# Tuning Fuzzy Inference Systems by Q-Learning

Mohamed Boumehraz* , Khier Benmahammed**
*Laboratoire MSE, Département Electronique, Université de Biskra, boumehraz_m@yahoo.fr
** Département Electronique, Université de Setif

**Keywords:** reinforcement learning, fuzzy inference systems, Q-learning

**Abstract:** Fuzzy rules for control can be effectively tuned via reinforcement learning. Reinforcement learning is a weak learning method wich only requires information on the succes or failure of the control application. In this paper a reinforcement learning method is used to tune on line the conclusion part of fuzzy inference system rules. The fuzzy rules are tuned in order to maximize the return function . To illustrate its effectivness, the learning method is applied to the well known Cart-Pole balancing system problem. The results obtained show significant improvements of the speed of learning.

## 1. Introduction

Reinforcement learning (RL) refers to a class of learning tasks and algorithms in which the learning system learns an associative mapping by maximizing a scalar evaluation (reinforcement) of its performance from the environment [1,2,3]. Compared to supervised learning , RL is more difficult since it has to work with much less information. Fuzzy inference systems have been shown able to provide excellent control in a number of practical applications. However, the problem in fuzzy systems is how to define the appropriate fuzzy rules. Several approaches have been proposed to autamitically extract rules from data ; gradient descent[4] , fuzzy clustering, genetic algorithms [5,6] and reinforcement learning[7,8,9,10,11]. In this paper we use Q-learning to determine the appropriate conclusions for a Mamdani fuzzy inference system. We assume that the structure of the fuzzy system and the membership functions are specified *a priori*.

## 2. Reinforcement Learning

### 2.1 Reinforcement learning model

In reinforcement learning an agent learns to optimize an interaction with a dynamic environment through trial and error. The agent receives a scalar value or reward with every action it executes. The goal of the agent is to learn a strategy for selecting actions such that the expected sum of discounted rewards is maximized[1].

In the standard reinforcement learning model, an agent is connected to its environment via percetion and action, as depicted in figure 1. At any given time step t, the agent perceives the state $s_t$ , of the environment and selects an $a_t$. The environment responds by giving the agent scalar reinforcement signal, $r(s_t)$ and changing into state $s_{t+1.}$ The agent should choose actions that tend to increase the long run sum of values of the reinforcement signal. It can learn to do this overtime by systematic trial and error, guided by a wide variety of algorithms.

The agent goal is to find an optimal policy, $\pi : S \rightarrow A$, which maps states to actions, that maximize some long-run mesure of reinforcement. In the general case of the reinforcement learning problem, the agent's actions determine not only its immediate rewards, but also the next state of the environment. As a result, when taking actions, the agent has to take the future into account. The reinforcement learning can be summarized In the following steps.

```
Initialize the learning system
repeat
  1-With the system in state s, choose an action a according to an exploration policy and apply it to the system
  2- The system returns a reward r, and also yields next state'.
  3- Use the experience, (s,a,r,s') to update the learning system
  4 – s ← s'
until s is terminal
```

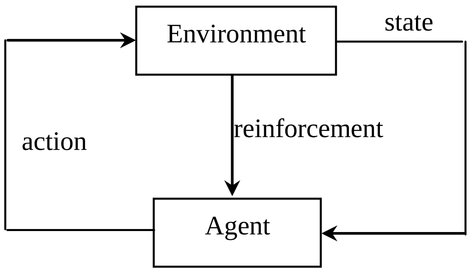

Figure 1. Reinforcement learning scheme

### 2.2 The return function

The agent's goal is to maximize the accumulated future rewards. The return function, or the return, R(t), is a long-term measure of rewards. We have to specify how the agent should take future into account in the decisions it makes about how to select an action now. There are three models that have been the subject of the majority of work in this area.

***The finite-horizon model***

In this case, the horizon corresponds to a finite number of steps in the future. It exists a terminal state and the sequence of actions between the initial state and the terminal one is called a period. The return is given by:

$$R(t) = r_t + r_{t+1} + \cdots + r_{t+K-1} \qquad (1)$$

where K is the number of steps before the terminal state.

