The Seventeenth International Symposium on Artificial Life and Robotics 2012 (AROB 17th '12),
B-Con Plaza, Beppu, Oita, Japan, January 19-21, 2012

A proposition of adaptive state space partition in reinforcement learning with Voronoi Tessellation

Kathy Thi Aung[1], Takayasu Fuchida[2]

[1,2]Kagoshima University, Kagoshima 890-0065, Kohrimoto 1-21-40, Japan
[1,2]Graduate School of Science and Engineering, Department of System Information Science
(Tel: 81-99-285-3408, Fax: 81-99-285-8464)

[1]kathythiaung@gmail.com , [2]fuchida@ibe.kagoshima-u.ac.jp

Abstract: This paper presents a new adaptive segmentation of continuous state space based on vector quantization algorithm such as LBG (Linde-Buzo-Gray) for high-dimensional continuous state spaces. The objective of adaptive state space partitioning is to develop the efficiency of learning reward values with an accumulation of state transition vector (STV) in a single-agent environment. We constructed our single-agent model in continuous state and discrete actions spaces using Q-learning function. Moreover, the study of the resulting state space partition reveals in a Voronoi tessellation. In addition, the experimental results show that this proposed method can partition the continuous state space appropriately into Voronoi regions according to not only the number of actions, and achieve a good performance of reward based learning tasks compared with other approaches such as square partition lattice.

Keywords: Q-learning, LBG, new Vector quantization method, State space partitioning

1 Introduction

Reinforcement learning [1] is a type of active learning in which the autonomous agent interacts with its initially un-known environment, observes the results of its actions, and adapts its behavior appropriately. This type of learning has been widely studied as a learning method for determining the optimal actions. In particular, Q-learning [2] is a common reinforcement learning algorithm that is being investigated in a variety of applications.

In this paper, we develop an efficient algorithm for partitioning the state space in terms of computation. There are multiple ways of partitioning the state space such that Voronoi tessellation. Since we consider a single-agent RL problem in high-dimensional continuous state space and discrete actions, we built two experimental models A and B in different environments, and conduct two experiments to test the efficiency of reward learning. Our proposed method is based on the use of LBG for experimental model B, therefore, we use adaptive vector quantization method and the quality of a partitioning algorithm could also be estimated according to the number of non-overlapping regions of the partitioned state space. However, in order to increase the performance of learning efficiency, it is essential to achieve a good reward on the action space.

The following is a detailed description of our approach. First, a review of reinforcement learning and basic algorithm of Q-learning is presented in Section 2. And then, the adaptive partitioning of the state space and performance of the algorithm is described in detail in Section 3. The characteristics of proposed algorithm using vector quantization method and the efficiency of reward learning are investigated in Section 4. Finally, a conclusion is given in Section 5.

2 Reinforcement learning framework

In a typical reinforcement learning model, a learning agent is connected to achieve a reward or to reach a goal through interactions with its environment. In each time step, an autonomous agent observes the environmental state and makes a decision for a specific action, and selects an action in a current state according to a control policy. The control policy specifies each admissible action as a function of the observed state. At the next time step, a reward is generated according to the agent's environment after executing the action, and learns by that reward. It works much like human learning, and search for the optimal policy that maximizes the total expected reward. This is achieved by estimating the action value function, which is the total expected reward of taking action at a state.

2.1 Q-learning algorithm

Q-learning algorithm due to Watkins [2] is a policy for estimating the optimal state-action value function, denoted by $Q(s_t, a_t)$, as the value of Q. In general Q-learning, it discretizes the state and gives the Q-value that corresponds to each state and action pair. Q-value shows the value of action, and these Q-values are initialized to small random values and gradually change to the optimal values through learning process. In the state where the agent observed, the

action with the highest Q-value is considered as the optimal action, and learns by adding the random action to it. Q-value is updated by this following equation.

$$Q(s_t,a_t) = Q(s_t,a_t) + \alpha(r_t + \gamma \max_a Q(s_{t+1},a) - Q(s_t,a_t))$$

In this equation, $0 \le \gamma < 1$ is the discount factor, α is the learning factor, and r_t is the reward given in current state s_t, $Q(s_t,a_t)$ is the Q-value of action a in state s at time t, $\max_a Q(s_{t+1},a)$ is a maximum Q-value of state at time $t+1$.

