Adaptive Exploration Strategies for Reinforcement Learning

Kao-Shing Hwang, Chih-Wen Li, Wei-Cheng Jiang

***Abstract*—Reinforcement learning through an agent to learn policy use trial and error method to achieve the goal, but when we want to apply it in a real environment, how to dividing state space becomes difficult to decide, another problem in reinforcement learning, agent takes an action in the learning process according to the policy, we will encounter how to balance exploitation and exploration, to explore a new areas in order to gain experience, or to get the maximum reward on existing knowledge. To solve problems, we proposed the decision tree-based adaptive state space segmentation algorithm and then use decreasing Tabu search and adaptive exploration strategies to solve the problem of exploitation and exploration on this method. Decreasing Tabu search will put the action into the Tabu list, after agent take an action. If the Tabu list is full, release the action, but the size of Tabu list will decreasing according to the number of successful reaching goals. Adaptive exploration strategy is based on information entropy, not tuning exploration rate by manually. Finally, a maze environment simulation is used to validate the proposed method, further to decrease the learning time.**

***Keywords*- Reinforcement learning, Exploration, Exploitation, decision tree**

I. INTRODUCTION

Reinforcement learning [1] [2] is not only limited to the discrete environment, but also can be used in a real continuous environment. State partition is a very important issue, how the continuous state space discretization, the easiest way is to divide the state space uniformly, but had a rough cutting performance will deteriorate, conversely, have fine dividing will generate a lot of state . To solve this problem, a better approach to dividing the state space.

Another very important issue in Reinforcement learning, that is the trade-off between exploitation and exploration [1] [7]. By takes an action according to policy that you want to explore the unknown state space as possible and get more information, but can Reinforcement learning in the number of visits state or state action would be huge, so how in a learning mode adopt effective strategies so that all states and actions can get enough information. How to decide when to explore to obtain new information or when to exploit the existing information in order to obtain maximum reward, which will greatly affect the efficiency of learning time. The method is currently used more often, such as, ε-greedy [1], softmax [4-6,9] and so on.

*Corresponding authors:

Kao-Shing Hwang, Chih-Wen Li, Wei-Cheng Jiang are with Electrical Engineering, National Sun Yat-sen University, Taiwan, Email: hwang@ccu.edu.tw; *kevinl117@msn.com; enjoysea0605@gmail.com*

II. BACKGROUND

A. MDP

The Markov Decision Process (MDP) is a mathematical framework. In a stochastic environment, if the conditional probability distributions of the future states depends only on the current state, then we said it has Markov property. We define MDP as $\langle S, A, P_a(s,s'), R(s), \gamma \rangle$, where S is the state space, A is the action space, $P_a(s,s')$ is the transition probability from the current state s to the next state s' when taking an action a, $R(s)$ is the reward function, our aim of MDP is to maximize the discounting accumulative reward$\sum_{t=0}^{\infty} \gamma^t R(s^t)$, where γ is discounting rate.

B. Reinforcement learning

Q-Learning is one of Reinforcement Learning proposed by Watkins [3], is used to meet the Markov property of the environment. Following is a brief Q-Learning, agent takes action according to policy in current state s, agent transfer from state s to the state s', get an immediate reward value r, agents updated Q value according to the state s and the immediate reward value r, the next time an agent to perform actions based on previous experience to select the best policy. Update formula is (1)

$$Q(s,a) = Q(s,a) + \alpha\left(r + \gamma max_{a'} Q(s',a') - Q(s,a)\right) \quad (1)$$

Where α (0 <α <1) is the learning rate, γ(0 <γ <1) is discounting rate. When the agent through interaction with the environment after several times, Q value will tend to optimize the value of Q^*. In this case the agent may be looking for the maximum value of Q to get the best policy.