***The discounted return (infinite-horizon model)***

In this case the longrun reward is taken into account, but rewards that are received in the future are geometrically discounted according to discount factor $\gamma$, $0 < \gamma < 1$ and the criteria becomes.

$$R(t) = \sum_{k=1}^{\infty} \gamma^k r_{t+k} \qquad (2)$$

***The average-reward model***

A third criteria, in which the agent is supposed to take actions that optimize its long-run average reward is also used :

$$R(t) = \lim_{n\to\infty} \frac{1}{n} \sum_{k=1}^{n} r_{t+k} \qquad (3)$$

**2.2 The state value function or value function**

The value function is a mapping from states to states values. The value function $V^{\pi}$ (s) of state s, associated with a given policy $\pi$(s) is defined as [1] :

$$V^{\pi}(s_t) \equiv E\left[\sum_{k=1}^{\infty} \gamma^k r_{t+k+1}\right] \qquad (4)$$

Where $s_t$ is the state at time t, $r_{t+k+1}$ is the reward received for performing action :

$$a_{t+k} = \pi(s_{t+k}) \qquad (5)$$

at time t+k, and $\gamma$ is the discount factor ( $0 < \gamma < 1$).

**2.3 Action-value function or Q-function**

The action-value function measures the expected return of executing action $a_t$ at state $s_t$, and then following the policy $\pi$ for selecting actions in subsequent states. The Q-function corresponding to policy $\pi$(s) is defined as [1]:

$$Q^{\pi}_{t+1}(s_t, a_t) \equiv r_{t+1} + \gamma Q^{\pi}_t(s_{t+1}, \pi(s_{t+1})) \qquad (6)$$

The advantage of using Q-function is that the agent is able to perform one-step lookahead search without knowing the one-step reward and dynamics functions.

The disavantage is that the domain of the Q-function increases from the domain of states *S* to the domain of state-action pairs (s,a).

**3- Reinforcement Learning Methods**

**3.1 Q-Learning**

It exists several approaches for reinforcement learning without models. Some are based on policy iteration, such as the Actor Critic Learning, and others on value iteration, such as Q-Learning or SARSA. The Q-Learning, proposed by Watkins [12], is perhaps the more popular of algorithms, by reason of its simplicity.

**One-step Q-Learning**

The first version of Q-Learning is based on the temporal differences of order 0, TD(0), while only considering the following step (one-step Q-Learning). The agent observes the present state, $s_t$, and executes an action, $a_t$, according to the evaluation of the return that it makes at this stage. It updates its evaluation of the value of the action while taking in account,

a) the immediate reinforcement, $r_{t+1}$, and
b) the estimated value of the new state, $V_t(s_{t+1})$, that is defined by:

$$V_t(s_{t+1}) = \max_{b\in A} Q_t(s_{t+1}, b) \qquad (7)$$

The update corresponds to the equation:

$$Q_{t+1}(s_t, a_t) = Q_t(s_t, a_t) + \alpha\{r_{t+1} + \gamma V_t(s_{t+1}) - Q_t(s_t, a_t)\} \qquad (8)$$

$\alpha$ is a learning rate such that $\alpha \to 0$ as $t \to \infty$.

In addition to its simplicity, Q-Learning presents several interesting characteristics.

- The evaluations of Q, the Q-values, are independent of the policy followed by the agent. This one can follow any policy, while continuing to construct correct evaluations of the value of actions.
- Q-values are exploitable a long time before the formal convergence that can be sometimes very slow.
- Lastly, there are proofs of convergence toward the optimal policy[12].

**4. Optimization of fuzzy inference systemes by Q-Learning**

Reinforcement learning has been used for optimization of fuzzy inference systems by two types of methods: Methods based on policy iteration, driving to Actor-Critic architectures [7,8], and the others based on value iteration, generalize Q-Learning[9,10,11], in [11] Glorennec uses Q-Learning for the optimization of a zero order Takagi-Sugeno FIS, with a constant conclusions. If the action space is continuous the conclusions are equally distributed between lower and upper bounds of the action.

In this paper, we consider a Madani FIS, and continuous state and action spaces. The FIS structure is fixed *a priori* by the user and the fuzzy sets for the inputs and output are supposed fixed. Our approach, consist in determining the optimal conclusions of the fuzzy inference system.

**4.1 Mamdani fuzzy inference system**

A Mamdani inference system is described by a set of fuzzy rules of the form [13]:

Rule i : if s is $\boldsymbol{A}^i$ then a is $\boldsymbol{B}^i$

Where s is the fuzzy system input, $\boldsymbol{A^i}$ is a fuzzy label for input in ith rule, a is the output of the fuzzy system and $\boldsymbol{B^i}$ is fuzzy label for the output in ith rule.

