A comparison of learning performance in two-dimensional Q-learning by the difference of Q-values alignment

Kathy Thi Aung, Takayasu Fuchida
(Department of System Information Science, Graduate School of Science and Engineering, Kagoshima University, Kagoshima, Japan)
(Tel: 81-99-285-8453; Fax: 81-99-285-8464)
(kathythiaung@gmail.com, fuchida@ibe.kagoshima-u.ac.jp)

Abstract: Q-learning is a kind of reinforcement learning where the agent solves the given task based on rewards received from the environment. In this paper, we compared the performance of the Q-learning based on the reward examined by the difference of Q-values alignment in two-dimensional (2D) state space under various conditions such as angle of VQE rotation which is arranged like a lattice and angle of the agent's action rotation to correctly evaluate the optimal Q-values for state and action pairs, in order to deal with continuous-valued inputs. We apply the proposed method with an agent that learns to reach the reward area successfully during the reward-based learning process. A reward is given to the learning agent if the agent reaches the reward area during a process of trial.

Keywords: Q-learning, Q-value, VQE

1 Introduction

Learning algorithms based on evaluative feedback signal is generally referred to as RL algorithms.[1,4] In a RL paradigm, [4] a system called agent senses the environment and produces control actions. The environment responds to these control actions. Based on these responses a reward function will evaluate the control actions. The agent tries to optimize the control policy to maximize the total expected reward over a finite time-span. Learning may occur using the prediction error of expected rewards.

Reinforcement learning (RL) methods are a powerful and useful way to control agents such as an autonomous robot. [1] Q-learning [2] is the most widely used in RL method which deals with only discrete-valued inputs (states) and outputs (actions) to represent Q-function (action value function) that evaluates state/action pairs.

The aim of this work is to significantly improve the learning performance of Q-learning between the state space and the action space. Therefore, we implement and investigate here in order to clearly show the effectiveness of proposed learning method under various problems.

In this paper, we first briefly explain our agent model in a single-agent environment. Secondly, we define VQE with radius to decide the position. Next we describe each of the simulation methods, and then the performance of each strategy is examined by computer simulations on competitive situations of several strategies. We also show that the performance of each strategy strongly depends on the situation of our simulation methods and the simulation results are explained.

2. Q-Learning

In Q-learning, [2] the expected value of each action in each state is stored. In the other way, the Q-value is the expected value of each action in a certain state, which is the discounted sum of the rewards agent received for state and action pair. We can estimate and update the Q-value, which is denoted by $Q(s_t, a_t)$ according to the following equation by taking the one with the maximum Q-value (highest expected value) for the current state.

$$Q(s_t,a_t) = Q(s_t,a_t) + \alpha(r + \gamma \max_a Q(s_{t+1},a) - Q(s_t,a_t)) \quad \text{Eq. (1)}$$

When the agent is in state *t*, the agent observes the state s_t and executes the action a_t. The agent obtains the reward r_t, and senses a new state s_{t+1} by selecting an action a_t in state s_t. In this equation, α is the learning rate and γ is the discount rate, both are between 0 and 1.

In the standard Q-learning implementation, Q-values are stored in a table, is known as Q-table. It looks like a square lattice in two dimensions and one cell is required per combination of state and action. This implementation is not amenable to continuous state and action problems. [3] As the number of state and action variables increase, the size of the table used to store Q-values grows exponentially, is called *Curse of dimensionality*. On the other hand, as the number of

dimensions increases, the state space also increases exponentially and the learning speed decreases dramatically.

2.1. Behavioral Decision

The agent selects the next action which has the highest Q-value. An action which has a large Q-value is considered to be a good way to achieve the reward. However, selecting the highest Q-value continually decreases the opportunity to find a better way. Therefore, the agent sometimes selects the next action at random. This random selection is useful for exploring the state space and finding a new better way which has not been found yet.

2.2. Voronoi Q-value Element (VQE)

VQE is a point that has Q-value but we don't know where VQEs should be placed in the states space at the initial time. If we got many rewards, we would be able to evaluate the positions of VQEs for the optimal policy of state-action pair.

