3

Evolution of Fuzzy Controllers and Applications

Dilip Kumar Pratihar[1]* and Nirmal Baran Hui[2]

[1] Associate Professor
dkpra@mech.iitkgp.ernet.in
[2] Research Scholar
nirmal@mech.iitkgp.ernet.in
Soft Computing Lab.
Department of Mechanical Engineering
Indian Institute of Technology, Kharagpur
Kharagpur-721 302
India

Summary. The present chapter deals with the issues related to the evolution of optimal fuzzy logic controllers (FLC) by proper tuning of its knowledge base (KB), using different tools, such as least-square techniques, genetic algorithms, back-propagation (steepest descent) algorithm, ant-colony optimization, reinforcement learning, Tabu search, Taguchi method and simulated annealing. The selection of a particular tool for the evolution of the FLC, generally depends on the application. Some of the applications have also been included in this chapter.

Keywords: Fuzzy logic controller, Evolution, Least-square technique, Genetic-fuzzy system, Neural-fuzzy system, Ant-colony optimization, Reinforcement learning, Tabu search, Taguchi method, Simulated annealing

3.1 Introduction

Real-world problems are generally associated with different types of uncertainties. In the past, considerable effort has been made to model these uncertainties. Prior to 1965, it was considered that *probability theory* working based on Aristotelian two-valued logic was the sole agent available to deal with uncertainties. This particular logic uses the concept of the classical crisp set. That is a set with a fixed boundary. Prof. Zadeh developed the concept

* Corresponding author: Associate Professor, Department of Mechanical Engineering, Indian Institute of Technology, Kharagpur-721 302, India; http://www.facweb.iitkgp.ernet.in/∼dkpra

D.K. Pratihar and N.B. Hui: *Evolution of Fuzzy Controllers and Applications*, Studies in Computational Intelligence (SCI) **66**, 47–69 (2007)
www.springerlink.com

of fuzzy sets, in the year 1965 [1]. Those are the sets having the vague boundaries. He argued that probability theory can handle only one out of several different types of possible uncertainties. Thus, there are uncertainties, which cannot be dealt with by using the probability theory. Taking an example, in which Mr. X requests Mr. Y, to bring some *red* apples for him from the market. There are two uncertainties at least, which relate to the following: (i) the availability of the apples, and (ii) a guarantee that the apple is red. Depending on the season, there is a probability of obtaining the apples, which varies between 0 and 1. But, the colour – *red* cannot be defined by the classical set. It is not between red (1) and not-red (0). In the fuzzy set, the colour – *red* can be defined as follows (Fig. 3.1) using the concept of membership of an element to a class. That is the function value (μ): If the colour is perfectly red PR, then it may be said red with a membership value of 1.0; if it is R, then it is considered to be red with a membership value of 0.65; if it is slightly red SR, then it is red with a membership value of 0.39. If it is not red (NR), then also it is red with a membership value of 0.0. In this way, the uncertainty related to the colour of the apples can be handled. Thus, a fuzzy set may be considered to be a more general concept than the classical set.

The concept of fuzzy set theory has been used in a number of applications, such as the Fuzzy Logic Controller (FLC), fuzzy clustering, fuzzy mathematical programming, fuzzy graph theory and other examples. Out of all such applications, FLC is the most popular application for the following reasons – (i) ease of understanding and implementations, (ii) ability to handle uncertainty etc. An exact mathematical formulation of the problem is not required for the development of an FLC. This feature makes it a natural choice for solving complex real-world problems. These are either difficult to model mathematically or the mathematical model becomes highly non-linear. It is to be noted that a fuzzy logic controller was first developed by Mamdani and Assilian, in the year 1975 [2]. The concept of fuzzy set was published in the

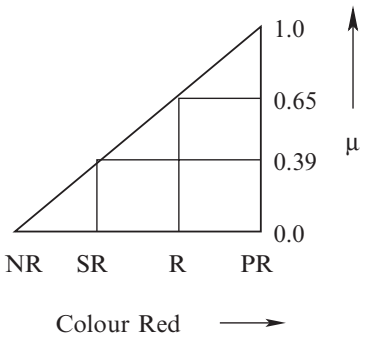

Fig. 3.1. A schematic diagram explaining the concept of membership function distribution.

year 1965. Human beings have the natural ability of determining the input-output relationships of a process. The behavior of a human being is modeled artificially, when designing a suitable FLC. The performance of an FLC depends on its knowledge base (KB), which in turn consists of both Data Base (DB) and a Rule Base (RB). The DB consists of data related to membership function distributions of the variables of the process to be controlled. Designing a proper KB of an FLC is a difficult task, which may be implemented in one of the following ways:

- Optimization of the data base only,
- Optimization of the rule base only,
- Optimization of the data base and rule base in stages,
- Optimization of the data base and rule base simultaneously.