The problem is how to choose the appropriate rules in order to optimize system performance (in RL maximize the accumulated future rewards)[13]. In this paper we use Q-learning to optimize rule conclusions.

Several competing conclusions are associated to each rule, and a quality value is assigned to each conclusion. The conclusion with the high quality is used by the system to generate actions . The fuzzy rule becomes:

Rule i : if s is Ai then a is $\arg\max_{b\in B} Q(s,b)$

**4.2 Learning process**

At each rule, several conclusions are associated, and each conclusion has a Q-value:

The fuzzy rule is of the form:

Rule i : if s is $\boldsymbol{A}^i$ then a is $\boldsymbol{B_1}$ with $Q^i(s, \boldsymbol{B_1})$
or a is $\boldsymbol{B_2}$ with $Q^i(s, \boldsymbol{B_2})$
or a is $\boldsymbol{B_3}$ with $Q^i(s, \boldsymbol{B_3})$

or a is $\boldsymbol{B_m}$ with $Q^i(s, \boldsymbol{B_m})$

where $\boldsymbol{B_1}$, $\boldsymbol{B_2}$,….., $\boldsymbol{B_m}$ are the fuzzy sets of the outputs and $Q^j(s, \boldsymbol{B_i})$ is the Q-value of the conclusion a is $\boldsymbol{B_i}$ of the rule j.

During learning the Q-value of each conclusion is updated using Q-learning ( equation 8):

$$Q^i_{t+1}(s_t,B_j)=Q^i_t(s_t,B_j)+\alpha\mu_i(s_t)\left\{r_{t+1}+\gamma V_t(s_{t+1})-Q^i_t(s_t,B_j)\right\} \qquad (9)$$

Where $\mu_i(s_t)$ is the truth value of the ith rule and $B_j$ is the jth conclusion of the ith rule.

With the value of the new state given by:

$$V_t(s_{t+1})=\sum_{i=1}^{N}\frac{\mu_i\left(s_{t+1}\right)}{\sum_{j=1}^{N}\mu_j\left(s_{t+1}\right)}\mu_i\left(s_{t+1}\right)\max_{b\in B}Q^i_t\ (s_{t+1},b) \qquad (10)$$

if $s_{t+1}$ is a final state then :

$$V_t\ \left(s_{t+1}\right)=0 \qquad (11)$$

**5. Results**

The proposed method is applied to a classic problem; the pole balancing problem or inverted pendulum problem. In this problem a pole is hinged to a motor-driven cart which moves on rail tracks to its right or its left. The primary control task is to keep the pole vertically balanced.

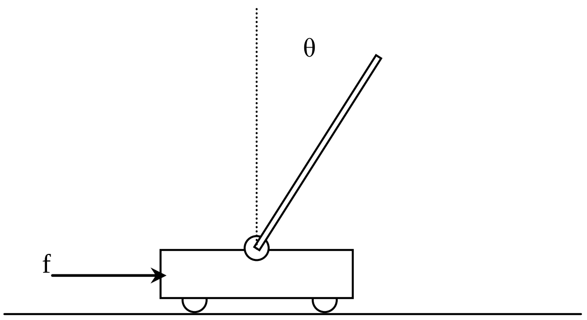

Figure 2. The Cart-Pole System

The dynamics of the cart-pole system are modeled by the following non linear differential equation [7,13]:

$$\ddot{\theta}=\frac{g\sin\theta+\cos\theta\left[\frac{-f-ml\dot{\theta}^2\sin\theta}{m_c+m}\right]-\frac{\mu_p\dot{\theta}}{m\,l}}{l\left[\frac{4}{3}-\frac{m\cos^2\theta}{m_c+m}\right]} \qquad (12)$$

where g is the gravity, $m_c$ is the mass of the cart, m is the mass of the pole, l is the half-pole length and $\mu_p$ is the coefficient of friction of pole on cart. The sample period is 20 ms.

We assume that a failure happen when $|\theta| > 45°$. Also, we assume that the equation of motion is not known to the controller and that only a vector describing the cart-pole system's state at each time step is known.

The inputs of the fuzzy controller are error e and error change $\Delta e$:

$$e(k)=\theta(k) \qquad (13)$$

$$\Delta e(k)=e(k)-e(k-1) \qquad (14)$$

The output is the force f and the Q-values of conclusions.

The fuzzy partitions of the inputs and output are described in figure 3.

