Avoiding Negative Side Effects by Considering Others

Parand Alizadeh Alamdari, Toryn Q. Klassen, Rodrigo Toro Icarte, Sheila A. McIlraith

Paper review for reinforcement learning end semester project

Dinesh Adhithya H, Surya Prasath R

Motivation

In multi-agent systems, multiple agents are present and each tries to optimize its tasks such that it can maximize its reward. Agents act in such a manner that they maximize their rewards, neglecting other agents could lead to negative side effects which may not be explicitly part of the objective function but is something we would expect an agent to learn implicitly through some sort of modified learning algorithm. A simple example to illustrate the above point is considering a robot whose task is to deliver coffee to customers in a coffee shop. The robot would try to minimize the distance travelled from the coffee machine to the customer's table. In such a case an optimal path could be such that another customer is in the optimal path, the robot can't just run over a customer. Hence such a robot shall consider the other agent in the coffee shop so that such negative side effects can be avoided.

In this work, the authors make use of Q-Learning, in multi-agent system environments. Such that some agents also care about the rewards of other agents to avoid negative side effects.

Problem Statement

We wish to construct RL agents which will go after the well being of other agents in the environment. The model is defined as a Markov Decision Process (MDP) (S,A,T, r, γ) , where S is a set of states, A is a set of actions , T gives transition probability from a given state and action to another state , r is the reward function and γ is the discount factor.A policy is defined which gives a state map to an action. The value function gives the expected return in a state given a policy . An optimal policy tries to maximize the value for every state.

They defined an augmented reward function:

rvalue(s, a, s') = **α1 · r1(s,a,s')** if s' is not terminal

α1 · r1(s,a,s') + γ · α2 · F(V,P,s') if s' is terminal

r1 is the acting agent's reward , F is some function . α1 and α2 are called caring coefficients.There are different definitions of F possible such as :

Minimum (V(s0) , V(s)) penalize only negative reward
Minimum V(s') worst-case future reward
V(s) expected future return

Using V(s) would lead to "positive effects as well" , such as agent 1 trying to maximize another agent 2. Using Minimum (V(s0) , V(s)) would lead to "avoiding negative effects" , in which agent 1 only tries to avoid causing negative side effects to another agent 2 , but doesnt try to maximize. S0 is the initial state and S is the current state.

Experiments and approach

We constructed a 2 agent system, where both the agents can move along the grid world and such that each agent has a particular door through which he has to enter to get a positive reward. To open the door an agent needs to pick the key for its corresponding door while traveling along the grid. We want to experiment with various augmented reward functions, and caring parameters and observe the overall reward obtained from the environment. Reaching the goal gets an agent of a score of 0 and every step gets the agent of a score of -1.

