ECE 6604 : Project 4
Reinforcement learning:Policy Iteration

Author : Parisha Joshi- parishamaheshj18@vt.edu

Introduction

The project was based on reinforcement learning algorithm and implementation of policy iteration to train the agent based on reward and penalty given by the environment to achieve maximum cumulative reward. There are total 4 states : ["00","01","10","11"] and 4 possible actions:["00","01","10","11"] . The reward is computed based on number of collisions and number of channel the environment uses to communicate. Policy can be considered as a mapping between state to the action. Here utility function is used to iterate over policies. The convergence takes place if policy in previous iteration matches the policy aquired in current iteration

1 Question 1

Question 1 asks to make a transition matrix and reward matrix on the basis of 10000 observations. The environment was coded based on the given table in the question paper and agent starts to collect rewards and makes the transition matrix. At the end of 10000 iterations the resulted Transition matrix is shown in Figure 1.

```
val(:,:,1) =

    1.0000         0         0         0
         0    1.0000         0         0
         0         0    1.0000         0
    0.1916         0         0    0.8084


val(:,:,2) =

         0         0    1.0000         0
         0         0    0.7879    0.2121
         0         0    0.7851    0.2149
    0.1877         0         0    0.8123


val(:,:,3) =

         0    1.0000         0         0
         0    0.8131         0    0.1869
         0    0.7638         0    0.2362
    0.1951         0         0    0.8049


val(:,:,4) =

    0.1741         0         0    0.8259
    0.2383         0         0    0.7617
    0.1899         0         0    0.8101
    0.2226         0         0    0.7774
```

Figure 1: Transition Matrix, initial state "01",Dimension: (Initial-state,Next-state,Action)

The dimension of the transition matrix states –initial-state, next-state and action-taken. Hence,it is a 4*4*4 matrix. The Transition matrix matches the table provided in the question set.

```
val(:,:,1) =

     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0

val(:,:,2) =

     0     0    10     0
     0     0    10   -40
     0     0    10   -40
    10     0     0   -40

val(:,:,3) =

     0    10     0     0
     0    10     0   -40
     0    10     0   -40
    10     0     0   -40

val(:,:,4) =

    20     0     0   -80
    20     0     0   -80
    20     0     0   -80
    20     0     0   -80
```

Figure 2: Reward Matrix, Initial state "01",Dimension: (Initial-state,Next-state,Action)

On the other hand, the reward matrix Figure 2 shows the immediate rewards agent gets after taking action a, from state s' to reach state s,which matches the Table given in the question paper as well.

2 Question 2

Policy iteration algorithm consists of two main functionalities. 1) policy evaluation and 2) Policy improvement. Policy is the mapping between state to action. In simpler terms it instructs agent if it observes x state is should take y action.In the question there are 4 states and 4 possible actions. So my policy would be an array of 4 elements with action for state 1 is the first element, action for state 2 would be second element and so on. The initial policy is assigned randomly. For iterating policy there is an approach of Dynamic. In this particular problem reward is based on previous state, action and next state. So in calculation of optimum utility function it is important to take R inside the summation.