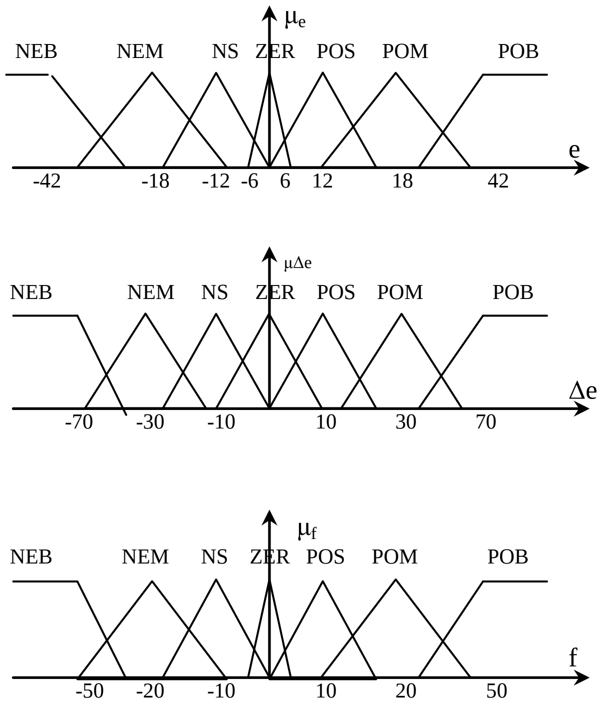

Figure 3. Membership functions

The rule base is choosen arbitrary and the Q-values of the conclusions are set initially to zero. We use center of area defuzzification and the min operator to implement the premise and implication.

A trial in our experiments refers to starting with the cart-pole system set to an initial state and ending with the appearance of a failure signal or successful control of the system for an extended period (1000 time steps or 20 seconds). The Q-learning was applied to tune fuzzy rule conclusions. The free constants were $\gamma=0.95$ and $\alpha$ set initially to 0.1 and decreases. Figure 4. shows the average return per trial performance of the controller during the learning process; the average return per trial and figures 5 and 6 show the response of the system , after learning, for initial angle equal to –50° and 28° respectively.

It is clear that the average return increases during learning until it reaches a sub-optumal value. The obtained fuzzy controller is able to stabilize the pole for angles inferior to 55°.