The membership function distributions are assumed to be either Linear such as, triangular, trapezoidal or Non-Linear. The Non-Linear can be Gaussian, bell-shaped, sigmoidal in nature. To design and develop a suitable FLC for controlling a process, its variables need to be expressed in the form of some linguistic terms (such as VN: Very Near, VF: Very Far, A: Ahead for example). The relationships between the input (antecedent) and output (consequent) variables are expressed in the form of rules. For example, a rule can be expressed as indicated in Fig. 3.2:

$$\textbf{IF } I_1 \text{ is } N \textbf{ AND } I_2 \text{ is } A \textbf{ THEN } O \text{ is } AR,$$

The number of such rules will be present in the rule base. The number of linguistic terms used to represent the variables increases in order to improve the accuracy of the prediction. The computational complexity of the controller will increase with a larger number of rules. For easy implementation in either

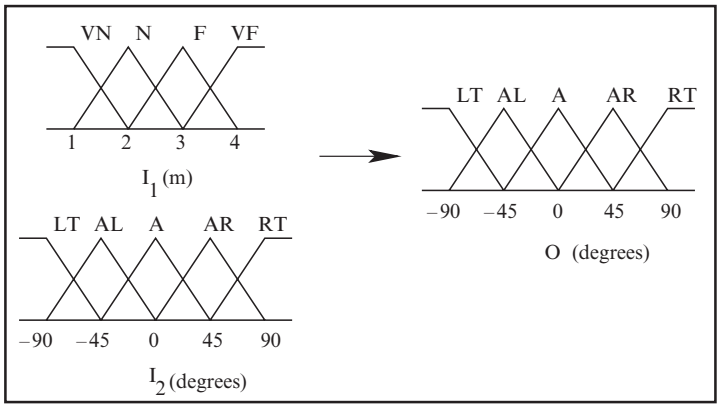

Fig. 3.2. A diagram showing some membership function distributions of input and output variables of the Fuzzy Logic Controller.

the software or the hardware, the number of rules present in the Rule Base should be as small as possible. Consequently, some investigators have tried to design and develop a hierarchical FLC, in which the number of rules will be kept to the minimum [3, 4]. It has been observed that the performance of an FLC largely depends on the rule base and optimizing the data base is a fine tuning process [5]. A fuzzy logic controller does not have an internal optimization module. An external optimizer is used to develop an optimal Knowledge Base through a proper tuning and this helps to improve the performance.

In this chapter, the focus is on the issues related to design and development of an optimal fuzzy logic controller using different optimization tools. Some of the applications of FLC are cited.

The remainder of the text is organized as follows. Two major forms of FLC are discussed in Section 2. Various methods of designing optimal FLCs are given in Section 3. A summary of this work is presented in Section 4.

3.2 Two Major Forms of Fuzzy Logic Controller

System modeling done by using the fuzzy set concept can be classified into two groups. That is *linguistic fuzzy modeling* and *precise fuzzy modeling*. Linguistic fuzzy modeling, such as *Mamdani Approach* is characterized by its high interpretability and low accuracy. The aim of precise fuzzy modeling such as *Takagi and Sugeno's Approach*, is to obtain high accuracy at the cost of interpretability. Interpretability of a fuzzy modeling is defined as a capability to express the behavior of a system in an understandable form. This is expressed in terms of compactness, completeness, consistency and transparency. The accuracy of a fuzzy model indicates how closely it can represent the system modeled. The working principles of both these approaches are briefly explained below.

3.2.1 Mamdani Approach [2]

An FLC consists of four modules namely, a fuzzy rule base, a fuzzy inference engine, fuzzification and de-fuzzification. Fig. 3.3 shows a schematic diagram explaining the working of an FLC.

(a) The condition known as the antecedent and the action called the consequent variables needed to control a process are identified and measurements are taken of all the condition variables.
(b) The measurements taken in the previous step are converted into appropriate fuzzy sets to express measurement uncertainties. This process is known as *fuzzification*.
(c) The fuzzified measurements are then used by the inference engine to evaluate the control rules stored in the fuzzy rule base and a fuzzified output is determined.

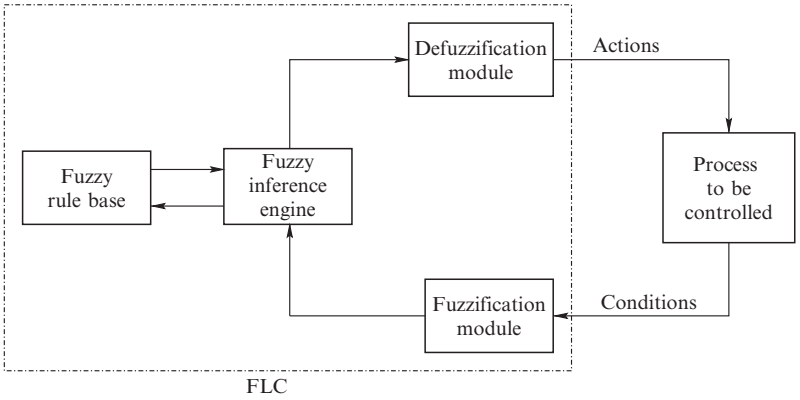

Fig. 3.3. The working cycle of an FLC.

