CIMSA 2009 - International Conference on Computational Intelligence for Measurement Systems and Applications
Hong Kong, China
May 11-13, 2009

Incorporating Expert Knowledge in Q-Learning by means of Fuzzy Rules

Mojgan Pourhassan and Nasser Mozayani
Department of Computer Engineering
Iran University of Science and Technology
Tehran, Iran
m_pourhassan@comp.iust.ac.ir, mozayani@iust.ac.ir

Abstract— **Incorporating expert knowledge in reinforcement learning is an important issue, especially when a large state space is concerned. In this paper, we present a novel method for accelerating the setting of Q-values in the well-known Q-learning algorithm. Fuzzy rules indicating the state values will be used, and the knowledge will be transformed to the Q-table or Q-function in some first training experiences. There have already been methods to initialize the Q-values using fuzzy rules, but the rules were the kind of state-action rules and needed the expert to know about environment transitions on actions. In the method introduced in this paper, the expert should only apply some rules to estimate the state value while no appreciations about state transitions are required.**

The introduced method has been examined in a multiagent system which has the shepherding scenario. The obtaining results show that Q-learning requires much less iterations for getting good results if using the fuzzy rules estimating the state value.

Keywords: Q-learning; Expert knowledge; Fuzzy rules

I. INTRODUCTION

Reinforcement learning is learning how to map situations to actions in order to maximize the reward coming from the environment [1]. In so many cases the actions do not only affect the instant coming reward, but the rewards of the next steps as well. In reinforcement learning algorithms, a value function of the state (Q(s)) or a value function of the state-action (Q(s,a)) is estimated by the learning process. Q-learning, proposed in [2] is one of the most popular algorithms that use state-action values and has theoretical convergence property.

There have been many successful applications using reinforcement learning, but as the methods scale up to problems which have a large state space, learning becomes very slow. One effective method to speed up these algorithms is to leverage expert knowledge [3]. There are some algorithms proposed to use expert knowledge [3-7]. In [4] initialization of Q-values is studied, but as mentioned in [5] if the agent uses a more complex representation than a simple Q-table, it would make it difficult or impossible to initialize the Q-values to specific values. When dealing with large scale problems, the simple tabular representation can not be applied even if states and actions are discrete [6], thus, function approximators such as Fuzzy Rule-based Systems are used for representing the value function.

In [7] fuzzy rule-based Q-learning is proposed, which was extended and used later in many applications [6]. In [8] the performance and the convergence of fuzzy rule-based Q-learning and the conventional Q-learning is compared in a discrete environment, and the results show that although fuzzy rule-based Q-learning is much faster than the conventional Q-learning, it is not stable and leads to poor performance if its parameters are not properly specified. It is also proposed to initialize the Q-values by the same fuzzy rules as used in fuzzy rule-based Q-learning to have the advantages of both algorithms. What we propose in this paper is different in the kind of fuzzy rules we use.

II. INCORPORATING EXPERT KNOWLEDGE IN Q-LEARNING BY MEANS OF FUZZY RULES

As mentioned in previous section, there exists a method called fuzzy rule-based Q-learning which can accept the expert knowledge in the form of fuzzy rules estimating each state-action value. In this method $Q(x,a_i)$ is calculated by the following fuzzy rules [8]:

$$R_{ij}: \text{If } x_1 \text{ is } A_{j1} \text{ and } ... \text{ and } x_K \text{ is } A_{jK} \text{ then } Q(x,a_i)= Q_{ij}, \quad i= 1,2,...,M;\ j=1,2,...,N, \tag{1}$$

where Rij is the label of rule, i is an index of actions, j is an index of rules, M is the number of possible actions, N is the number of fuzzy rules for each action and A_{j1} to A_{jk} are the antecedent fuzzy sets. These N fuzzy rules are used for calculating $Q(x,a_i)$. We should first calculate the compatibility grade of the current state x with each fuzzy rule Rij as $\mu_{ij}(x)=A_{j1}(x_1)\cdot A_{j2}(x_2)\cdot ...\cdot A_{jk}(x_k)$ where $A_{j1}(\cdot)$ to $A_{jk}(\cdot)$ are the membership functions of the antecedent fuzzy sets A_{j1} to A_{jk} respectively. Then $Q(x,a_i)$ for each action a_i is calculated as follows:

978-1-4244-3820-4/09/$25.00 ©2009 IEEE

$$Q(x, a_i)=\sum_{j=1}^{N}\mu_{ij}(x).Q_{ij} \Big/ \sum_{j=1}^{N}\mu_{ij}(x) \quad ,i=1,2,...,M \qquad (2)$$

The fuzzy rules we saw in (1) can accept the expert knowledge but introducing such rules requires the expert to know how the actions affect the environment, and if there are any other agents, what they would do as a reaction.