3 Experiments A

We experiment the implementation of the partitioning algorithm on 2-dimensional one state input and N-actions variables alternatively known as continuous state space and discrete actions. In this model A, the position of the agent is expressed as a state input, and the reward area is placed at the center of the action space as shown in Figure 1, and action space and state space completely coincide with each other. The aim of the learning agent is to develop an optimal control policy, in other words, to achieve a good reward within a short period of time.

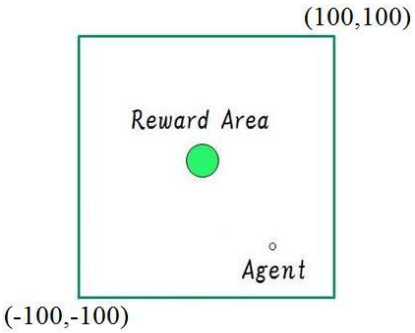

Fig.1. Experimental model A

We conducted a simple simulation experiment with a learning rate of 0.1, discount rate of 0.9 for Q-learning algorithm, and a random action rate was initialized to 0.3. A continuous action learning time is 20 thousand time for 1 episode and acts for 50 episodes, and perform 10 trials on each episode by changing the different initial seeds of random number. Furthermore, we examine the number of accumulated rewards by taking the average of each 10 trials for each episode.

3.1 Adaptive state space partitioning method (not use LBG method)

The idea of proposed partitioning algorithm is to group together states with similar action. In this way, the agent learns a suitable partitioning for a particular task and easily partition the state space using a clustering state transition vector (STV) or nearest-neighbor method. Our algorithm

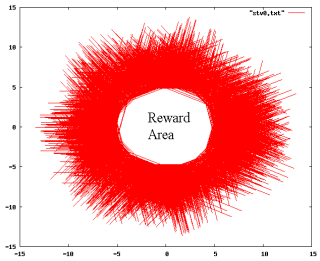

Fig.2.1. Collecting STV

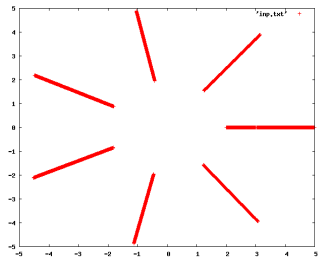

Fig.2.2. Seeking RV

for partitioning is performed on two steps. In the first step, we arrange the temporary points into lattice structure and collect the STV in regard to reward area that is a distance pointing from position of the agent to reward area that are described in Figure 2.1 when the agent enters the reward area. If the amount of rewards, in other words, the number of STV is accumulated 1000 or more, we group together STV by continuously taking the same actions, and seek those STV groups into representative vectors (RV) according to the number of actions are illustrated in Figure 2.2.

The accumulated STV with respect to a particular action is the sum of the total rewards received by taking the same actions. Then, the new Voronoi points are generated at the place of those representative vectors in relation to the reward area based on the number of actions, after all, we remove the temporary points in which the new Voronoi points are added. In essence, the continuous state space is partitioned into the number of N-actions subspaces as Voronoi regions shown on bottom of Figure 3.1.

In the second stage of partition formation, we collect the STV that come from temporary points to new Voronoi points of the quantized first-stage output, and group STV again which exceeds a threshold value by taking the same action. We do not take the STV that come from new Voronoi points to new Voronoi points. A threshold value is calculated dividing the number of accumulated STV 1000