III. THE TRADE-OFF BETWEEN EXPLORATION AND EXPLOITATION

A. Decreasing Tabu Search

In this paper ,we use Tabu list to store state - action pairs assume agents in state $\boldsymbol{s_t}$ and takes action a, we will check state - action pair $\boldsymbol{s_t}, a$ is in Tabu list or not when epoch finish , if not then store to Tabu list. When the Tabu list is full, release the first state-action pair. The ε-greedy and reduce Tabu search [10] for the combine to produce a new strategy to select actions. When a random variable is less than ε, randomly selected action, otherwise they will according to the maximum Q-value to select the action, if this state-action pair of action is stored in Tabu list, state-action value is temporarily set to negative infinity. This action in Tabu list will not be selected. The size of Tabu list will be a problem. Our approach is based on the number of successes reaches the target to reduce the Tabu list.

B. Adaptive exploration strategy

Here we proposed an adaptive exploration strategy, based on entropy to adjust the value of exploration rate ε in the learning process, let ε is not a fixed value. The beginning of the learning process that we hope agents more to explore, which is an agent not to take focus only on several actions ,to enhance the rate explore, on the contrary, when the agent is sufficient to obtain enough information, we want to reduce exploration but to enhance exploitation, which is the | ΔQ (s, a) | is small. We will store the exploration rate in each state, that is, each state will have its own exploration rate. In the action space A = $a_1, a_2, \cdots, a_N$, where N is the total number of actions that can be taken, we define the probability of takes action a_i in the state s as follows

$$P(s, a_i) = N(s, a_i) / \sum_{a_i \in A} N(s, a_i) \quad (2)$$

Where $N(s, a_i)$ are the number of takes action a_i in state s, if $\sum_{a_i \in A} N(s, a_i)$=0 then $P(s, a_i) \equiv 0$.we define the entropy in the state s ,$H(s)$ as follows

$$H(s) = -\sum_{a_i \in A} P(s, a_i) \log P(s, a_i) \quad (3)$$

where $1 \le i \le N$ and $0 \log 0 \equiv 0$ .we normalization of the entropy $H(s)$ is defined as

$$\overline{H(s)} = H(s) / \log |A| \quad (4)$$

We take entropy into consideration, the value of ε can be used as the quantity of $H(s)$,

$$\varepsilon = \begin{cases} \left(1 - \overline{H(s)}\right)\left(1 - exp\left(\frac{-|\Delta Q(s,a)|}{\sum_{a_i \in A} N(s,a_i)}\right)\right), if \left(\frac{|\Delta Q(s,a)|}{(\sum_{a_i \in A} N(s,a_i) * C_n)}\right) < C_h \\ \left(1 - \overline{H(s)}\right), otherwise \end{cases} \quad (5)$$

Where C_h is a threshold of exploration rate convergence, $|\Delta Q(s,a)| = \alpha\left(r + \gamma max_{a'} Q(s', a') - Q(s, a)\right)$,is derived from Q-learning, to check Q-value is converge or not, then we can view the total number of all actions takes in state s as a temperature, when it increase enough means the exploration rate in state s need to converge. C_n is a small constant, it is makes the scale between$|\Delta Q(s,a)|$ and the total number of all actions takes in state s are similar. $H(s)$ is used to measure the degree of confusion of action space in state s, if $H(s)$ of the value of small, agent focus on select part of the action, then need to explore, so we define $\varepsilon = \left(1 - \overline{H(s)}\right)$ to do more exploration, furthermore, to make sure exploration rate convergence, the factor $\left(1 - exp\left(\frac{-|\Delta Q(s,a)|}{\sum_{a_i \in A} N(s,a_i)}\right)\right)$,when $|\Delta Q(s,a)|$is smaller · $\sum_{a_i \in A} N(s, a_i)$ is increasing · this part will be a very small value. The agent will turn to exploitation.