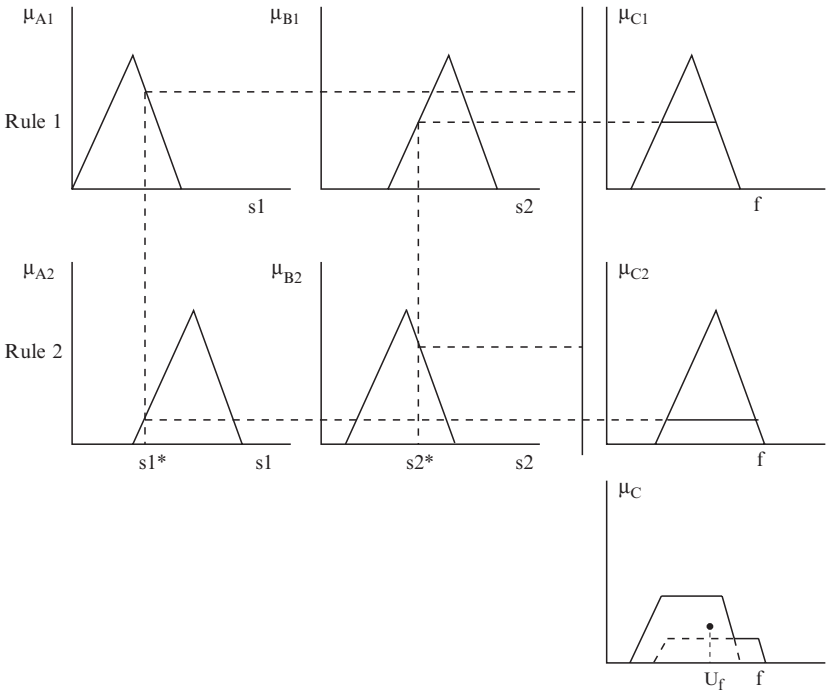

Fig. 3.4. A schematic diagram showing the working principle of an FLC.

(d) The fuzzified output is then converted into a single crisp value. This conversion is called *de-fuzzification*. The de-fuzzified values represent actions which need to be taken by the FLC in controlling the process.

The fuzzy reasoning process is illustrated in Figure 3.4. Let us assume for simplicity that only two fuzzy control rules (out of many rules present in the

rule base) are being 'FIRED' as shown below. This is for a set of inputs – ($s1^*$, $s2^*$).

RULE 1: IF $s1$ is $A1$ and $s2$ is $B1$ THEN f is $C1$
RULE 2: IF $s1$ is $A2$ and $s2$ is $B2$ THEN f is $C2$.

If $s1^*$ and $s2^*$ are the inputs for fuzzy variables $s1$ and $s2$. If μ_{A1} and μ_{B1} are the membership function values for A and B, respectively, then the grade of membership of $s1^*$ in $A1$ and the grade of membership of $s2^*$ in $B1$ are represented by $\mu_{A1}(s1^*)$ and $\mu_{B1}(s2^*)$, for rule 1.

Similarly, for rule 2, where $\mu_{A2}(s1^*)$ and $\mu_{B2}(s2^*)$, are used to represent the membership function values. The firing strengths of the first and second rules are calculated as follows:

$$\alpha_1 = \min\left(\mu_{A1}(s1^*), \mu_{B1}(s2^*)\right), \tag{3.1}$$

$$\alpha_2 = \min\left(\mu_{A2}(s1^*), \mu_{B2}(s2^*)\right). \tag{3.2}$$

The membership function of the combined control action C is given by

$$\mu_C(f) = \max\left(\mu^*_{C1}(f), \mu^*_{C2}(f)\right). \tag{3.3}$$

There are several methods of defuzzification (shown in Fig. 3.5). These are explained below.

1. **Center of Sums Method:** According to this method of defuzzification (refer to Fig. 3.5(a)), the crisp output can be determined by the following.

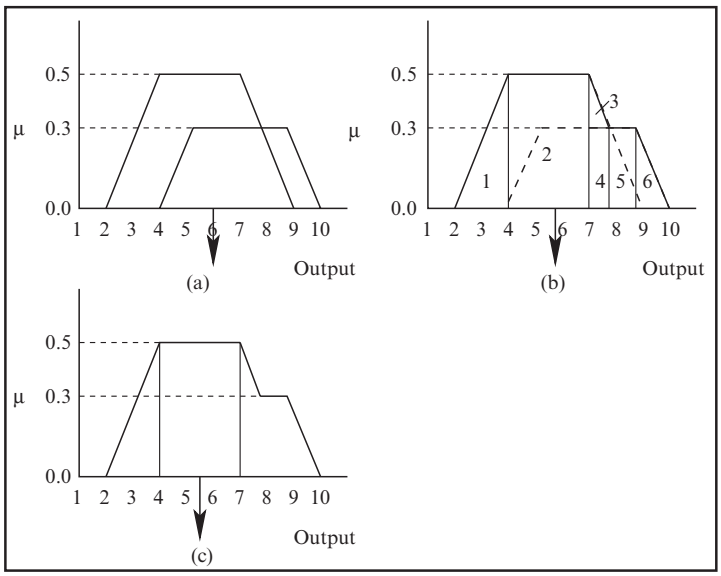

Fig. 3.5. Different methods of defuzzification.

$$U_f = \frac{\sum_{j=1}^{p} A(\alpha_j) \times f_j}{\sum_{j=1}^{p} A(\alpha_j)}, \tag{3.4}$$

Where U_f is the output of the controller. $A(\alpha_j)$ represents the firing area of the j-th rule. p is the total number of the fired rules. f_j represents the centroid of a membership function.