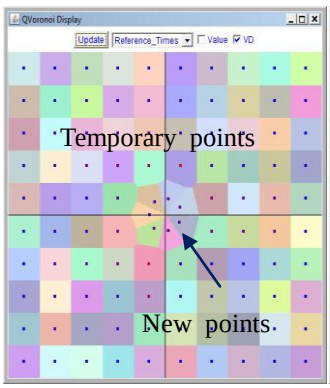

Fig.3.1. First stage of partition formation

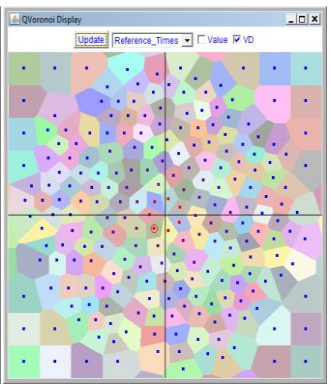

Fig.3.2. State space partition of 7-actions

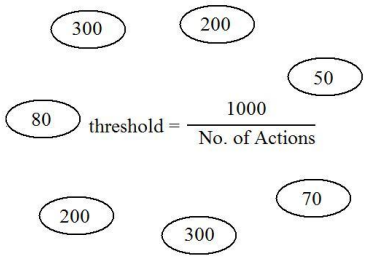

Fig.3.3. Calculation of threshold values for second stage formation

by the number of actions such that 7-actions, 6-actions. Figure 3.3 shows the formula quantization in order to make the group defined by threshold settings. Hence, the new Voronoi points are produced but it does not exactly the number of actions as new Voronoi points created in first stage formation (Fig.3.2). Moreover, we calculate the minimum distance between two new added Voronoi points to merge the points into one that are very close when we add the new Voronoi points of quantized second-stage process. If the distance between two new Voronoi points is less than the ratio of a minimum distance, we add the first entry Voronoi point. Furthermore, the reward numbers of state space partition are compared with square lattice partition.

3.2 Results

A summary of our experimental result is illustrated in Figure 3.4. It shows the comparison of two learning methods with the same number of Voronoi points. These two learning methods are partitioning of the state space using 7-actions on an experimental model A, and the regular arrangement of Voronoi points in a lattice structure are compared and discussed. As a result, adaptive partitioning the state space with N-actions has shown that

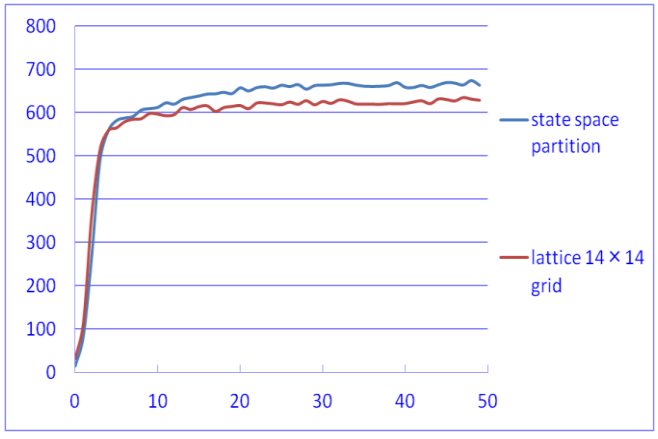

Fig.3.4. Comparison of learning performance using experimental model A for 7-actions
(X-axis: Episode number, Y-axis: Rewards number)

improved performance of reward learning, and it can accurately segment the state space as Voronoi tessellation with N-actions. However, the new points are not added if the ratio of the minimum distance is over 0.5 on the number of 3-actions and 4-actions.

4 LBG vector quantization algorithm

LBG (Linde-Buzo-Gray) is a typical technique of vector quantization algorithm. Modification of adaptive vector quantization method was introduced Enhanced LBG (Patane & Russo, 2001) [3], Adaptive incremental LBG (Shen & Hasegawa, 2006) [4]. Vector quantization method assumes that a set of input points is organized into a number of groups having approximately the same number of points closest to them. Each group is represented by its centroid point, as quantum vector (QV) idyllically known as representative vector. However, since the quantum vectors are not represented very well, we implemented the improved vector quantization algorithm as follow:

1. Collect the input vectors and tentatively place the QVs in random position.
2. Move the QVs in random directions by putting noise at the position of QVs.
3. Then, tentatively group the input vectors that are very close from the QVs, and move the QVs to the center position of input vectors.

We then repeated the above process until the number of QVs is -1, and reduce the amount of motion. Figure 4 describes an example of modified group vector quantization. Since the number of QV in clustering problem is not known a priori, we determines the number of QV by the index of their closest centroid which have smallest measurement error rates in group by changing the number of QV, and represents using that QV at a given time.

However, one group was represented with two QVs or two groups were represented with one QV. Therefore, we move the QVs of smallest measurement error to the

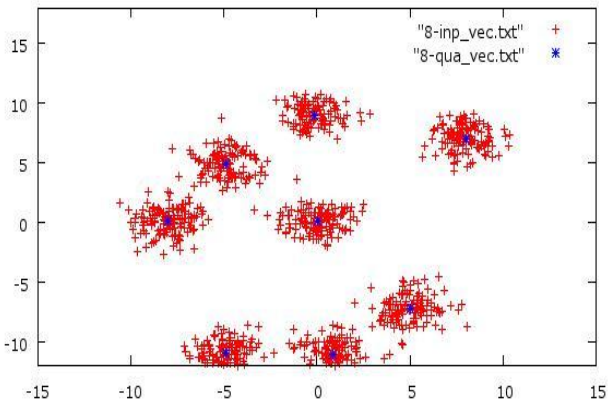

Fig.4. Input random vector quantization

position of the largest measurement error if the standard deviation is large. A standard deviation indicates that data are widely scatter condition of one group, and a small standard deviation is being able to represent as a representative vector, also known as quantum vector surely.