By using the VQE, we can place the Q-value arbitrarily in the state space and reduce the waste of space. Therefore, as the degree of freedom to select the position of VQE is so numerous, we have to decide carefully where the VQEs are placed in the state space.

3. Experimental Model

In these experiments, we tested with one agent and one reward area. The working environment of the agent is shown in Figure1, in where, it is intended to learn efficient action of an agent which is to reach the reward area in the state space.

The agent's action of maximum Q-value is selected while observing in a certain condition and the agent is a random action with a fixed probability. The agent observes the distance (r) and the angle (θ) toward the reward area.

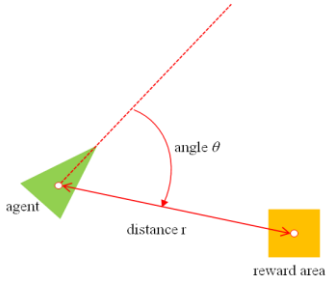


Fig.1. Working environment of an agent

There are two input variables: 1, the distance between an agent and the reward area; 2, an angle between the direction of agent' action and the reward area. The state space is constructed with these two input values. In this model, the agent is represented by a triangular arrowhead which indicates the direction in which the agent is moving. The rectangular area represents as the reward area.

3.1. Experimental Environment

An agent moves in a closed two-dimensional state space. Figure 2 shows an image of the agent that learns to reach the reward area. A reward is given to the learning agent if the agent reaches to the reward area, in the other states the reward is always zero.

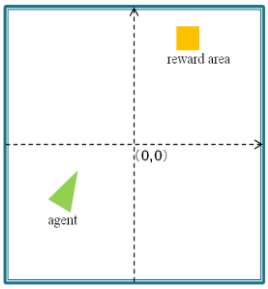


Fig.2. Agent Problem

In Figure 2, since the number of reward area is one, the agent observes two parameters. These two parameters construct the two-dimensional state space. The dimension of the state space goes up by two dimensions when one reward area is increased.

The possible actions of the agent are: 1, straight ahead; 2, turn left; 3, turn right. If the agent reaches the reward area, the position of the agent is randomly changed in this state space.

3.2. Experimental Parameters

width and height of space	-100,100
size of reward area	5
learning rate α	0.1
discount rate γ	0.9
random action rate	0.3
initial Q-value	$0 \le Q(s_t, a_t) \le 0.01$
travel distance of agent	2.0~5.0
Number of execution times	10 trials
One trial	20 episodes
One episode	100000 times

4. Simulation Methods and Results

We carried out three types of simulation experiment.

4.1. Square Lattice with random noise

The lattice below is a general 2-dimensional lattice on partition number 10, shown in figure 4.1.1. The idea of the lattice was proposed by VQEs and designates a 2-

dimension of 100 VQEs in lattice by symbol 10*10 grid environment. If we conduct this general 2-dimensional lattice with random noise (i.e., 0, 0.01, 0.02, 0.03, 0.04, 0.05 randomly arranged VQEs), we get the lattice like figure 4.1.2.

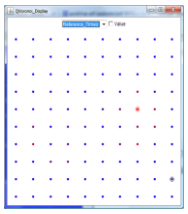


Fig.4.1.1 Original 2-D lattice

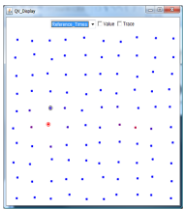


Fig.4.1.2. Random Lattice

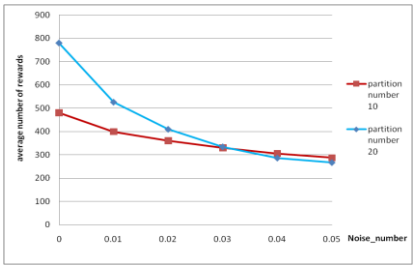


Fig.4.1.3. From regular to random VQE

As we can see from this above figure, the number of rewards is slowing down, thus its performance by this strategy depends on the arrangement of VQEs.