IV. ADAPTIVE STATE PARTITION METHOD

A. Semi-Markov Decision Process

Semi-Markov Decision Process, MDP is very similar to the Markov decision-making process (MDP), in addition to sojourn time and reward functions [8] are different. Semi-Markov Decision Process can be defined as $\langle S, A, T_a(s,s'), P_a(s,s'), R_a(s,\tau)\rangle$, where S is the state space, A space is the action space, $T_a(s,s')$ is a non-negative random variable, the transition time from state s to the next state s', $P_a(s,s')$ the transfer probability of the current state s takes an action a transfer to next state s', $R_a(s,\tau)$ is defined as the discounting accumulated reward value. It written as formula (6)

$$R_a(s,\tau) = r_t + \gamma r_{t+1} + \cdots + \gamma^{\tau-1} r_{t+\tau-1} \quad (6)$$

When the agent enters a state representative to leave the state after a period of time, we called a epoch When the agent enters a state, will first meets decision point, we need to start this epoch of the select an action to takes, and then continue this action from the observation point observation obtained information until the end of this epoch so far. In a epoch, the agents will be transferred from the observation point to another observation point, to takes this action many times, but all belong to the same state and corresponds to the decision tree leaf node. Therefore, each epoch sojourn time is different. sensory input vectors received at time t, we will as he said s_t starting a period of time t after τ expressed as $s_t, s_{t+1}, \cdots s_{t+\tau-1}$, in the state s_{t+k} can be expressed as a value function (7).

$$V(s_{t+k}) = r_t + \gamma r_{t+1} + \cdots + \gamma^{\tau-k-1} r_{t+\tau-1} + \gamma^{\tau-k} \bar{V}(s'), \quad k = 0, \cdots, \tau - 1 \quad (7)$$

Where s' is the next state for the epoch finish, $\bar{V}(s')$ is the estimated state value of state s'. Since a state will get a lot of sensory input vectors, so our estimate is defined as the state of the input vector state average estimate. It written as

$$\bar{V}(s_t) = [V(s_t) + V(s_{t+1}) + \cdots + V(s_{t+\tau-1})]/\tau \quad (8)$$

B. Decision Tree

Adaptive State Partition method purpose is to build a decision tree [10], which is based on localized density to make discrete state space. In the beginning, only the root node of the decision tree corresponding to the entire state space, after several times of iteration, the root node is split into two sub-nodes, which corresponds to two separate subspaces. When the training process continues, the decision tree will continue to grow, while the state space will be partition to more subtle subspace.

In training iteration, the perception of the agent received input vector $\boldsymbol{s}_t$, and find the corresponding leaf node, expressed as s_t. Since each leaf node are recorded in all of the state-action values, we can according to the state-action values to select action, after takes an action, the agent will observe next perceived input vector s_{t+1} and the resulting immediate reward value r_{t+1}, then continue to take the same action until other nodes or agents to enter this epoch finish. We use formula (9) and (10) update state-action value .It is calculated as follows:

$$Q(s_{t+k}, a) = r_{t+k} + \gamma r_{t+k+1} + \cdots + \gamma^{\tau-k-1} r_{t+\tau-1} + \gamma^{\tau-k} \bar{V}(s'), k = 0, \cdots, \tau - 1 \quad (9)$$

And estimated state- action value of the node s_t is updated as follows:

$$\bar{Q}_{t+\tau}(s_t,a) = (1-\alpha)\bar{Q}_{t+\tau}(s_t,a) + \alpha[Q(\boldsymbol{s}_t,a) + Q(\boldsymbol{s}_{t+1},a) + \cdots + Q(\boldsymbol{s}_{t+\tau-1},a)]/\tau \quad (10)$$

We define state-error pair,(s_{t+k},δ_k), is a sensory input vector estimation error ,where the estimation error, $\delta_k = Q(s_{t+k},a) - \bar{Q}_t(s_t,a)$. Before we update estimated state-action value of s_t that will be calculated for all sensory input estimation error vector and stored state-error pair into the leaf nodes.

After updating the state-action value, depend on the presence of the leaf node state-error pair to determine the leaf nodes partition or not, if samples of the state-error pair N_a is less than a threshold N_{th} [10], which is a prior set of constants representing the state-action value will not converge within the range.