4.1 Experiments B

We test our approach by applying it in a continuous state space as shown in Figure 5. The agent learns to achieve optimal action, it means to arrive at a feeding area or reward area, and test its performance. If an agent reaches the goal, it receives a reward of +1, otherwise, if an agent collides into a border or wall, it was penalized by -1. For every reward, the agent is bounced to its new random position. The agent has 3 actions of straight forward, right rotation and left rotation, and observes the distance to a reward area and the angle between the direction of the agent forward movement and reward area, and learns by considering them as state inputs.

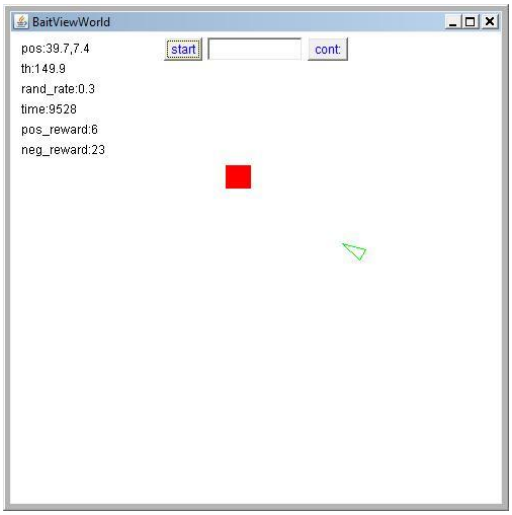

Fig.5. Experimental model B

Since the number of reward area is 1, these two state input values construct the 2-dimensional state space. The dimension of the state space goes up by two dimensions when one reward area is increased. We conducted experiments with one hundred thousand action learning times as 1 episode to 100 episodes and tested on 10 trials, and execute experiment as above.

4.2 Adaptive state space partitioning with LBG vector quantization algorithm

In this section, we do the same process of partitioning the state space that described in section 3.1 above. However, since the state input values and environments are different, the continuous state space is not partitioned into a number of N-actions subspaces. Further, we group together STV by using the LBG type of vector quantization algorithm and extract the representative vectors. We all take the STV (i.e, do not calculate a threshold value) for the second stage of partition formation.

4.3 Experimental results

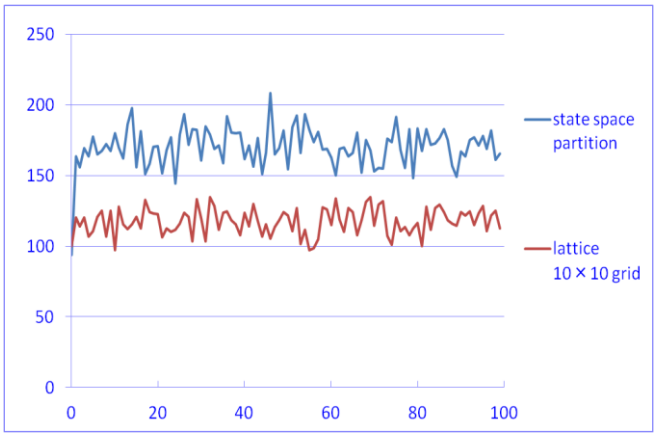

Fig.6. Comparison of learning performance using model B for 3-actions
(X-axis: Episode number, Y-axis: Rewards number)

The experimental results of Figure 6 show that our proposed approach raises the efficiency of the reward learning than the regular arrangement of lattice points.

5 Conclusions

This paper presented a new LBG vector quantization algorithm for partitioning the state space to Voronoi regions and adaptive continuous state space partition method. In our experiments, a simple experiment model by an agent was used in different environments with different state input values, and the effectiveness and efficiency of reward learning for two algorithms has been checked. When looking in terms of the reward, we see that our technique performs almost as good as square partition lattice and much better than its performance.

REFERENCES

1. Sutton RS, Barto AG (1998), Reinforcement learning an introduction, MIT Press, Cambridge.
2. Watkins CJCH, Dayan P. Technical notes: Q-learning. Machine Learning 1992;8:279-292.
3. G.Patane and M.Russo, The enhanced LBG algorithm, In proceedings of Neural Networks, 2001, pp.1219-1237.
4. F.Shen, O.Hasegawa, An adaptive incremental LBG for vector quantization, Neural Networks 19 (2006) 694-704.