00000000e-01	5.30459865e-04
s29	0.00000000e+00	1.69123223e+01	0.00000000e+00	0.00000000e+00
s30	0.00000000e+00	0.00000000e+00	0.00000000e+00	-4.00000000e-01
s31	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s32	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s33	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s34	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s35	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s36	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s37	0.00000000e+00	2.08285969e+01	-1.00000000e-01	0.00000000e+00
s38	0.00000000e+00	0.00000000e+00	0.00000000e+00	-4.00000000e-01
s39	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s40	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s41	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s42	-1.00000000e-01	0.00000000e+00	0.00000000e+00	0.00000000e+00
s43	-1.00000000e-01	0.00000000e+00	0.00000000e+00	0.00000000e+00
s44	-1.00000000e-01	3.00094179e+01	0.00000000e+00	0.00000000e+00
s45	0.00000000e+00	-1.00000000e-01	2.51812250e+01	0.00000000e+00
s46	0.00000000e+00	0.00000000e+00	0.00000000e+00	-4.00000000e-01
s47	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s48	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s49	0.00000000e+00	0.00000000e+00	0.00000000e+00	-1.00000000e-01
s50	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s51	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s52	0.00000000e+00	3.53637111e+01	0.00000000e+00	0.00000000e+00
s53	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s54	0.00000000e+00	-4.00000000e-01	-1.00000000e-01	-4.00000000e-01
s55	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s56	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s57	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s58	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s59	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s60	3.01912101e+01	0.00000000e+00	0.00000000e+00	0.00000000e+00
s61	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s62	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s63	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00

