Fifth IFSA World Congress (1993), 1033-1036

FAM APPROACH TO DESIGN A FUZZY CONTROLLER

M. Lo Presti♥, R. Poluzzi♠, G.G. Rizzotto♠, A. Zanaboni♦

♠Corporate Advanced System Architectures
SGS-THOMSON Microelectronics

♥Fuzzy Logic Research Group
Co.Ri.M.Me.

♦Neural Networks Laboratory
Dipartimento di Scienze dell'Informazione
Universita' di Milano

Abstract
Most of the today realized fuzzy logic control applications has been designed using different heuristic approaches for synthesis and implemented with conventional programming languages on general purpose microcontrollers.
This paper aims to present a new methodology to design a fuzzy controller. The methodology is based on the Cell-to-Cell approach to extract the control law. A set of fuzzy rules is then found by using a FAM (Fuzzy associative memories) approach.
The proposed procedure was implemented to control the rotor position of a DC motor.

Section 1 Introduction

The synthesis of a control process generally requires a human expert who, on the basis of his heuristic knowledge about the system **establishes** and, above all, **describes** a control algorithm. In this way we obtain an approximate control. To improve control performances it's required a great effort to tune, in a **heuristic** way, the membership functions defining the shape and the position of **each** fuzzy set. This action can take a relatively long time to be performed, representing in the majority of the cases the most time consuming task in the whole design of a fuzzy controller. For this reason the possibility of automatically performing the synthesis and analysis of a fuzzy controller becomes of particular interest in order to obtain high performances while shorting the design cycle.
Starting from the consideration that fuzzy systems are **non linear systems**, classical techniques available for analysis and synthesis of linear systems **cannot** be employed. It becomes therefore necessary to develop new methods that allow to easily design fuzzy controllers.

Section 2 Cell-to-Cell approach.

The growing interest towards the dynamics of nonlinear systems has driven the development of research methods like Poincarre`s maps. This approach allows to study complex nonlinear systems using a point-to-point technique through a great number of integrations. Poincarre`s maps allow to show typical aspects of nonlinear systems, like chaotic phenomena, periodic motions, etc., carrying out their importance in the analysis on local ambit. The major limitation of this technique is the impossibility to be used in a global analysis of the systems, because of the great computational burden resulting from a point-to-point analysis. Moreover, the knowledge of the system model and the inaccuracy of computational methods makes this approach redundant. For these reasons it's far more convenient an analysis approach like the Cell-to-Cell Mapping, according to which a subdivision of the state space in a finite numbers of cells is performed, and a system structural analysis for each cell is carried out. In this way, each state variable is considered like a collection of a finite numbers of cells. Each point belonging to a cell is associated to the variations of the same cell. Let:

$$\dot{x}(t) = F(t, x(t)) \qquad (1)$$

the mathematical model of the system, and **x** the state vector. The coordinate axis of a state variable x_i is divided into a large number of intervals, with interval size $\mathbf{h_i}$. Let $\mathbf{Z_i}$ an integer that identifies the interval i. The interval Z_i of the x_i-axis is defined to be the one which contains all the x_i's satisfying the relation:

$$(Z_i - 1/2)h_i \le x_i \le (Z_i + 1/2)h_i \qquad (2)$$

The N-tuple $Z_i = 1,2,\dots,N$, denoted **Z**, is called "cell vector" of the state space. The state space is now considered as a collection of cells, and the mapping from x(n) to x(n+1) of the point-to-point approach is here substituted by cell mapping **C** from **Z**(n) to **Z**(n+1):

$$\mathbf{Z}(n+1) = \mathbf{C}(\mathbf{Z}(n)) \quad Z_i(n+1) = C_i(\mathbf{Z}(n))$$

Section 3 FAM.

A Fuzzy Associative Memory is a collection of fuzzy rules of the form:

if $X_1 = A_1$ ***and*** $X_2 = A_2 \dots$ ***and*** $X_m = A_m$ ***then*** $Y = B$.

Given:

n_{Xi}, the number of fuzzy sets defined for each state variable X_i,

n_Y, the number of fuzzy sets defined for the control variable Y

then the number *r* of possible fuzzy rules that can be defined on the state-control space is:

$$r = n_Y \cdot \prod_{i=1}^{m} n_{Xi}$$

This value grows quickly, and all the possible rules are certainly not needed to model the control task. The actual number of rules is much smaller.