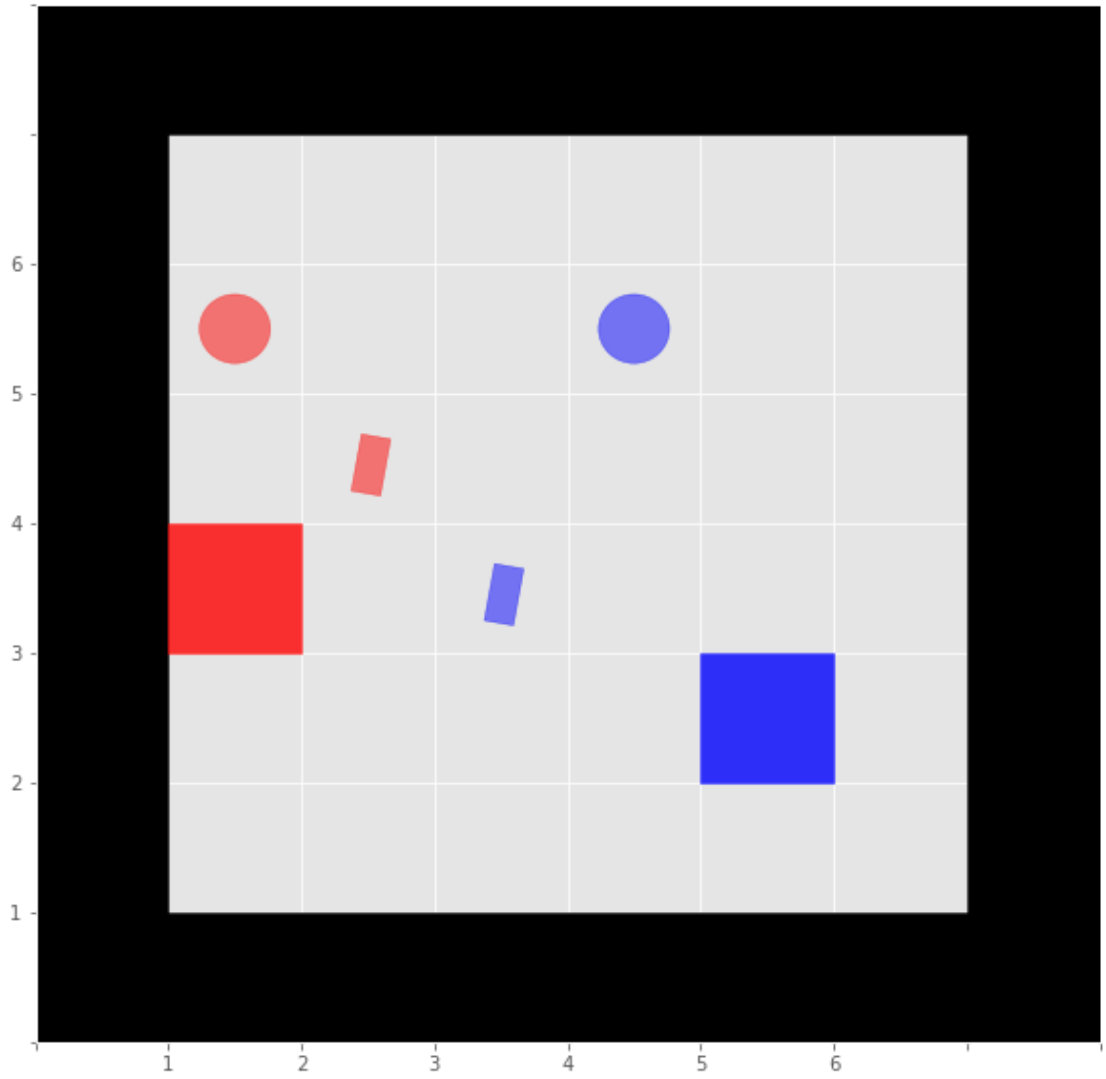


Figure1: Shows a screenshot of the environment made for the below experiments. The 2 agents are marked in red and blue circles, doors as squares and keys as rectangles.

The impact of considering others

The paper makes use of a Q-Learning based approach to learn the Q-function which gives the expected reward for a state and action pair. We wish to learn this function and try to find the optimal path.

Learning the Q-function of agent 1 alone, where we simply try to maximize the reward of agent 1 and agent 2 rewards aren't considered.