Table 1 Q Table for The Simpler Maze Q-Learning



	up	down	left	right
s0	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s1	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s2	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s3	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s4	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s5	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s6	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s7	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s8	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s9	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s10	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s11	-4.00000000e-01	0.00000000e+00	-4.00000000e-01	3.97228838e-03
s12	-4.00000000e-01	0.00000000e+00	0.00000000e+00	9.07472330e-03
s13	-4.00000000e-01	2.22276885e-02	0.00000000e+00	-1.00000000e-01
s14	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s15	-4.00000000e-01	0.00000000e+00	-1.00000000e-01	0.00000000e+00
s16	-4.00000000e-01	-1.00000000e-01	0.00000000e+00	0.00000000e+00
s17	-4.00000000e-01	-1.00000000e-01	0.00000000e+00	0.00000000e+00
s18	-4.00000000e-01	0.00000000e+00	0.00000000e+00	-4.00000000e-01
s19	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s20	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s21	0.00000000e+00	0.00000000e+00	-4.00000000e-01	0.00000000e+00
s22	0.00000000e+00	0.00000000e+00	0.00000000e+00	6.24560455e-17
s23	0.00000000e+00	5.14749076e-02	0.00000000e+00	-1.00000000e-01
s24	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s25	0.00000000e+00	0.00000000e+00	-1.00000000e-01	-1.00000000e-01
s26	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s27	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s28	0.00000000e+00	0.00000000e+00	-1.00000000e-01	-4.00000000e-01
s29	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s30	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s31	0.00000000e+00	-1.00000000e-01	-4.00000000e-01	6.45760406e-16
s32	0.00000000e+00	-1.00000000e-01	0.00000000e+00	1.89731874e-13
s33	0.00000000e+00	-1.00000000e-01	0.00000000e+00	1.12605356e-01
s34	-1.00000000e-01	-1.00000000e-01	0.00000000e+00	2.32529785e-01
s35	0.00000000e+00	-1.00000000e-01	0.00000000e+00	4.53063157e-01
s36	-1.00000000e-01	8.32847124e-01	0.00000000e+00	-1.00000000e-01
s37	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s38	0.00000000e+00	-1.00000000e-01	-1.00000000e-01	-4.00000000e-01
s39	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s40	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s41	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s42	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s43	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s44	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s45	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s46	0.00000000e+00	1.44506958e+00	-1.00000000e-01	-1.00000000e-01
s47	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s48	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s49	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s50	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s51	-1.00000000e-01	5.90490000e-06	-4.00000000e-01	0.00000000e+00
s52	-1.00000000e-01	0.00000000e+00	0.00000000e+00	0.00000000e+00
s53	-1.00000000e-01	0.00000000e+00	0.00000000e+00	-1.00000000e-01
s54	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s55	-1.00000000e-01	0.00000000e+00	-1.00000000e-01	0.00000000e+00
s56	0.00000000e+00	2.36929722e+00	0.00000000e+00	0.00000000e+00
s57	-1.00000000e-01	0.00000000e+00	0.00000000e+00	0.00000000e+00
s58	-1.00000000e-01	0.00000000e+00	0.00000000e+00	-4.00000000e-01
s59	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s60	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s61	0.00000000e+00	1.68643714e+01	-4.00000000e-01	0.00000000e+00
s62	0.00000000e+00	-1.00000000e-01	1.33642525e+01	0.00000000e+00
s63	0.00000000e+00	-1.00000000e-01	1.02790567e+01	0.00000000e+00
s64	-1.00000000e-01	-1.00000000e-01	7.62525587e+00	0.00000000e+00
s65	0.00000000e+00	0.00000000e+00	5.42295965e+00	0.00000000e+00
s66	0.00000000e+00	0.00000000e+00	3.67813935e+00	0.00000000e+00
s67	0.00000000e+00	-1.00000000e-01	0.00000000e+00	0.00000000e+00
s68	0.00000000e+00	-1.00000000e-01	0.00000000e+00	-4.00000000e-01
s69	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s70	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s71	0.00000000e+00	2.07799505e+01	-4.00000000e-01	-1.00000000e-01
s72	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s73	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s74	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s75	2.13081269e-09	0.00000000e+00	-1.00000000e-01	0.00000000e+00
s76	0.00000000e+00	0.00000000e+00	0.00000000e+00	-1.00000000e-01
s77	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s78	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s79	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s80	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s81	0.00000000e+00	-4.00000000e-01	-4.00000000e-01	2.51325129e+01
s82	-1.00000000e-01	-4.00000000e-01	0.00000000e+00	2.99608889e+01
s83	0.00000000e+00	3.53154287e+01	0.00000000e+00	-1.00000000e-01
s84	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s85	0.00000000e+00	-4.00000000e-01	-1.00000000e-01	0.00000000e+00
s86	0.00000000e+00	-4.00000000e-01	0.00000000e+00	0.00000000e+00
s87	-1.00000000e-01	-4.00000000e-01	0.00000000e+00	0.00000000e+00
s88	-1.00000000e-01	-4.00000000e-01	0.00000000e+00	-4.00000000e-01
s89	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s90	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s91	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s92	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s93	3.01431812e+01	0.00000000e+00	-4.00000000e-01	-4.00000000e-01
s94	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s95	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s96	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s97	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s98	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s99	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00