The fuzzy rules of the method proposed in this paper are only based on state; in other words, they estimate the value of each state regardless of what action would be selected. Introducing such rules is possible even if the expert knows very little about the environment; such as some milestones on the path to the goal.

If the value of the states is properly estimated, it would highly accelerate the learning process to use this estimation in the updating rule of Q-learning. In (3) and (4) you can find the updating rule of Q-learning where 's' represents the state, 'a' the action, 't' the time step, 'α' the learning rate, 'r' the reward, and 'γ' is the discounting factor [1].

$$Q(s_t,a_t) \leftarrow (1-\alpha)Q(s_t,a_t)+\alpha\{r_t+\gamma V_t(s_{t+1})\} \qquad (3)$$

$$V_t(s_{t+1})=\max_a Q(s_{t+1},a) \qquad (4)$$

As we would like to use the estimated state value, our updating rule can be:

$$\begin{aligned} Q(s_t,a_t) \leftarrow{}& (1-\alpha)Q(s_t,a_t)+ \\ & \alpha\{r_t+\gamma((1-\beta)V_t(s_{t+1})+\beta V_{FR}(s_{t+1}))\} \end{aligned} \qquad (5)$$

Where $V_{FR}(s_{t+1})$ is the value estimated for state s_{t+1} by means of the fuzzy rules and β is the rate of using this value instead of $V_t(s_{t+1})$ which was defined in (4). Like α, β is a real number between 0 and 1.

In the beginning of the learning process where the Q-values are uniformly or randomly initialized, the value obtained from (2), would not bring a suitable value to use in the update rule. Therefore, having a β close to 1 would seem to be rational. In other words, the expert knowledge will be injected to the Q-values by this β rate during some first iterations of the learning process. On the other hand, as we are eager to save the convergence property of conventional Q-learning, β should gradually decrease to 0 during the learning.

The block diagram of "Fig. 1" indicates the operations in the learning process and the agent's interactions with the environment.

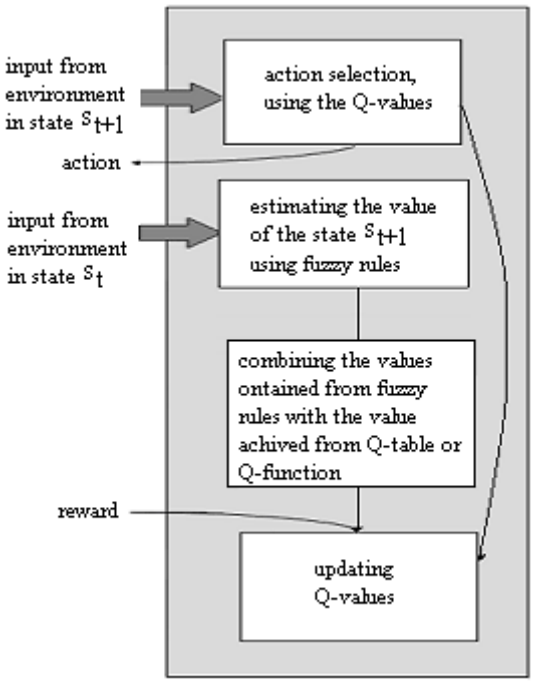

Figure 1. Operations in the learning process and the agent's interactions with the environment

III. IMPLEMENTATION

We have used the Agent Contest 2008[1] simulation platform to implement our idea. The Agent Contest 2008 scenario is the *shepherding problem*: There is a field and some cows spread in it. Each team agent should move in the field to guide the cows to the team's corral. The cows tend to remain in herds and get away from the agents [9]. Part of the field is shown in "Fig. 2" where the white dots are the cows, blue dots are the agents, the green objects are obstacles, and the blue square is the corral of the team.

We have implemented the Q-learning algorithm and the proposed method to incorporate the expert knowledge. In both cases we have used a learning rate of 0.3 and a discounting factor of 0.8. The team has 6 agents which use a common Q-table and share their experiences. Each agent receives a positive reward if the mean distance of the cows around it from the corral decreases in the next time step and a negative reward if the distance increases. In addition whenever a cow enters the corral, all agents near the corral receive 2 positive points as the reward.