2. **Centroid Method:** The total area of the membership function distribution used to represent the combined control action is divided into a number of standard sub-areas. Their area and the center of area can be determined easily (refer to Fig. 3.5(b)). The crisp output of the controller can be calculated by using the expression given below.

$$U_f = \frac{\sum_{i=1}^{N} A_i f_i}{\sum_{i=1}^{N} A_i}, \tag{3.5}$$

Where N indicates the number of small areas or regions, A_i and f_i represent the area and the center of area of i-th small region.

3. **Mean of Maxima Method:** From the membership function distribution of the combined control action, the range of the output variable is located. This is where the maximum value of the membership function is reached. The mid-value of this range is considered to be the crisp output of the controller (refer to Fig. 3.5(c)).

3.2.2 Takagi and Sugeno's Approach [6]

Here, a rule consists of the fuzzy antecedent and the functional consequent parts. Thus, a rule can be represented as follows:

$$\begin{gathered}\textbf{If } x_1 \text{ is } A_1^i \text{ and } x_2 \text{ is } A_2^i \text{ ..... and } x_n \text{ is } A_n^i \\ \textbf{then } y^i = a_0^i + a_1^i x_1 + \ldots + a_n^i x_n\end{gathered}$$

where $a_0, a_1, \ldots, a_n$ are the coefficients. In this way, nonlinear system is considered as a combination of several linear systems. Control action of i-th rule can be determined for a set of inputs $(x_1, x_2, \ldots, x_n)$ as follows.

$$w^i = \mu_{A_1}^i(x_1)\mu_{A_2}^i(x_2)\ldots\mu_{A_n}^i(x_n), \tag{3.6}$$

Where $A_1, A_2, \ldots, A_n$ indicate the membership function distributions of the linguistic terms used to represent the input variables. The membership function value is given by μ. Thus, the combined control action can be determined as

$$y = \frac{\sum_{i=1}^{k} w^i y^i}{\sum_{i=1}^{k} w^i}, \tag{3.7}$$

where k is the total number of rules.

3.3 Methods of Designing Optimal Fuzzy Logic Controllers

In order to establish the input-output relationships of a process, a designer tries to design the KB of an FLC manually, based on a knowledge of the process. In most of the cases, it is difficult to gather prior information of a process. The manually-designed KB of the FLC may not be optimal. As an FLC does not have a in-built optimizer, an optimization tool is used, while tuning a KB. Several methods have been developed and some of these are discussed below.

3.3.1 Least-square Method

Attempts were made to determine an appropriate shape of the membership function distributions by using least-square methods. In this connection, see Pham and Valliappan [7], Bustince et al. [8]. The membership function distribution of a fuzzy set was assumed to follow a power function such as $\mu_A(x_i) = ax_i^b$. Here x indicates a variable represented by a fuzzy set A, $i = 1, 2, \ldots, n$, n is the number of training cases, μ_A is the membership function value of the fuzzy set A lying between 0 and 1, a (greater than zero) and b are the constants to be determined by the least-square method. Two equations were solved for this [8]:

$$nlna + \left(\sum_{i=1}^{n} lnx_i\right) b = \sum_{i=1}^{n} ln\mu_A(x_i) \tag{3.8}$$

$$\left(\sum_{i=1}^{n} lnx_i\right) lna + \left(\sum_{i=1}^{n} ln^2 x_i\right) b = \sum_{i=1}^{n} lnx_i ln\mu_A(x_i) \tag{3.9}$$

where $ax_i^b \leq 1$.

3.3.2 Genetic-Fuzzy System

Genetic algorithm (GA) [9] is a population-based search and optimization technique based on the principle of natural selection and mechanics of natural genetics, was used by several researchers, for a genetic-fuzzy system. The performance of a Fuzzy Logic Controller (FLC) is dependent on its KB. Fig. 3.6 shows the schematic diagram of the genetic-fuzzy system. Here, a GA is used to determine optimal KB of the FLC. Thus, the GA improves the performance of the FLC. During optimization of the FLC, the feedback which is a deviation in prediction is calculated. This is based on a set of training cases and it is utilized as the fitness of the GA. A GA is computationally expensive and the tuning is done off-line. Once optimized, the FLC will be able to predict the outputs for a set of inputs, within a reasonable accuracy