Table 2 Q Table for The Complex Maze Q-Learning



	up	down	left	right
s0	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s1	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s2	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s3	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s4	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s5	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s6	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s7	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s8	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s9	-4.00000000e-01	0.00000000e+00	-4.00000000e-01	2.39886142e+00
s10	-4.00000000e-01	3.50294525e+00	0.00000000e+00	-1.00000000e-01
s11	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s12	-4.00000000e-01	0.00000000e+00	-1.00000000e-01	0.00000000e+00
s13	-4.00000000e-01	0.00000000e+00	0.00000000e+00	0.00000000e+00
s14	-4.00000000e-01	0.00000000e+00	0.00000000e+00	-4.00000000e-01
s15	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s16	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s17	0.00000000e+00	0.00000000e+00	-4.00000000e-01	0.00000000e+00
s18	0.00000000e+00	0.00000000e+00	0.00000000e+00	4.97862596e+00
s19	-1.00000000e-01	-1.00000000e-01	0.00000000e+00	6.66974353e+00
s20	0.00000000e+00	0.00000000e+00	0.00000000e+00	8.46926850e+00
s21	0.00000000e+00	1.02736922e+01	0.00000000e+00	0.00000000e+00
s22	0.00000000e+00	5.31441000e-07	0.00000000e+00	-4.00000000e-01
s23	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s24	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s25	0.00000000e+00	-1.00000000e-01	-4.00000000e-01	0.00000000e+00
s26	0.00000000e+00	-1.00000000e-01	0.00000000e+00	-1.00000000e-01
s27	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s28	0.00000000e+00	-1.00000000e-01	-1.00000000e-01	7.98841486e-04
s29	0.00000000e+00	1.20231028e+01	0.00000000e+00	0.00000000e+00
s30	0.00000000e+00	6.89419815e-05	0.00000000e+00	-4.00000000e-01
s31	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s32	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s33	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s34	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s35	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s36	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s37	0.00000000e+00	1.37207911e+01	-1.00000000e-01	0.00000000e+00
s38	0.00000000e+00	0.00000000e+00	2.33473609e-03	-4.00000000e-01
s39	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s40	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s41	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s42	0.00000000e+00	-1.00000000e-01	0.00000000e+00	0.00000000e+00
s43	-1.00000000e-01	9.00000000e-02	0.00000000e+00	0.00000000e+00
s44	-1.00000000e-01	1.72039979e+01	0.00000000e+00	0.00000000e+00
s45	0.00000000e+00	-1.00000000e-01	1.54222237e+01	0.00000000e+00
s46	0.00000000e+00	0.00000000e+00	2.76291000e-03	-4.00000000e-01
s47	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s48	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s49	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s50	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s51	0.00000000e+00	1.90000000e+00	0.00000000e+00	0.00000000e+00
s52	0.00000000e+00	1.91349735e+01	0.00000000e+00	-1.00000000e-01
s53	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s54	0.00000000e+00	-4.00000000e-01	-1.00000000e-01	-4.00000000e-01
s55	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s56	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s57	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s58	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s59	1.71000000e-01	0.00000000e+00	-4.00000000e-01	0.00000000e+00
s60	0.00000000e+00	0.00000000e+00	1.01535998e+01	0.00000000e+00
s61	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s62	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s63	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00