[1] The computational intelligence group in the department of informatics of Clausthal University of Technology in Germany has been organizing a yearly agent contest since 2005. This competition is an attempt to stimulate research in the area of multi-agent programming. More information about the team and the contest they are providing can be found in the site of the university: http://www.in.tu-clausthal.de/de/abteilungen/cig/cigroot/research/projects/projectmassim/

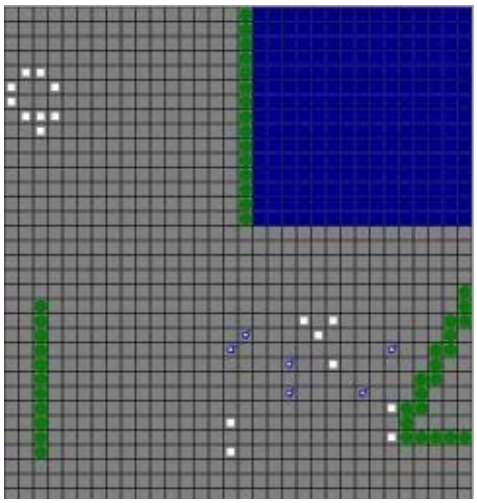

Figure 2. A part of the field

We have used a 4-field Q-table structure for both q-learning algorithm and the proposed method. The first field is referred to the corral direction from the agent's position which can be one of these: "North", "Northeast", "Northwest", "South", "Southeast", "Southwest", "East" and "West". The second field is the corral distance from the agent. This distance is generalized in a simple way. It is divided to 3 parts: "0 to 10", "10 to 20" and "more than 20". The third field is the cow direction if there exist any cow in the agent's view. It can be one of the directions mentioned above or "Nowhere". The forth field is the agent's action which can be going to one of the mentioned directions or doing nothing.

We used 100 runs of the game for training with an exploration rate of 0.9. After each 10 runs we stopped learning and tested the agent's behaviors for 10 other runs with 0.1 exploring. You can see the mean results of the test runs after each 10 runs for training in "Fig 3".

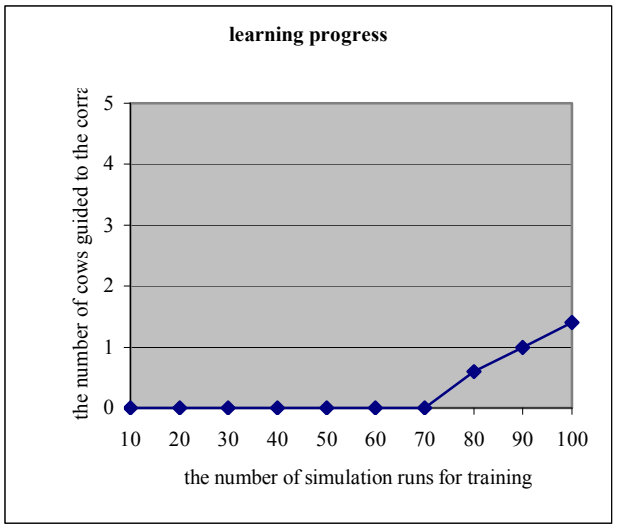

Figure 3. Results of the Q-learning algorithm, without expert knowledge

To implement the proposed method, we used a few simple fuzzy rules indicating:

1) If there are some cows around the agent, the closer the corral to the agent, the more value for that state.
2) If there are no cows around the agent, the farther the corral from the agent, the more value for that state. This rule will cause the agents to search for cows in a random direction.

As mentioned earlier, the rate of using estimated values, β, which we saw in (3), should gradually decrease to 0. f we do not decrease β and continue the training with β=1, it means that all the Q-table updates will be based on the value estimated by the fuzzy rules and no knowledge can be obtained during the learning process except what was proposed by the expert in the form of fuzzy rules.

To see the impact of decreasing β, we have tested the proposed method twice. First with a constant β=1, and then with a decreasing β. In the second test β was decreased 0.2 after each 10 iterations. The results are given in "Fig. 4" and "Fig. 5" respectively. As we can see in the diagram of "Fig. 4", the result is no more getting better after the expert knowledge in introduced to the Q-table in the first 30 or 40 runs. But we can see in "Fig. 5" that the learning is not limited to what the expert has proposed.

As we can get it from comparing the 2 diagrams of "Fig. 3" and "Fig. 5", we could accelerate the learning process by means of some simple fuzzy rules, estimating the value of each state. We could get the same result of more than 70 runs for training Q-learning, with only 10 runs using the estimated values.

RESULTS

As it can be seen in the diagrams of the previous section, we managed to incorporate the expert knowledge into the Q-learning process with a few simple fuzzy rules. It can be really helpful to use the expert knowledge especially when the state space is large or the reward's delay is too much.