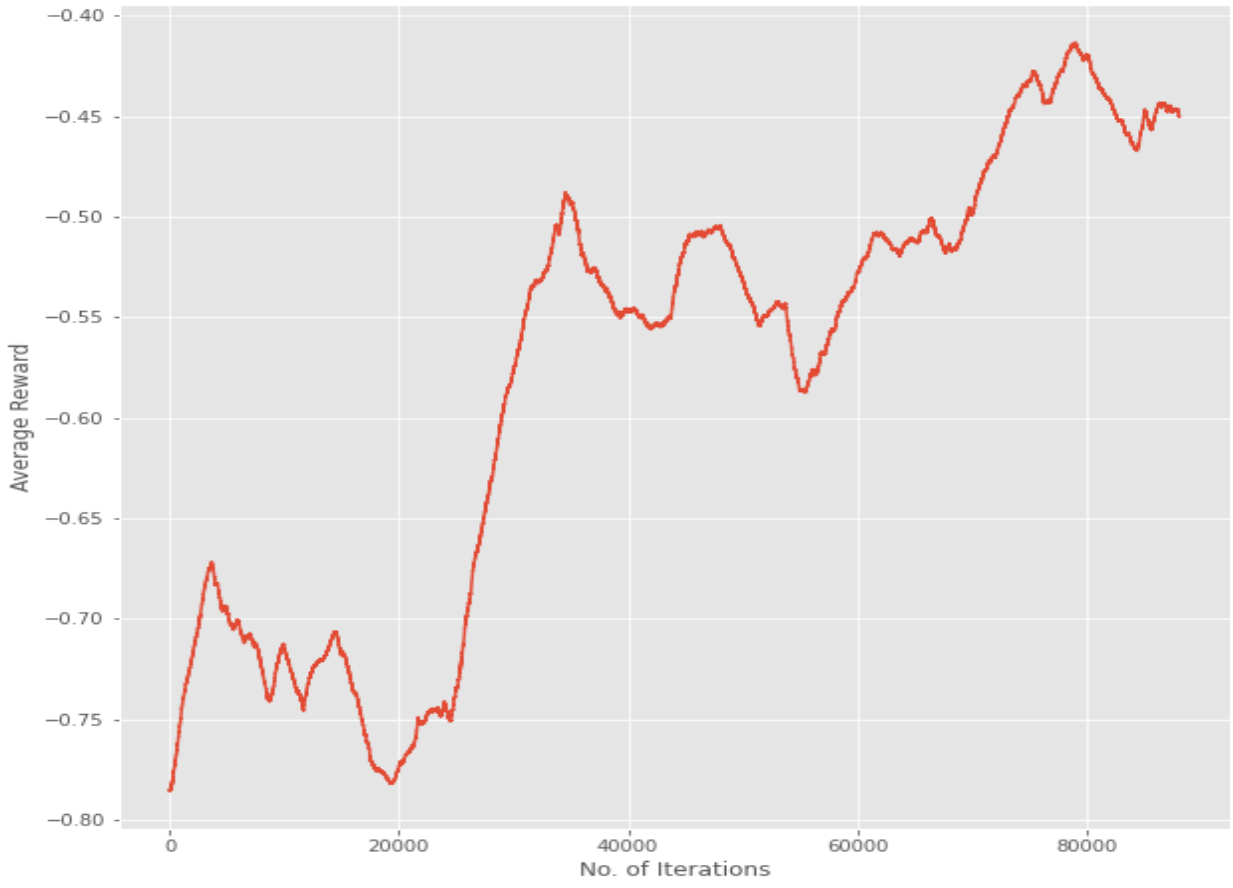


Figure2: Figure showing the average reward obtained over various iterations of Q-Learning based approach for agent1