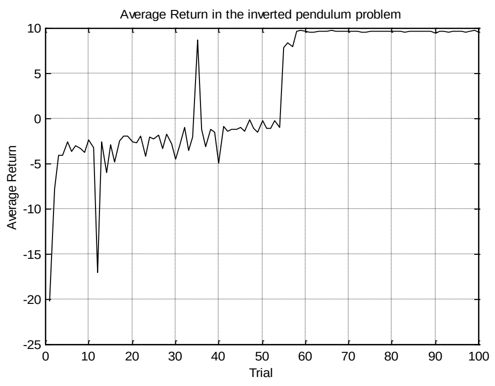

figure 4. The Average Return

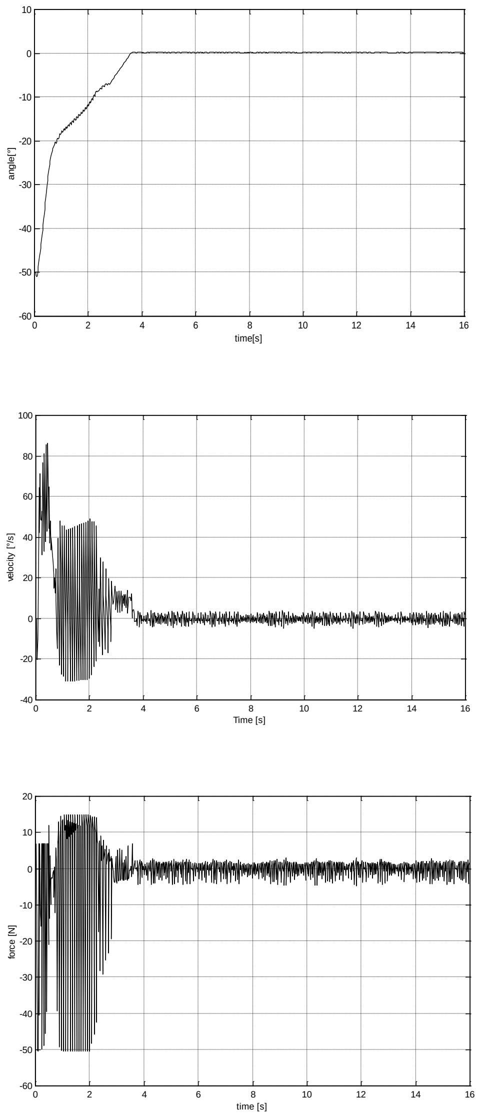

Figure 5. Angle, velocity and force for initial angle equal 50°

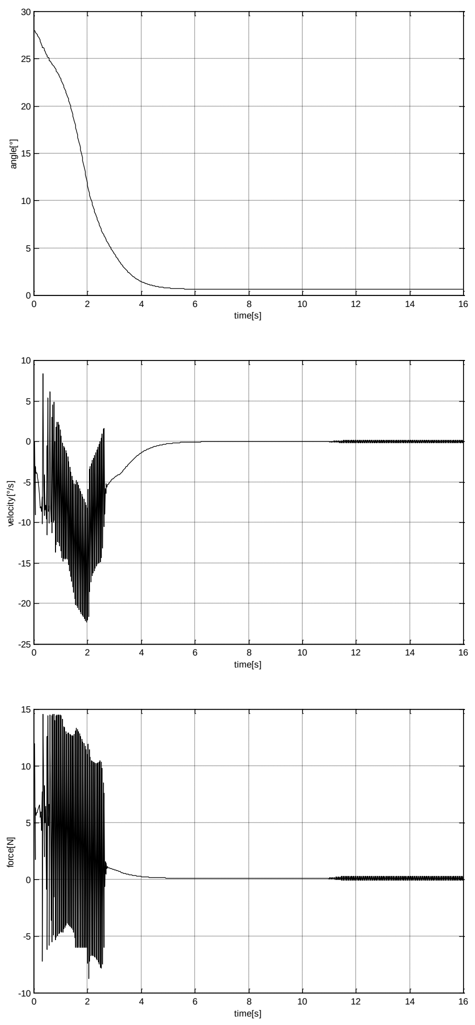

Figure 6. Angle, velocity and force for initial angle equal to 28°

## 5. Conclusions

In this work we have proposed a new method of optimizing fuzzy inference system based on Q-learning. This method was applied to cart-pole system. After learning , the controller is able to stabilize the pendulum. We assume that structure of the fuzzy system is fixed *a priori*. The optimization of membership function parameters and number of rules will improve the performance of the proposed method.

## References

[1] R. S. Sutton, A. G. Barto, Introduction to reinforcement learning, MIT Press/Bradford Books, Cambridge, MA, 1998.

[2] V. Gullapalli, Reinforcement learning and its appication to control, Ph. D. Thesis, University of Massachusetts, Amherst, MA, USA,1992.

[3] L. P. Kaelbling, M. L. Littman, A. W. Moore, Reinforcement learning: a survey, Journal of Journal Artificial Intelligence Research 4, 1996.

[4] J. R. Jang, Self-Learning Fuzzy Controllers Based on Temporal Back Propagation, IEEE Transactions on Neural Networks, Vol. 3 No. 5, September 1992.

[5] M. G. Cooper, J. J. Vidal, Genetic Design of Fuzzy Controller, Proceedings of Second Inernational Conference on Fuzzy Theory and Technology; Durham, NC, October, 1993.

[6] A. Bonarini, Evolutionary learning of fuzzy rules:competition and cooperation, in Fuzzy modeling : paradigms and practice, Kluwer Academic Publishers, Norwell, MA, 1995.

[7] H. R. Berenji P. Khedkar, Learning and Tuning Fuzzy Logic Controllers Through Reinforcement, IEEE Transactions on Neural Networks, Vol. 3 No. 5, September 1992.

[8] M. V. Buijtenen, G. Schram, R. Babuska, B. Verbruggen, Adaptive Fuzzy Control of Satellite Attitude by Reinforcement Learning, IEEE Transactions on Fuzzy Systems, Vol. 6, No. 2, May 1998.

[9] H. R. Berenji, Fuzzy Q-Learning: a new approach for fuzzy dynamic programming, Proceedings of IEEE international conference on Fuzzy Systems, Nj, 1994.

[10] P. Y. Glorennec, L. Jouffe, Fuzzy Q-Learning, Procedings of FUZZ-IEEE’97, Barcelona, Spain, July 1997.

[11] P. Y. Glorennec, Reinforcement Learning: an Overview, ESIT 2000, Aachen, Germany, 14-15 September 2000.

[12] C. Watkins Learning from Delayed Rewards, PhD. Thesis, University of Cambridge, England, 1989.

[13] K. Passino, S. Yurkovich , Fuzzy Control, Addison Wesley, California, 1998.