Kosko [Kos92] proposes an adaptive algorithm for the definition of a suitable set of fuzzy rules. He also suggests the use of neural networks for the identification of such set of rules, starting from a training set consisting of examples of the input-output space of the problem.

The network we used is a winner-take-all network, that performs a clustering on the state-control space according to a well-known competitive learning algorithm (see [PDP86]). The network consists of two layers of units (the input and the output layer). Given m state variables, there are m+1 units in the input layer. Given *r* possible fuzzy rules for the problem, there are at most *r* units in the output layer. The rule of propagation of signals in the network is the usual inner product between the input vector **x** and each weight vector $\mathbf{w}_i$. Namely, for each output unit i, its activation value a_i is: $a_i = \mathbf{w}_i * \mathbf{x}$.

The output unit for which the activation value is highest wins the competition, and its weight vector is updated according to the simple rule of moving the weight vector towards the direction of the input vector, in such a way to minimize the distance E between the center of the cluster the input vector belongs to and the input vector:

$$E = \sum_{p \in training\,set} \left\| X^p_{input} - w^p_{centroid} \right\|^2$$

then the updating rule for weight vector $\mathbf{w^p}_{centroid}$ at time t+1 is given by:

$$w^p_{centroid,j}(t+1) = w^p_{centroid,j}(t) + \eta \cdot (x^p_j - w^p_{centroid,j}(t))$$

where η is a positive learning speed.

At the end of the training phase, each weight vector corresponding to an output unit that has won many times the competition represents the center of a cluster in the state-control space, i.e. it represents a fuzzy rule for the problem at hand.

In this way it is possible to determine through a neural network the number of suitable fuzzy rules for the problem, and the center of the involved fuzzy sets for each state variable and for the control variable. In fact, each vector $(x_1, x_2, \dots, x_m, y)$ obtained through the clustering algorithm corresponds to the fuzzy rule:

if $X_1 = x_1$ ***and*** $X_2 = x_2 \dots$ ***and*** $X_m = x_m$ ***then*** $Y = y$

where each value represents the center of a fuzzy set.

Section 4 Fuzzy controller synthesis.

In this section a method based on cell-to-cell mapping to perform the automatic synthesis of a fuzzy controller is described. The procedure of controller synthesis is divided into two different phases. In the first phase the "optimum" value of the control variable for each cell of the space state is computed. This ccmputation is carried out by using an optimization procedure of a particular performances index, obtaining a map of control input for the system. In the second phase, this map is transformed in a fuzzy control algorithm by using a FAM approach.

4.1 Optimization phase

Let's suppose to have a discrete model of the system to be controlled:

$$x_1(k+1) = f_1(x(k),u,k)$$
$$x_2(k+1) = f_2(x(k),u,k)$$
$$x_n(k+1) = f_n(x(k),u,k)$$

where $x \in R^n$, is the system state vector, u is the input signal, and k is the discrete time. This procedure is available also for those systems that cannot be mathematically modelized, but a neural or linguistic model can be obtained. Once the state variables of the system to be controlled are identified, the maximum range of variation for each of these variables is imposed. These variables represent the fuzzy controller inputs not yet fuzzyficated. The space state is divided into an adequate number of cells, each cell representing a possible initial condition, from which we want to control the system. The space state discretization strongly influences the control system performances: an approximate discretization allows an approximate control law. For each cell the optimum control law $u_0(.)$ is computed by using the following optimization procedure:

1) a functional F(.,u) is fixed, resuming the desired system performances. For example: F can be equal to the weighted sum of the control variable quadratic errors towards the reference values.

2) for each cell the functional initial value F_{in} is computed: the controller input variable values are those determined starting from the initial conditions represented by the cell and under the action of u_{nom};

3) it's calculate u_0 minimizing the functional.

This procedure is a gradient descent based algorithm, and for this reason the problem of local minima is present; to overcome this problem a simulated annealing has been implemented. At the end of the optimization procedure we obtain for each cell an "optimum" value,u_0, of the control variable to be imposed to the system in order to obtain the desired performances in a finite number of sampling period. This map represents, with more details, that part of information that, otherwise, should be developed by an expert.

4.2 Controller synthesis

The control law determined in the described way, is then used for the generation of the training set for the neural network: for each cell of the space state a pattern is generated of the form: (x1 x2 ... xm y).