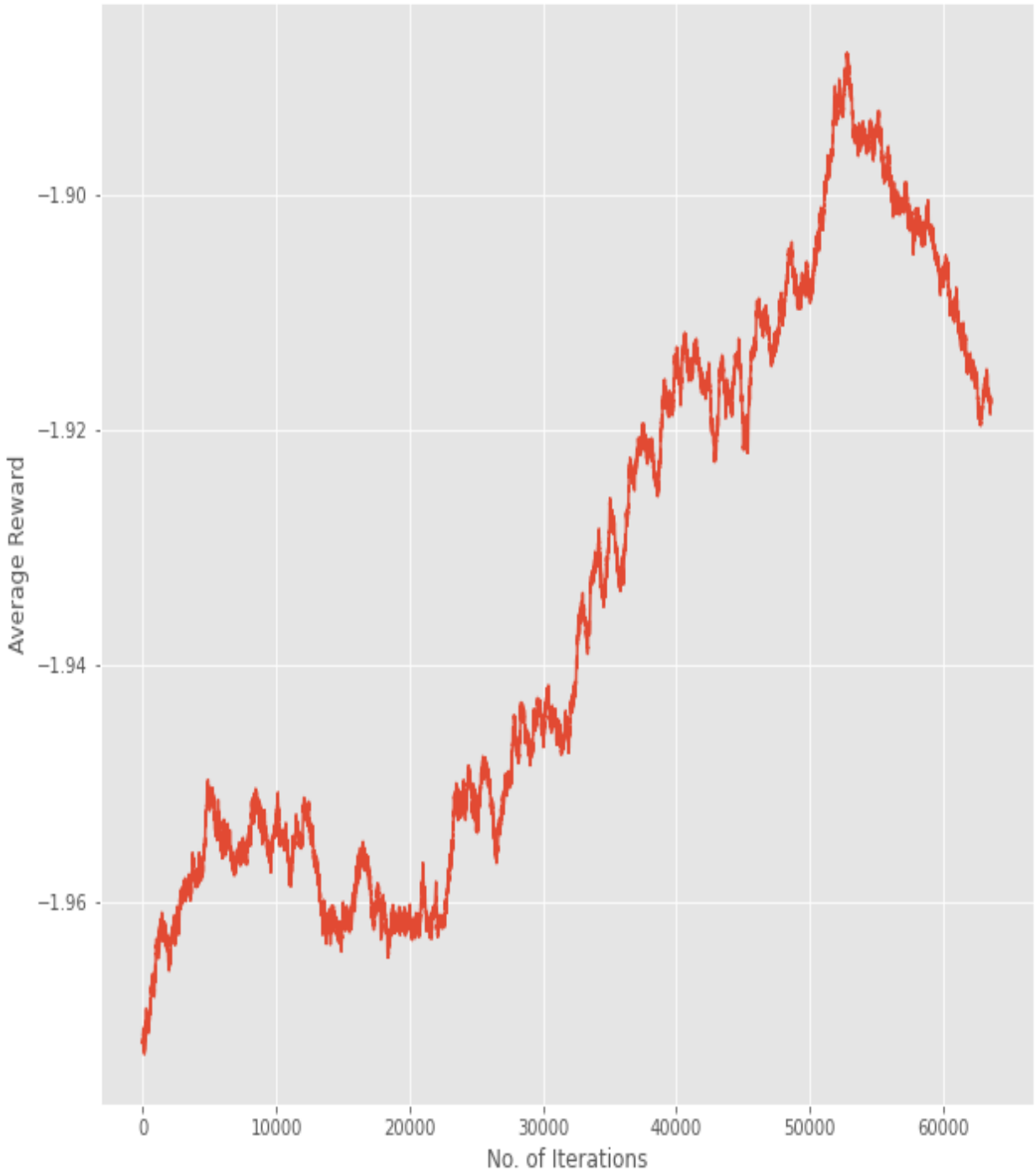


Figure3: Figure showing the average reward obtained over various iterations of Q-Learning based approach for agent2, with caring parameter as 1.0 for agent 1's Q-function.