We updated state-action value by the estimation error, that is when the error is positive, estimated value will increase, and vice versa. Thus, the mean error of estimation should be convergence in a value close to zero. If the mean of the absolute value $|\mu_e|$ is larger than a threshold valueμ_{th}, represents still learning, the leaf node do not have to split. When the mean of the absolute value$|\mu_e|$ is less than the threshold μ_{th}, than use variance σ_e^2 to determine the quality of estimates. The smaller of the error variance is similar and represent the majority of small, on the contrary, the greater of the variation in the number of representatives of most of the errors are different, and big. If the variance is less than the threshold value σ_{th}^2, on behalf of the state- action value can be a good estimate and not a leaf node split, on the contrary, the representative of the range of variance is too much leaf node has no way a good estimate. Based on the above statement, the leaf nodes meet the following three conditions at the same time need to split:

1. N_a is smaller than a threshold N_{th}
2. $|\mu_e|$ is smaller thanμ_{th}
3. σ_e^2 is larger than σ_{th}^2

When we decide to split the leaf node, the next question to be considered is which dimension should be split ,we using T statistic to select spilt dimensions. We can separated state- error pair record in the leaf node into two groups, the estimated error is positive group, expressed as g_1 and the estimated error is negative into another group, expressed as g_2. Each group, the same dimensions in the resulting sensory input vectors as a set of statistical sample, written $X_{j,i}\sim(\mu_{j,i},\sigma_{j,i}^2)$, wherein $\mu_{j,i}$ and $\sigma_{j,i}^2$, respectively, the mean and variance of group i , g_j dimension. If $X_{1,i}\sim(\mu_{1,i},\sigma_{1,i}^2)$ and $X_{2,i}\sim(\mu_{2,i},\sigma_{2,i}^2)$ has similar mean and variance smaller in dimension i on behalf of the state variables for the estimation error of. We take T statistics used in the test $X_{1,i}\sim(\mu_{1,i},\sigma_{1,i}^2)$ and $X_{2,i}\sim(\mu_{2,i},\sigma_{2,i}^2)$ the degree of similarity, T statistics in dimension i, t value calculated by the formula (12).

$$t_i = range_i \frac{|\mu_{1,i}-\mu_{2,i}|}{\sqrt{\sigma_{1,i}^2/n_1 + \sigma_{2,i}^2/n_2}} \quad (12)$$

where $range_i$ is the range of leaf nodes in dimension i, $\mu_{j,i}$ is the mean of the sample, $\sigma_{j,i}^2$ is the variance of samples, n_j is the number of samples i of the dimension j group. if t value is small, the higher degree of similarity on behalf of two sets, on the contrary, the larger of t value, the lower the similarity of representatives of the two sets, we chose larger t value dimensions do split.

Finally, we decided the split points [11] after split or not and split dimensions. We use the statistics of the state -error to determining it. Since we have the mean and variance of the state variable, density function in two groups of dimension i is expressed as $\frac{1}{\sqrt{2\pi\sigma_{j,i}^2}}\exp(\frac{-(x-\mu_{j,i})^2}{2\sigma_{j,i}^2})$, where j = 1,2. Two density functions appropriate split point, the root can be obtained from the formula (13).

$$\frac{n_1}{\sqrt{2\pi\sigma_{1,i}^2}}\exp\left(\frac{-(x-\mu_{1,i})^2}{2\sigma_{1,i}^2}\right) = \frac{n_2}{\sqrt{2\pi\sigma_{2,i}^2}}\exp(\frac{-(x-\mu_{2,i})^2}{2\sigma_{2,i}^2}) \quad (13)$$

V. SIMULATION

A. Maze

In this Simulation, we construct a maze environment for mobile robot can find the target, the environment shown in Figure 3, Maze walls around 300x300. Above the square in the middle of the green type as a starting point and the target of the middle of the red circle, gray circular shape is an obstacle, the robot has four action to choose, up, down, left, right, respectively, every step from the set 5.0, select each action has added Gaussian noise, with an mean is 0, variance is 0.3. If the mobile robot to take action hit the wall or obstacle, the robot will stay put action and get feedback value of -1, when mobile robot to reach the target, mobile robot will get feedback the value of +5, otherwise mobile robot will get the reward of moves one step -0.01. When the mobile robot reaches the end or the number of steps to reach the 5,000 steps, moves back home to end this episode, each training 4000 episode. learning rate is set to 0.3, the discount rate set 0.99. In the adaptive strategy exploration, exploration rate of the initial value is set to 1. We will complete experimental parameters are listed in Table 1.