Since the values of the state variables are uniformly distributed in the state space, it is expected that the clustering carried out by the neural network is determined by the control values. For the same reason, an optimal initialization of the network is possible: training patterns are used as the initial centroids in the state-and-control space, so that the network starts the learning process from a good initial situation. This initialization is improved by forecasting the number of fuzzy sets for each state variable, and consequently by allowing a small overlap between neighbor clusters.

After training, some weight vectors will be interpreted as fuzzy rules governing the system: given the weight vector w_r corresponding to output unit r, $w_r = (w_1 \dots w_{m+1})$, if each w_i ($i = 1 .. m$) is in the range of variation of the state variable X_i, and w_{m+1} is in the range of variation of control variable y, and unit r has won many times the competition, then w_r can be interpreted as a fuzzy rule R governing the controller behavior:

$$R = \textit{if } X1 = w_1 \textit{ and } X2 = w_2 \ldots \textit{and } Xm = w_m \textit{ then } Y = w_{m+1}$$

After having found the interesting fuzzy sets and rules, it is possible to define the membership function for each fuzzy set. For each fuzzy set, the corresponding membership function f(x) has value 1 in $x = w_i$, and its shape is triangular of height 1. Moreover, the membership functions of two adjacent fuzzy sets initially have an overlap such that the sum of the activation values of each point in the discourse universe is equal to 1.

In this way, the fuzzy controller structure is completely determined.

Section 5 DC motor control

To show the capability of the proposed procedure, an example for the design of a fuzzy controller to regulate both speed and position of the rotor of a **DC MOTOR** is presented.

The motor is supposed to be connected to a load with a friction **f** and inertia **J**. The control task is to track a variable trajectory with a zero state error, minimum overshoot and minimum settling time. The system model is a second order one. The input variables to the controller are the error between rotor position and setpoint, and the rotor angular speed. The fuzzy controller output is the statoric voltage. The ranges of these variables are imposed from physical limit. By using the proposed procedure a fuzzy controller was implemented and the control performances was tested. Figure(1) shows the result of the control realized in the described way; these result are compared with the ones obtained by using a PID. We found that the fuzzy controller proposed has better performances than PID both for the control task and control robustness. In fact the fuzzy controller is more insensitive both to the system parametric variations and to external noises than PID does.

References

[IEEE92] IEEE International Conference on Fuzzy Systems, San Diego, CA, 1992

[IIZUKA92] International Conference on Fuzzy Logic and Neural Networks, Iizuka, Japan, 1992

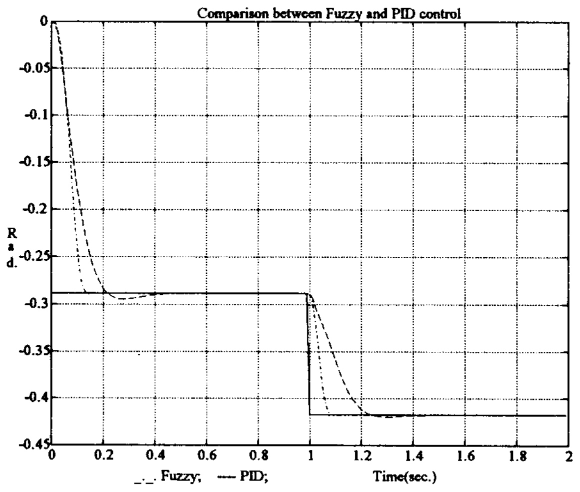
[PDP86] J. Mc.Clelland, D. Rumelhart and the PDP group, Parallel Distributed Processing, vol. 1, MIT press, 1986

[Kos92] B. Kosko, Fuzzy Systems and Neural Networks, Prentice Hall, 1992

[Zad73] L. Zadeh, Outline of a new approach to the analysis of complex systems and decision process. IEEE transactions on Systems, Man and Cybernetics, vol. SMC-3, n. 1, jan. 1973

[HsGu] C. Hsu and r. Guttalu, An unravelling algorithm for global analysis of dynamical system: an application of Cell-to-Cell Mapping. Journal of Allied Mechanics, vo. 7.

[Chen] The analysis of fuzzy dynamic systems using Cell-to-Cell Mapping. IEEE transactions.



Fig(1) Comparison between Fuzzy and PID control