Table 3 Q Table for The Simpler Maze SARSA

	up	down	left	right
s0	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s1	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s2	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s3	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s4	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s5	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s6	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s7	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s8	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s9	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s10	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s11	-4.00000000e-01	4.01807307e-03	-4.00000000e-01	0.00000000e+00
s12	-4.00000000e-01	0.00000000e+00	0.00000000e+00	0.00000000e+00
s13	-4.00000000e-01	0.00000000e+00	0.00000000e+00	-1.00000000e-01
s14	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s15	-4.00000000e-01	0.00000000e+00	-1.00000000e-01	0.00000000e+00
s16	-4.00000000e-01	-1.00000000e-01	0.00000000e+00	0.00000000e+00
s17	-4.00000000e-01	-1.00000000e-01	0.00000000e+00	0.00000000e+00
s18	-4.00000000e-01	0.00000000e+00	0.00000000e+00	-4.00000000e-01
s19	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s20	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s21	0.00000000e+00	9.17768388e-03	-4.00000000e-01	0.00000000e+00
s22	0.00000000e+00	6.10972198e-15	0.00000000e+00	0.00000000e+00
s23	0.00000000e+00	0.00000000e+00	8.74555670e-17	-1.00000000e-01
s24	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s25	0.00000000e+00	0.00000000e+00	-1.00000000e-01	-1.00000000e-01
s26	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s27	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s28	0.00000000e+00	0.00000000e+00	-1.00000000e-01	-4.00000000e-01
s29	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s30	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s31	0.00000000e+00	-1.00000000e-01	-4.00000000e-01	2.24736754e-02
s32	0.00000000e+00	-1.00000000e-01	0.00000000e+00	5.20272405e-02
s33	0.00000000e+00	-1.00000000e-01	0.00000000e+00	1.13768722e-01
s34	-1.00000000e-01	-1.00000000e-01	0.00000000e+00	2.34823937e-01
s35	0.00000000e+00	-1.00000000e-01	0.00000000e+00	4.57290733e-01
s36	-1.00000000e-01	8.40114091e-01	0.00000000e+00	-1.00000000e-01
s37	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s38	0.00000000e+00	-1.00000000e-01	-1.00000000e-01	-4.00000000e-01
s39	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s40	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s41	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s42	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s43	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s44	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s45	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s46	0.00000000e+00	1.45670447e+00	-1.00000000e-01	-1.00000000e-01
s47	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s48	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s49	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s50	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s51	-1.00000000e-01	5.90490000e-06	-4.00000000e-01	0.00000000e+00
s52	-1.00000000e-01	0.00000000e+00	0.00000000e+00	0.00000000e+00
s53	-1.00000000e-01	0.00000000e+00	0.00000000e+00	-1.00000000e-01
s54	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s55	-1.00000000e-01	0.00000000e+00	-1.00000000e-01	0.00000000e+00
s56	0.00000000e+00	2.38663168e+00	0.00000000e+00	0.00000000e+00
s57	-1.00000000e-01	0.00000000e+00	0.00000000e+00	0.00000000e+00
s58	-1.00000000e-01	0.00000000e+00	0.00000000e+00	-4.00000000e-01
s59	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s60	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s61	0.00000000e+00	1.69123223e+01	-4.00000000e-01	0.00000000e+00
s62	0.00000000e+00	-1.00000000e-01	1.34103565e+01	0.00000000e+00
s63	0.00000000e+00	-1.00000000e-01	1.03217148e+01	0.00000000e+00
s64	-1.00000000e-01	-1.00000000e-01	7.66276726e+00	0.00000000e+00
s65	0.00000000e+00	0.00000000e+00	5.45400681e+00	0.00000000e+00
s66	0.00000000e+00	0.00000000e+00	3.70217127e+00	0.00000000e+00
s67	0.00000000e+00	-1.00000000e-01	0.00000000e+00	0.00000000e+00
s68	0.00000000e+00	-1.00000000e-01	0.00000000e+00	-4.00000000e-01
s69	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s70	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s71	0.00000000e+00	2.08285969e+01	-4.00000000e-01	-1.00000000e-01
s72	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s73	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s74	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s75	0.00000000e+00	0.00000000e+00	-1.00000000e-01	0.00000000e+00
s76	0.00000000e+00	0.00000000e+00	0.00000000e+00	-1.00000000e-01
s77	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s78	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s79	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s80	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s81	0.00000000e+00	-4.00000000e-01	-4.00000000e-01	2.51812250e+01
s82	-1.00000000e-01	-4.00000000e-01	0.00000000e+00	3.00094179e+01
s83	-1.00000000e-01	3.53637111e+01	0.00000000e+00	0.00000000e+00
s84	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s85	0.00000000e+00	-4.00000000e-01	-1.00000000e-01	0.00000000e+00
s86	0.00000000e+00	-4.00000000e-01	0.00000000e+00	0.00000000e+00
s87	-1.00000000e-01	-4.00000000e-01	0.00000000e+00	0.00000000e+00
s88	-1.00000000e-01	-4.00000000e-01	0.00000000e+00	-4.00000000e-01
s89	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s90	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s91	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s92	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s93	3.01912101e+01	0.00000000e+00	0.00000000e+00	0.00000000e+00
s94	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s95	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s96	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s97	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s98	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00
s99	0.00000000e+00	0.00000000e+00	0.00000000e+00	0.00000000e+00

Table 4 Q Table for The Simpler Maze SARSA