4.2. Lattice VQE rotation by Degrees

In this subsection, we rotated the above original 2-dimensional lattice by the angle of clockwise rotation on partition number 10 and 20. It turns for each degree of rotation by five degrees of angle intervals in the ranges from 0 to 90 degrees.

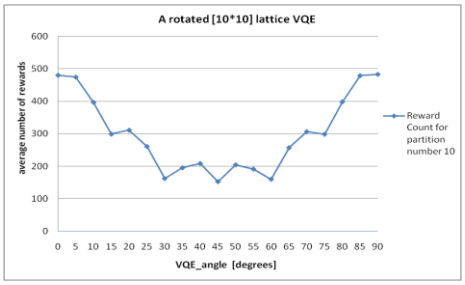


Fig.4.2.1. Reward count in a 10*10 grid environment

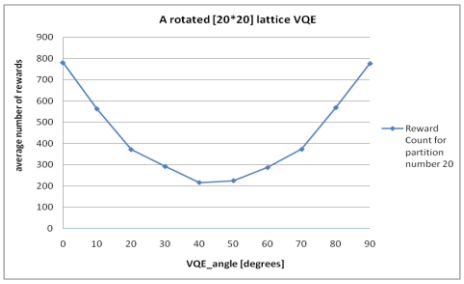


Fig.4.2.2. Reward count in a 20*20 grid environment

As in the experimental results shown in figure 4.2.1 for the 10*10 grid environment, the number of rewards clearly was slow, and a similar result also occur in a 20*20 grid environment because the reward propagation is delayed.

As the number of reward count decreases, we considered the following possible casual points.

- When the state changes, a state certainly transit to a different state if the action is different.
- But if lattice VQE rotates, also take in different action, grows the probability that the state of result go into the same state.
- Q-value decreases if the action is acting in the same area by the Q-learning method as shown in above equation (1).
- Since the angle of lattice VQE rotation for learning is enormous, we implemented the next strategy.

4.3. Rotation of Lattice VQE and agent's action

Here, we propose a new simulation experiment method in which the reward area is denoted by closed circle and the agent is denoted by open circle. The new strategy of learning for our target problem is defined as figure 4.3.1. Note that the agent takes actions with four directions: go up, go down, go left and go right.

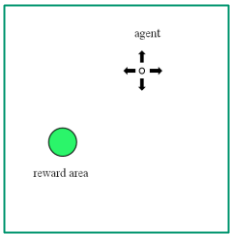


Fig.4.3.1. Structure of a new agent

In this case, we rotated the location of reward area and initial/default position of agent also. In each degree of our rotation, the action selection is simultaneously performed by the agent.

In our computer simulations, we used two elements rotations: act-angle and VQE-angle of rotations which measure between 0 degrees and 90 degrees by five degrees of angle intervals. (Figure 4.3.2)