Illustration of optimal behaviour under different reward augmentations

Minimum (V(s0), V(s)) penalize only negative reward

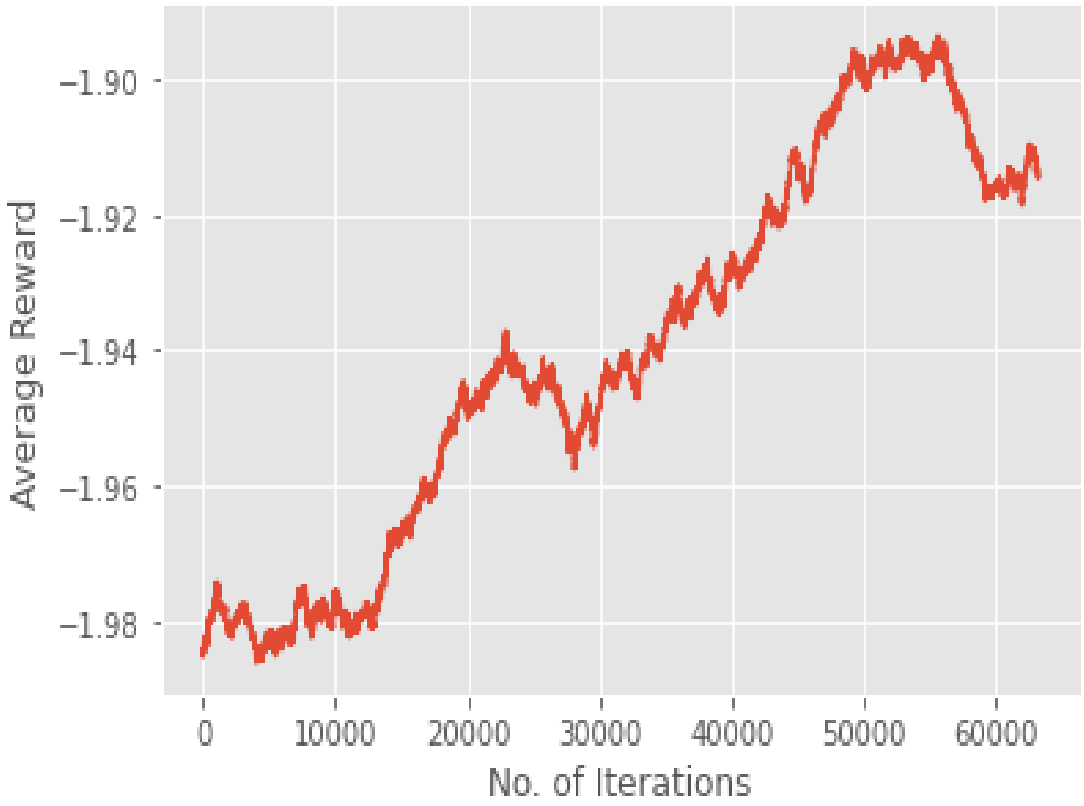


Figure 4: Figure showing the average reward obtained over various iterations of Q-Learning based approach for agent2, with caring parameter as 0.5 for agent 1's Q-function with min(V(s), V(s0)) augmented reward function.

V(s) expected future return

Figure 5: Figure showing the average reward obtained over various iterations of Q-Learning based approach for agent2, with a caring parameter as 0.5 for agent 1's Q-function with V(s) augmented reward function.

Minimum V(s') worst-case future reward

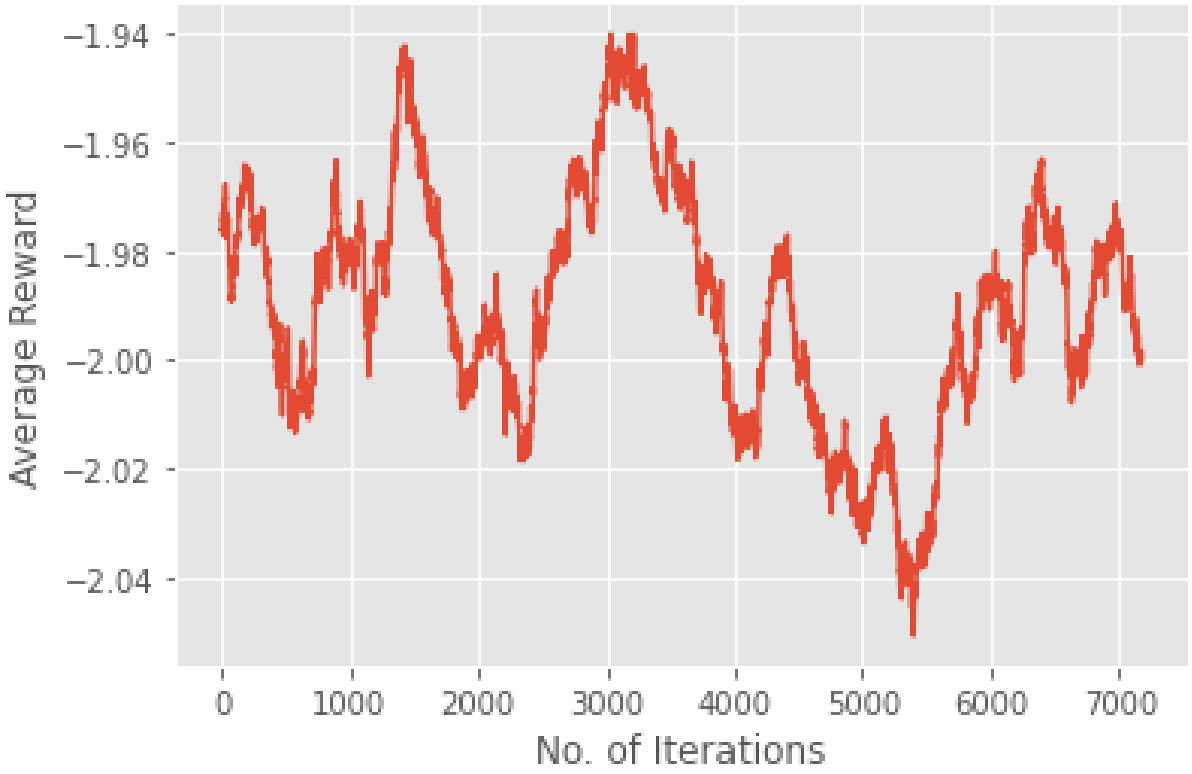
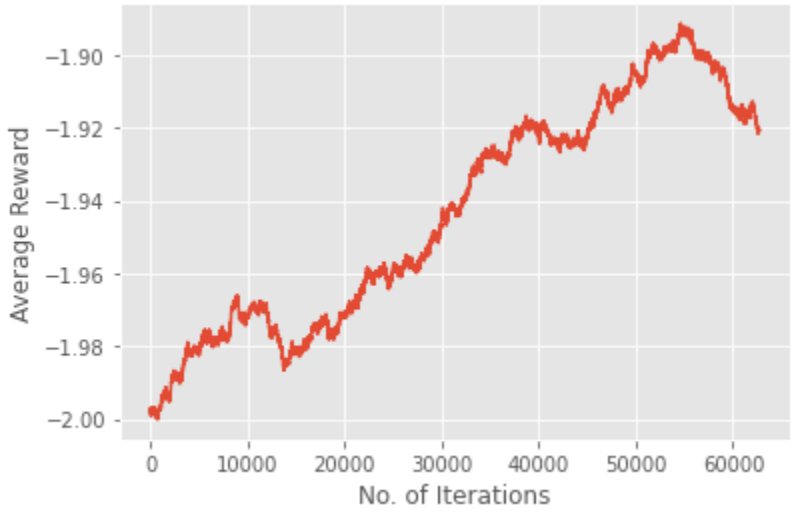