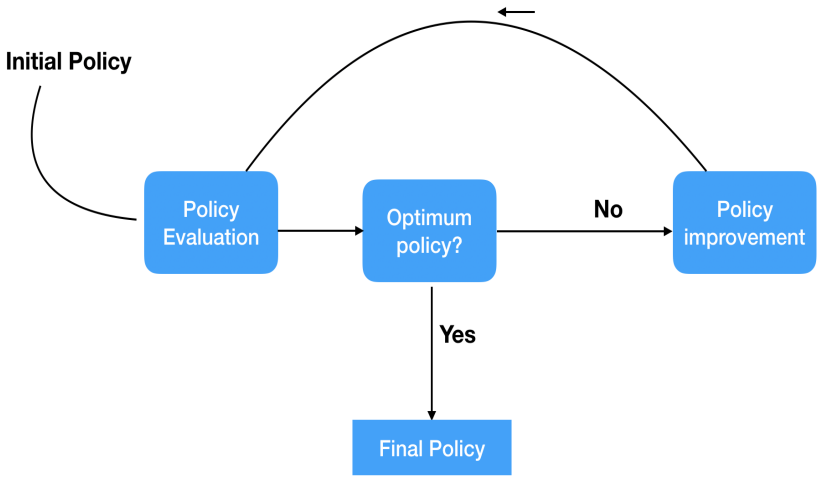

Figure 3: Policy Iteration Algorithm

Figure 3 shows the algorithm implemented for policy iteration. The question arises as to how can the policy be evaluated. For this purpose, the code makes use of the utility function. Utility function is the value determined by calculating immediate reward with the discounted utility from the future states. as given in [1] the value function is defined as $U_\pi(s) = R(S,\pi) + \gamma \sum_{s'} T(s,a,s')U_\pi(s')$. Here, U is the Utility function.The final policy iterates to be ["01","01","01","00"] after finite number of iterations. Ebsilon is a threshold value which is to be compared with abs(utility - old-utility). Here the utility function would converge once the difference between new and old value of utility will be less than Ebsilon(set at 0.2).Gamma is a discount factor which is multiplied with the utility value of previous iteration. Here gamma equals to 0 means there is no Exploitation of previous information in the learning progress based on the past utility values. gamma 1 means there is 100 percent exploitation of the information. The success value of gamma is between 0.9 to 0.99 for both exploration and exploitation balance.

3 Question 3

Taking the optimum policy derived from the task 2, initial state "10" and with 10000 steps rewards plot is shown in Figure 4. As can be seen from the graph, the agent always chooses the path which would result in the maximum cumulative reward in future. Which does not mean that the number of collisions would glide to 0 drastically. The frequency of collisions would decreases.As can be seen from Figure 6, the frequency of 1 collision is pretty low compared to taking random actions where there could be at least one collision on every step.

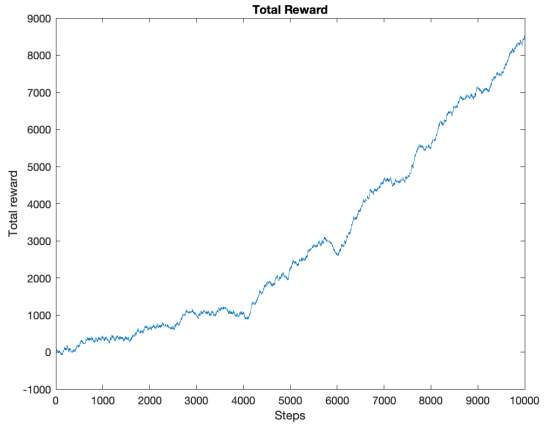

Figure 4: Total Reward till step

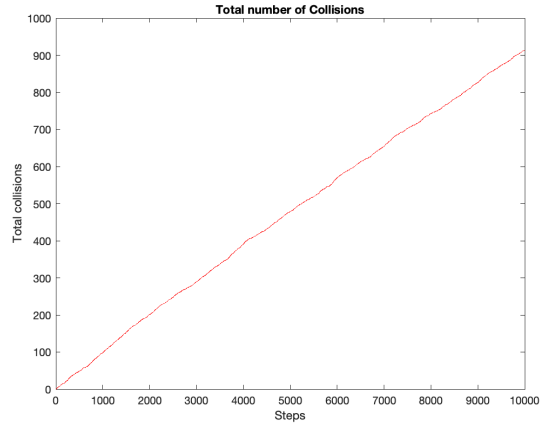

Figure 5: Total number of Collisions till respective step

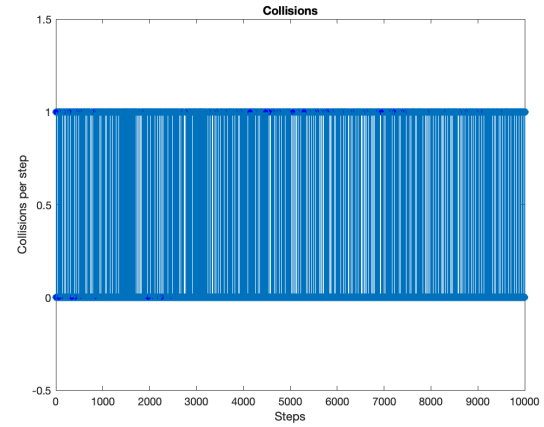

Figure 6: collision at step

4 Question 4

In the reactive system the environment would give response in a way that the number of collisions would be 0 for every step. As the environment only uses the frequency bad that is not in use. The Figures below shows the comparison between the previous system and the reactive system rewards and collisions. As the number of collisions are always going to be 0, The reward stays constant for every step unless the action is "11" which is not in the optimum iterated policy.