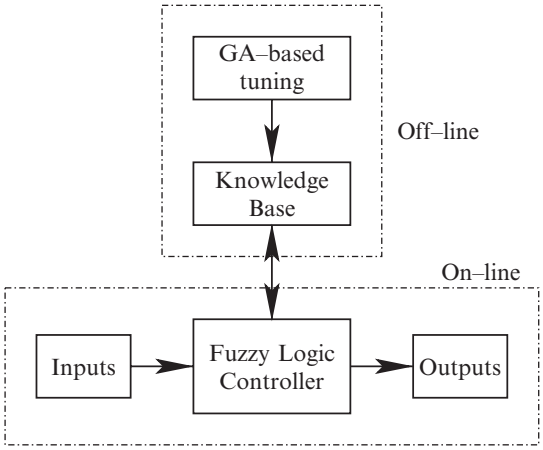

Fig. 3.6. A schematic diagram showing a genetic-fuzzy system

limit. This concept has been used to solve a number of physical problems. See Karr [10], Thrift [11], Pham and Karaboga [14]. A detailed review on this scheme is done by Cordon et al. [12].

There are three basic approaches of this scheme, the Pittsburgh [13, 14], Michigan [15] and iterative rule learning [16, 17] approaches. In *Pittsburgh approach*, the entire rule base of the FLC is represented by a GA-string. Thus, the GA-population indicates the population of candidate rule sets. The genetic operators are used to modify the rule sets and obtain the optimal rule base. In the *Michigan approach*, members of the population are individual rules. Thus, a rule set is represented by the entire population. The main drawback of these two approaches lies in the fact that for the large number of fuzzy rules, the GA requires a huge amount of computer memory. To overcome the problem, using an *iterative rule learning approach*, chromosomes code individual rule, a new rule is added to the rule set, in an iterative fashion, for every run of GA. It requires a proper encoding scheme for extracting the rules from a chromosome. In this approach, the evolved RB of the FLC may contain some redundant rules, due to the iterative nature of the GA.

A considerable amount of work has been carried out in this field of research. Some of these attempts are mentioned below. Furuhashi et al. [18] developed a variable length decoding method, known as the *Nagoya Approach*. Using this approach, as the lengths of the chromosomes are not fixed, it is difficult to implement the necessary crossover operation in GA. Again the simultaneous design of the data base and rule base requires a proper optimization procedure. This can tackle both continuous as well as integer variables. Wang and Yen [19] proposed a method, in which a GA was used to extract the rule base, and the data base of an FLC was optimized using a Kalman filtering technique. Farag et al. [20] developed a new multi-resolutional dynamic GA for this purpose. In this, the initial parameters of the data base of an FLC were determined by

using Kohonen's self-organizing feature map algorithm and optimization was done by using a GA. Fuzzy rule generation and tuning using a GA was also tried by Ishibuchi et al. [21]. Recently, Abdessemed et al. [22] proposed a GA-based procedure for designing an FLC, to control the end effector's motion of a planar manipulator. Yupu et al. [23] used a GA to search for appropriate fuzzy rules. The membership function distributions were optimized by using a neural network. The FLC is becoming more popular nowadays, developing a suitable knowledge base for it, is not easy. The designer requires much time, to initially design the knowledge base (KB). It is further improved by using GA-based tuning. Thus, the designer must have a knowledge of the process to be controlled by the FLC. To overcome this requirement, a few investigators [24, 25] tried to automatically design the FLC by using a GA. Using search, the GA will develop the optimized data base and rule base for the FLC.

A GA is basically a fitness function-driven search method, therefore, it is blind for any other aspect that is not explicitly considered on fitness function. Hence, a GA might evolve some redundant rules, that have limited influence on the process to be controlled. Redundant rules are to be removed to make the rule base compact. This makes the implementation of the controller easier, particularly when it is done by hard-ware. Thus, there is a need to determine the contribution of each rule. In this context, the work of Nawa et al. [26], Ishibuchi and Nakashima [27], Ghosh and Nath [28], Hui and Pratihar [35] are important. Nawa et al. [26] measured the quality of a rule by determining its accumulated truth value. The accumulated truth value was considered to be the sum of probability of occurrences of a rule in the training data. A rule is said to be good, if its accumulated truth value is high. Ishibuchi and Nakashima [27] made an attempt to assign an importance factor to each rule. They calculated the importance factor of a rule, by considering the way it interacts with the neighbors. An evolutionary technique was utilized to find the interaction effect. Ghosh and Nath [28] investigated the effectiveness of a rule by measuring three parameters, namely *support count*, *comprehensibility* and *interestingness*. *Support count* of an item set is defined by the number of records in the data base that contains all the items of that set. *Comprehensibility* is used to justify the understandability of a rule. A rule is said to be more comprehensive, if the number of attributes associated with the antecedent part of the rule is less and *interestingness* is represented by the probability of generating a rule during the learning process. It was a theoretical approach of finding interesting rules in the rule base and is unable to predict the importance of a rule for a fixed number of attributes in both antecedent as well as in the consequent parts. The above methods considered the probability of occurrence of a rule only, for the determination of a good rule base. No attention was paid to calculate the contribution effect of a rule with respect to a specific objective. Hui and Pratihar [35] proposed a method of determining *importance factor* for each rule contained in the RB of an FLC, to check the redundancy, if any. The importance factor of a rule is calculated

by considering its probability of occurrence and worth (goodness). A rule is said to be redundant and thus may be eliminated, if its importance factor comes out to be smaller than a pre-specified value and the removal of which does not lead to any non-firing situation.