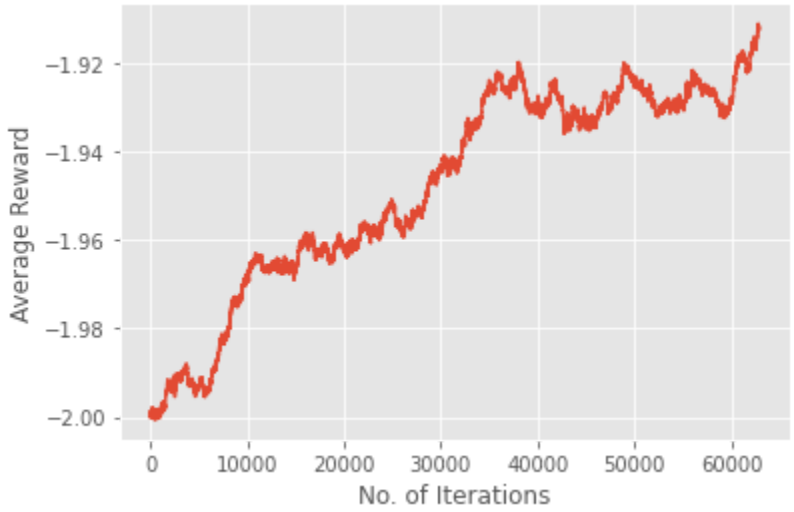
Figure 5: Figure showing the average reward obtained over various iterations of Q-Learning based approach for agent2, with a caring parameter as 0.5 for agent 1's Q-function with Minimum(V(s)) augmented reward function.