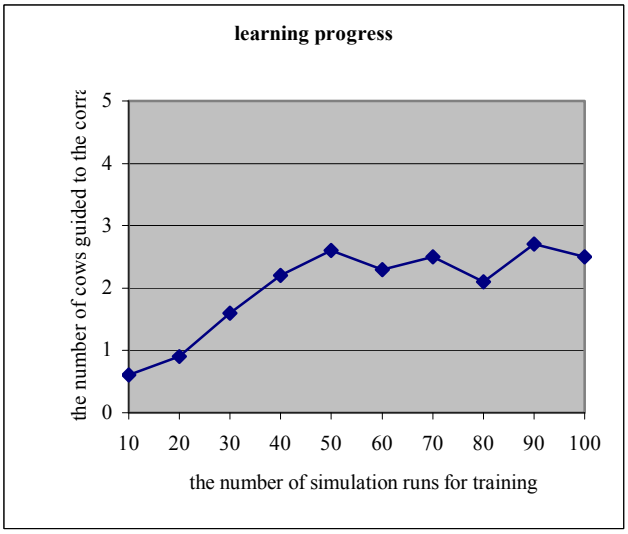

Figure 4. Results of the proposed method with constant β=1

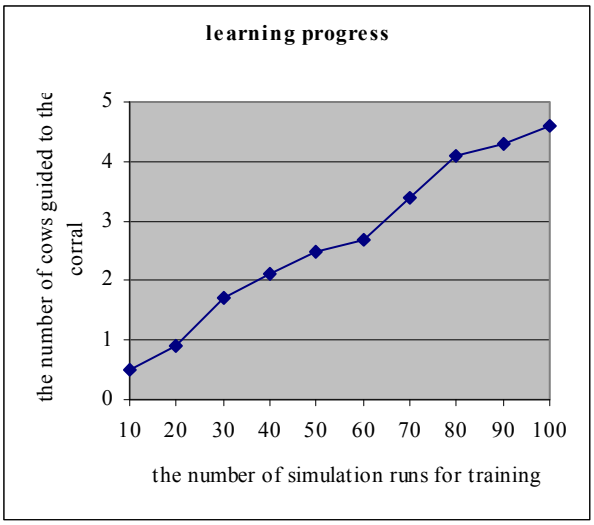

Figure 5. Results of the proposed method with constant β=1

For incorporating the expert knowledge into Q-learning, we mentioned in the second section that fuzzy rule based Q-learning was already used, but it has different kind of rules from the rules used in out method. In other words, in that algorithm, the fuzzy rules estimate a value based on state and action, requiring the expert to know about the environment transitions, but our rules make estimates of the states only, and make it possible to define simple useful rules when the expert knows so little about the environment. For example, in the simulation we used for implementation, we can introduce fuzzy rules even if we don't know the cows' behavior (would they come toward us or would they get away from us), which is not possible in case of using fuzzy rule based Q-learning.

REFERENCES

[1] R. Sutton, A. Barto, Reinforcement Learning, MIT press. USA, 1998.

[2] C. J. C. H. Watkins, P. Dayan, "Q-Learning," Machine Learning 8, 1992, pp.279-292.

[3] M. Ahmadi, M. Taylor, P. Stone, IFSA: "Incremental Feature-Set Augmentation for Reinforcement Learning Tasks", The Sixth Intl. Joint Conf. on Autonomous Agents and Multi-Agent Systems,Honolulu, Hawaii, 2007, pp.1120-1127.

[4] G. Hailu, G. Sommer, "On amount and quality of bias in reinforcement learning". IEEE International Conference on Systems, Man and Cybernetics, Tokyo, Japan, 1999, pp.1491-1495.

[5] E. Wiewiora, G. Cottrel, Ch. Elkan, "Principled Methods for Advising Reinforcement Learning Agents", Proceedings of the Twentieth International Conference on Machine Learning, Washington DC, 2003.

[6] A. Ferdowsizadeh Naeeni, Advanced multi-agent fuzzy reinforcement learning , Master thesis, computer science department, Högskolan Dalarna , Sweden, 2004.

[7] P. Y.Glorennec, "Fuzzy Q-learning and Dynamical Fuzzy Q-Learning," Proc. of 3rd IEEE International Conference on Fuzzy Systems, USA, 1994, pp.474-479.

[8] C.-H. Oh, T. Nakashima, and H. Ishibuchi, "Initialization of Q-values by fuzzy rules for accelerating Q-learning," IEEE International Joint Conference, vol. 3, no. 4-9, pp. 2051-2056, 1998.

[9] T. Behrens, M. Dastani, J. Dix, P. Novak, "Multi-Agent Programming Contest, Contest Scenario Description", Clausthal University of Technology, viewed 23 Apr 2008, <http://www.in.tu-clausthal.de/fileadmin/homes/techreports/ifi0715dastani.pdf > , 2008