The genetic-fuzzy system has been developed by the authors also, following the two different approaches discussed below.

Approach 1: GA-based tuning of the manually-constructed KB of the FLC. The KB of the FLC is designed manually and is based on the designer's experience of the problem to be solved. But, it may not be optimal in any sense. GA-based tuning is adopted, to further optimize the KB, to improve the performance. As a GA is found to be computationally expensive, the GA-based tuning is carried out, off-line. During optimization, the GA-string will carry information for both the data base as well as the rule base. The GA-search will find the optimal KB of the FLC. Once optimized, the FLC is able to determine its outputs in the optimal sense.

Approach 2: Automatic design of KB using a GA. In Approach 1, much time is spent on manual design of the KB of an FLC. It might be difficult beforehand to foresee the characteristics of the process to be controlled. Thus, designing a proper KB might be a difficult task. To overcome this, a method for automatic design of the KB is developed by using a GA. Here the task of designing a suitable KB is given to the GA. The GA through its exhaustive search will try to determine the optimal KB of the FLC.

The above concept has been used by the authors, to solve a number of physical problems. One of them is explained below.

Optimal Path and Gait Planning of a Six-legged Robot A six-legged robot will have to plan its time-optimal, collision-free path as well as the optimal gait, setting simultaneously the minimum number of ground-legs having the maximum average kinematic margin. This is while moving on a flat terrain with occasional hurdles, such as ditches and some moving obstacles. Its stability margin should always be positive to ensure static stability. This is a complicated task because the path planning and gait planning must be done simultaneously [29]. Fig. 3.7 shows the optimal path and gait for a six-legged robot. It has planned its optimal path and gait, after starting from an initial position S to reach the final position G. It faces three moving obstacles and a ditch on the way to-wards its goal. The total movement of the robot has been achieved through a number of segments called motion segments. The robot plans its optimal path and gait on-line, for each motion segment. The robot shown in Fig. 3.7, is found to reach its goal in the time-optimal sense at 79-th motion segment, after avoiding collision with the moving obstacles and generating its optimal gaits.