Varying the caring coefficient

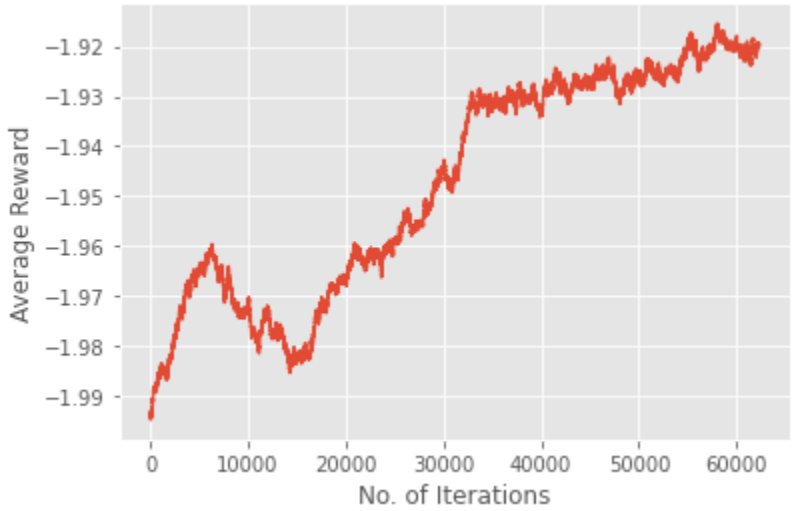
Caring coefficient = 0.1



Caring coefficient = 0.5

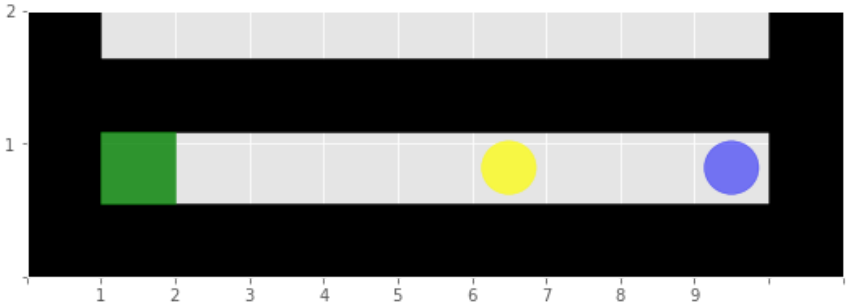


Caring coefficient = 1.0

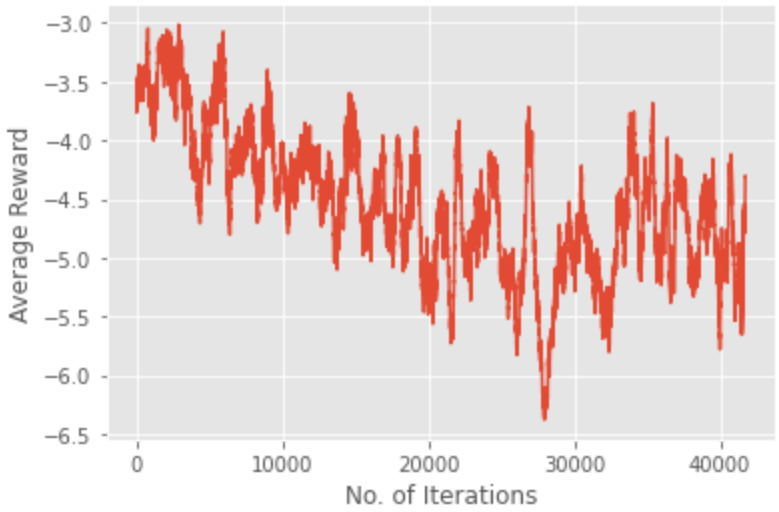


Results

We had experimented with the same experiments the authors of the paper had done with a simpler multi-agent environment where we are able to reproduce the results to some level of satisfaction. We choose a simple 2 system environment that trained the first agent's Q function and later used that for training the Q-function for the second agent. We experimented with various augmentation functions and caring parameter values as discussed in the paper. We later observed that the best results were obtained for the caring coefficient of 0.5 and augmentation function min(V(s),V(s0)). It is quite clear that the system maximises its reward when agents care for each other and hence caring has reduced negative side effects in the system. We could reproduce the results with complex environments such as a Minecraft-like environment.



Kitchen environment that we tried to use initially



But the performance was not up to expectations

We tried to show the effect of impact of considering others by using an alternative environment simulating a kitchen, where the goal of both agents is to search for ingredients. The agents are punished if the food is spoilt. However, it was not possible to reproduce this result due to technical difficulties. Instead we used the key-door environment for the same.