Fig. 3. The maze of the maze simulation

TABLE I. THE PARAMETERS OF THE MAZE SIMULATION

Learning rate	0.3	Threshold length	500
Discount rate	0.99	Threshold mean	0.5
Exploring rate	0.015	Threshold variance	0.03
Cn	0.001	Ch	0.025

As results in Fig. 4, the curve can be identified as Adaptive Exploration Strategy Method. It shows the average steps for the mobile robot to arrive at the target of each trail in 30 rounds of trainings. The proposed method reaches stable until the 900th trial

Is shown in Figure 4.3, the mobile robot in each round 30 times the average in the number of steps to reach target. The two curves represent adaptive exploration strategies and the ε-greedy, which exploration rate value of ε-greedy 0.015 is set, adaptive exploration strategies value Ch is 0.001 and Cn is 0.025. In the figure we can see, the adaptive exploration strategies of the average number of steps and learning speed convergence better than ε-greedy.

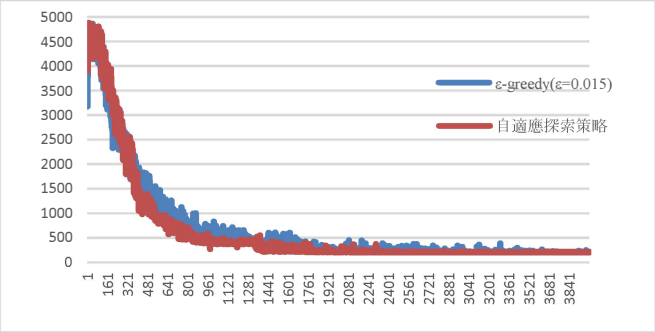

Fig. 4 Simulation results

VI. DISCUSSION

In this paper, we proposed a continuous state space partition method to solved when the reinforcement learning in continuous state space, hard to decide what the resolution is. Again, we entropy based on an adaptive strategy to explore, combine Decreasing Tabu search, to solve the problem of development and exploration of trade-offs. The simulation confirms the method that is learning performance improves.

REFERENCES

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

[2] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement Learning: A Survey," *Journal of Artificial Intelligence Research*, vol. 4, pp. 237-285, 1996.

[3] C. J. C. H. Watkins and P. Dayan, "Technical note: Q-learning," *Machine Learning*, vol. 8, no. 3, pp. 279-292, 1992.

[4] M. Tokic,"Adaptive ε-greedy Exploration in Reinforcement Learning based on Value Difference", *Advances in Artificial Intelligence*, vol.6359, pp.203-210, 2010.

[5] A. F. Atiya, A. G. Parlos, and L. Ingber, "A reinforcement learning method based on adaptive simulated annealing," in *Proceedings of the 46th IEEE International Midwest Symposium on Circuits and Systems, (MWSCAS '03)*, vol. 1, pp. 121-124, 2003.

[6] M. A. Wiering and H. V. Hasselt ,"Ensemble algorithms in reinforcement learning," *IEEE Trans. Syst., Man., Cybern. B*, vol.38,no. 4 ,pp.930 -936 ,2008

[7] M. Coggan,"Exploration and exploitation in reinforcement learning" , *in Proc. of the 4th Int. Conf. Comput. Intell. Multimedia Appl.*, pp.1 - 44, 2001.

[8] T. K. Das, A. Gosavi, S. Mahadevan, and N. Marchalleck, "Solving Semi-markov Decision Problems Using Average Reward Reinforcement Learning, "*Management Science*, vol. 45, pp. 560-574, 1999.

[9] R. Dearden, N. Friedman and D. Andre ,"Model based bayesian exploration" , in *Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence*,pp.150 -159, 1999.

[10] Yu-Jen Chen, "A Self-Organizing Decision Tree Approach to Policy Sharing of Multi-agent Systems", National Chung Cheng University, 2009

[11] W. Y. Loh, and Y. S. Shih, "Split Selection Methods for Classification Trees," *Statistica Sinica*, vol. 7, pp. 815-840, 1997.