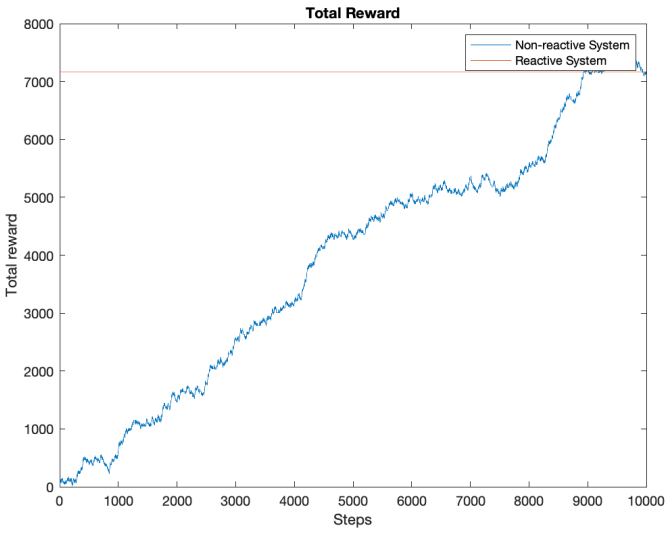

Figure 7: Comparison of Rewards between reactive and original system

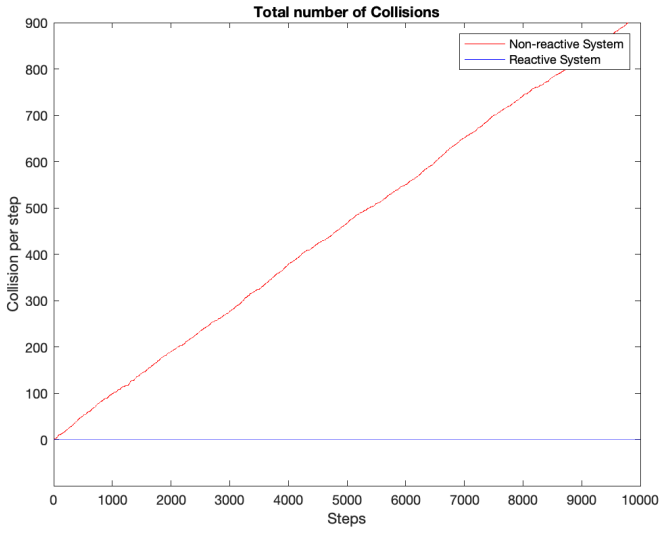

Figure 8: Comparison of cumulative Collisions of reactive and original systems

5 Conclusion

Conclusively,The reinforcement learning policy iteration can be used to train an agent to find an optimal policy. changing the constants such as gamma, ebsilon, and number of iterations can lead to different policies which is supposed to be tuned according to the long term reward achieved by the policy. For this project, ebsilon value was set at 0.2, gamma at 0.9, and number of iterations to 10000.After policy iteration, the optimum policy was [”01”,”01”,”01”,”00”] which is responsible for the long term high reward gain.

5.1 Matlab files

There are 4 matlab files: Question1.m, Question2.m, Question3.m and Question4.m. The results can be seen by running them without changing any variables.

References

[1] Policy Iteration : Carnegie Melon
`https://www.cs.cmu.edu/afs/cs/project/jair/pub/volume4/kaelbling96a-html/node20.html`

[2] Basic Algorithm
`https://webee.technion.ac.il/people/shimkin/LCS11/ch4-RL1.pdf`

[3] Value and Policy Iterations:
`https://medium.com/@m.alzantot/deep-reinforcement-learning-demysitifed-episode-2-policy-iteration-value-iteration-and-q-978f9e89ddaa`

[4] Value and Policy Iterations:
`Cource Handouts by Dr.Buherer, Dr.Dhillon, Virginia Tech`