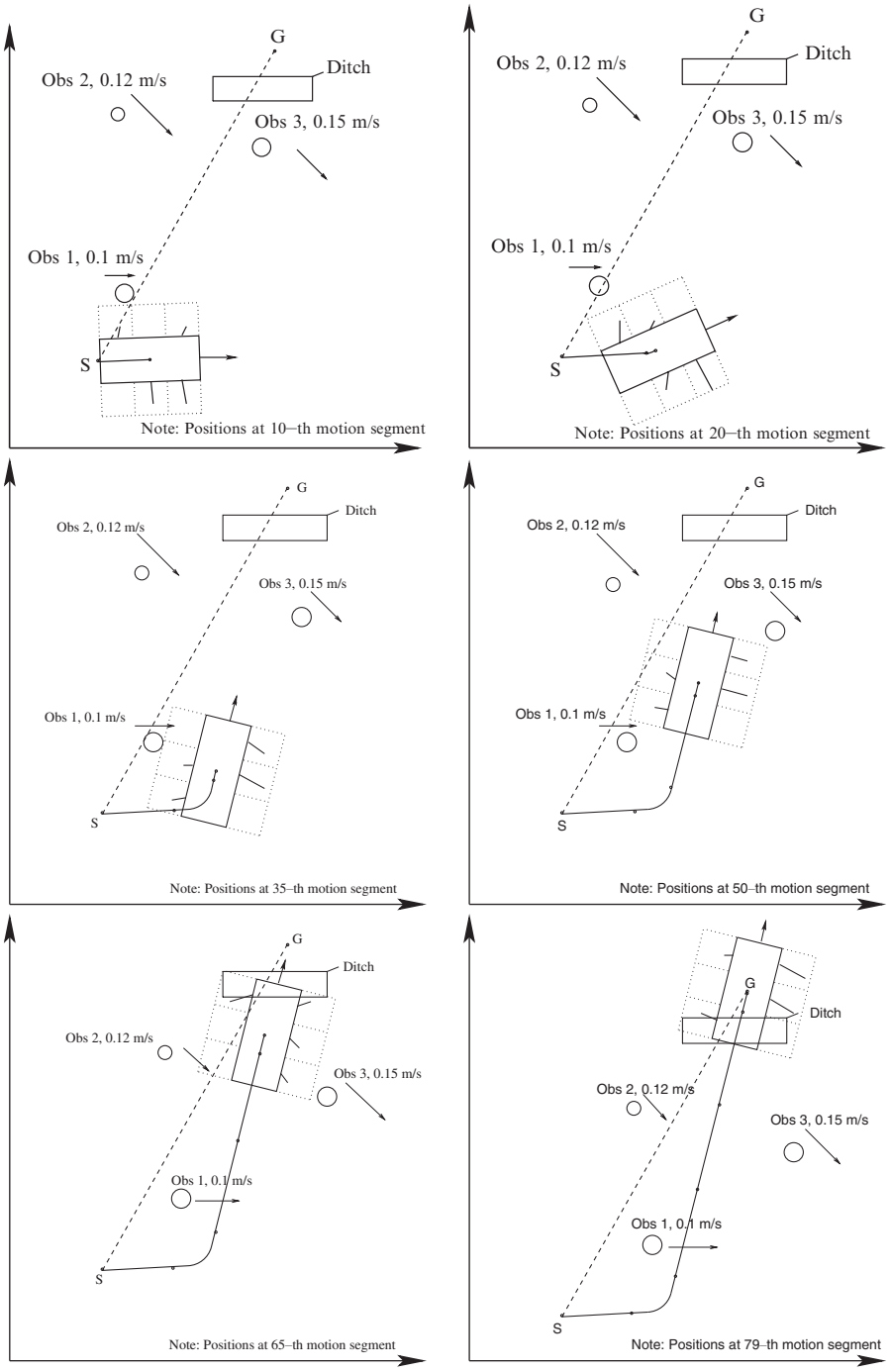

Fig. 3.7. Optimal path and gaits of a six-legged robot obtained using the genetic-fuzzy system [29]

3.3.3 Neural-Fuzzy System

The purpose of developing a neural-fuzzy system is to improve the performance of an FLC by using neural network-based learning. It had been utilized by a number of researchers to solve a variety of problems. Some of these are mentioned below. Marichal et al. [30] proposed a neuro-fuzzy approach to generate the motion of a car-like robot navigating among static obstacles. In their approach, a least mean squared algorithm was used for the learning purposes and Kohonen's self organizing feature map algorithm was considered to obtain the initial number of fuzzy rules and fuzzy membership function centers. They did not optimize the traveling time nor the approach was tested in a dynamic environment. Song and Sheen [31] suggested a pattern recognition approach based on a fuzzy-neuro network for the reactive navigation of a car-like robot. Li et al. [32] developed a neuro-fuzzy architecture for behavior-based control of a car-like robot, that navigates among static obstacles.

The present chapter includes two schemes of neural-fuzzy system developed by the authors. These are discussed below [33].

Scheme 1: Neural-fuzzy system based on Mamdani Approach. In the developed neural-fuzzy system, a fuzzy logic controller using Mamdani Approach is expressed by utilizing the structure of a Neural Network (NN) and a back-propagation algorithm is utilized to optimize the KB of the FLC. The back-propagation algorithm is a steepest descent algorithm. Fig. 3.8 shows the schematic diagram of the five layer neural-fuzzy system– Layer 1 is the input layer, fuzzification is done in Layer 2, Layer 3 indicates the AND operation. The OR operation is carried out in Layer 4, and Layer 5 is the output layer. The training cases are passed through the network and the total error is calculated. The average error is propagated in the backward direction, to determine the updated weights. The network will try to find an optimal set of weights, corresponding to which the error is minimum.

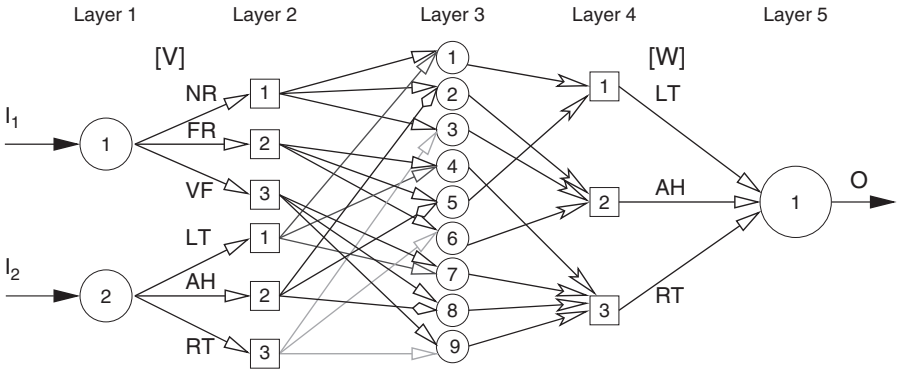

Fig. 3.8. A schematic diagram of the neural network-structured FLC

Three different approaches to Scheme 1 are developed. These are discussed in brief below.

Approach 1: NN-tuned FLC. The initial weights of the neural network representing the FLC are generated, at random. A batch mode of training is adopted. Training cases are passed through the NN (i.e., forward propagation) and average error is determined. As this error depends on the weights, it can be minimized by updating the weight values. A back-propagation algorithm is used to minimize the error.

Approach 2: Genetic-Neural-Fuzzy system. In Approach 1, the error is minimized using a steepest descent method. This may have the local minima problems. To overcome this problem, the back-propagation algorithm is replaced by a GA. As GA is a population-based search and optimization method, the chance of its solutions for getting trapped into the local minima is less. Thus, Approach 2 may be expected to perform better than Approach 1.