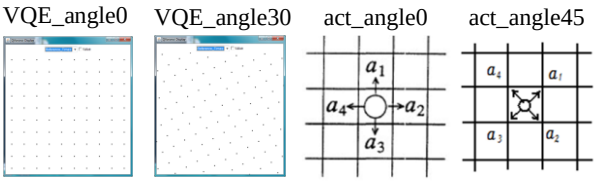


Fig.4.3.2. Rotation of VQE and agent's action

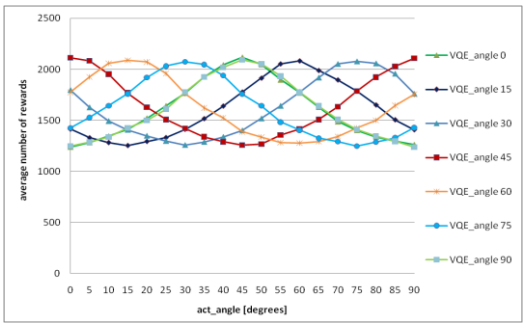


Fig.4.3.3. Reward count in a 10*10 grid environment

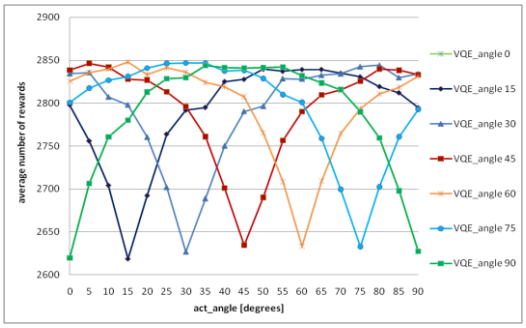
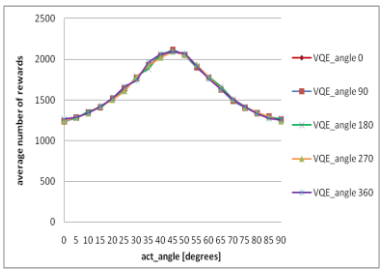


Fig.4.3.4. Reward count in a 20*20 grid environment

(a)



(b)

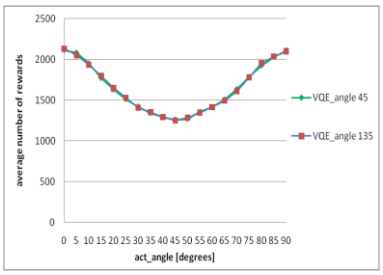


Fig.4.3.5. Angle of rotation for (a) 0,90,180,270,360 degrees and (b) 45,135 degrees

As Fig. 4.3.3 and Fig.4.3.4 reveals, when the turning angle of VQE are 0 degrees and 90 degrees, the results are entirely same at all, in fact there's no difference between them. In addition, Fig.4.3.5 also has totally same at the turning angle of 0, 90, 180, 270, 360 degrees and 45, 135 degrees. In this situation, we can see that the number of rewards increases due to the agent's optimum actions are four in this strategy.

Since Fig.4.3.6 generates the number of rewards that are put VQE randomly with noise on a number of (0,1,2,3,4,5) is relatively compared with in the Fig.4.1.3 above.

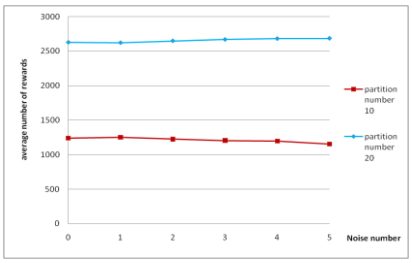


Fig.4.3.6. From regular to random VQE

5. Conclusion and Future Work

As a result, the learning performance was decrease by turning the angle of VQE only. When we turn act angle and VQE angle, it improves in performance. The difference between the number of actions 3 and 4 is according to the following reason.

The agent's action space and the state space is match or not. In the case of the agent's action space corresponds to the state space, the performance improves while shifting in the angle of rotation especially VQE angle and act angle is shifting by 45 degrees is greatly improved. Therefore, it is match of agent's action space and the state space that gives good performance.

In fact, it is different by expression of state input. It means when the number of action is 4, the agent observes by its position but in the case of action number 3, it observes the distance and the angle of agent relative to the reward area.

In the current research, we proposed learning method of Q-learning based on Voronoi Q-value element. In this paper, we examined the learning performance of various strategies in a different situation for our experimental environment. Although at the current stage, we only performed experiments in two dimensions, as future topics of research, we plan to a high-dimensional problem by generating an N-dimensional state space.

References

[1] R.S. Sutton and A.G. Barto, "*Reinforcement Learning: An Introduction"*, MIT Press, 1998.

[2] C.J.C.H. Watkins and P. Dayan, "*Q-learning*", *Machine learning*, Vol8, pp.279-292, 1992.

[3] Christopher J.C.H. Watkins. "*Learning from Delayed Rewards"*, PhD thesis, University of Cambridge, 1989.

[4] K.Kiguchi, T.Nanayakkara, K.Watanabe, T.Fukuda, *"Multi-Dimensional Reinforcement Learning Using a Vector Q-Net - Application to Mobile Robots"*, Internal Journal of Control, Automation and Systems, vol.1, no.1, pp.142-148, 2003.