Approach 3: Automatic design of neural-fuzzy system. To increase the search space of the GA, a method for automatic design of neural-fuzzy system is proposed. In this approach, the outputs of different rules are evolved solely by the GA itself. The GA through its exhaustive search, determines a good rule base for the FLC. There might be some redundant rules present in the GA-designed rule base. It may happen due to the iterative nature of the GA. To identify the redundant rules, a method is proposed, in which importance of a rule is decided by considering its *frequency of occurrence* and its *worth* with respect to the objective function of the optimization problem. Based on the value of this importance factor, a decision is taken whether a particular rule will be declared as redundant.

Scheme 2: Neural-fuzzy system based on Takagi and Sugeno Approach. A neural-fuzzy system has been developed based on the Takagi and Sugeno Approach. This is known as the ANFIS (i.e., Adaptive Neuro-Fuzzy Inference Systems) [34]. An ANFIS is a multi-layered feed forward network, in which each layer performs a particular task. The layers are characterized by the fuzzy operations they perform. Fig. 3.9 shows the schematic diagram of the ANFIS structure, which consists of six layers – Layer 1 (input layer), Layer 2 (condition layer), Layer 3 (rule base layer), Layer 4 (normalization layer), Layer 5 (consequence layer) and Layer 6 (output layer). Let us assume that there are two inputs – I_1 and I_2 and one output O of the network. The first two layers perform similar tasks to those done by Layers 1 and 2 of the neuro-fuzzy system developed in Scheme 1. The functions of other layers are explained below.

Layer 3: This layer defines the rules of the fuzzy inference system. As three linguistic terms are used to represent each of the two inputs, there is a maximum of $3 \times 3 = 9$ rules present in the rule base. Each neuron in this layer represents a fuzzy rule and is termed as a rule node. The output of each neuron lying in this layer is the multiplication of their

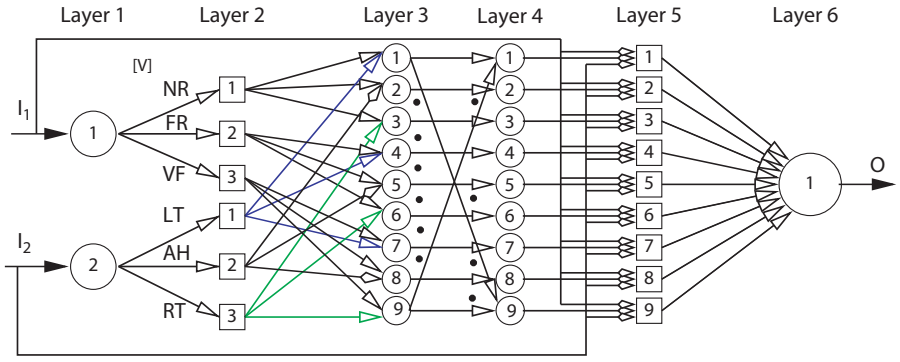

Fig. 3.9. A schematic diagram of the ANFIS architecture

respective two membership values. It is to be noted that each node output represents the firing strength of a rule.

Layer 4: This layer has the same number of nodes as the previous layer. It calculates the normalized firing strength of each node.

Layer 5: The output of a particular neuron (say, the q-th) lying on this layer is determined by

$$O_{5q} = (a_q I_1 + b_q I_2 + c_q) \tag{3.10}$$

where (a_q, b_q, c_q) represents one set of coefficients associated with the q-th node.

Layer 6: The output of the node lying on Layer 6, can be determined by summing up all incoming signals.

$$O_{61} = \sum_{q=1}^{R} O_{5q}, \tag{3.11}$$

where R indicates the total number of rules. A maximum of four rules (out of nine) will be fired, for one set of input variables. The performance of an ANFIS depends on the selection of consequence parameters and premise parameters. That is the half base-widths of the input membership function distributions. For the selection of optimal parameters, a GA might be used together with the ANFIS.

The developed neural-fuzzy systems have been used to plan collision-free, time-optimal paths of a car-like robot. This is explained below.

3.3.3.1 Collision-free, Time-optimal Path Planning for a Car-like Robot [33, 35]

A car-like mobile robot needs to find its time-optimal and collision-free path while navigating among some moving obstacles, and satisfy its kinematic (non-holonomic) constraints and dynamic constraints (such as sliding constraint,

motor torque constraint, curvature constraint). A detailed discussion on these constraints is beyond the scope of this chapter. Interested readers may refer to [33], for the same. The total path of the robot is divided into a number of distance steps having varying lengths, each of which is traveled during a time step. To calculate total traveling time of a robot to reach its destination, the time steps are summed and the time required to align its main axis towards the goal is added. There can be a saving in traveling time, particularly if the robot does not change its direction in two successive distance steps. It is subtracted from the total traveling time. The aim is to minimize the traveling time after ensuring a collision-free movement of the robot. A high positive value penalty is added to the total traveling time, if the robot collides with any one of the obstacles. Fig. 3.10 shows the near-optimal, collision-free paths of a robot in the presence of 16 moving obstacles. This is as obtained by using the three approaches of Scheme 1, Scheme 2 (explained above) and a

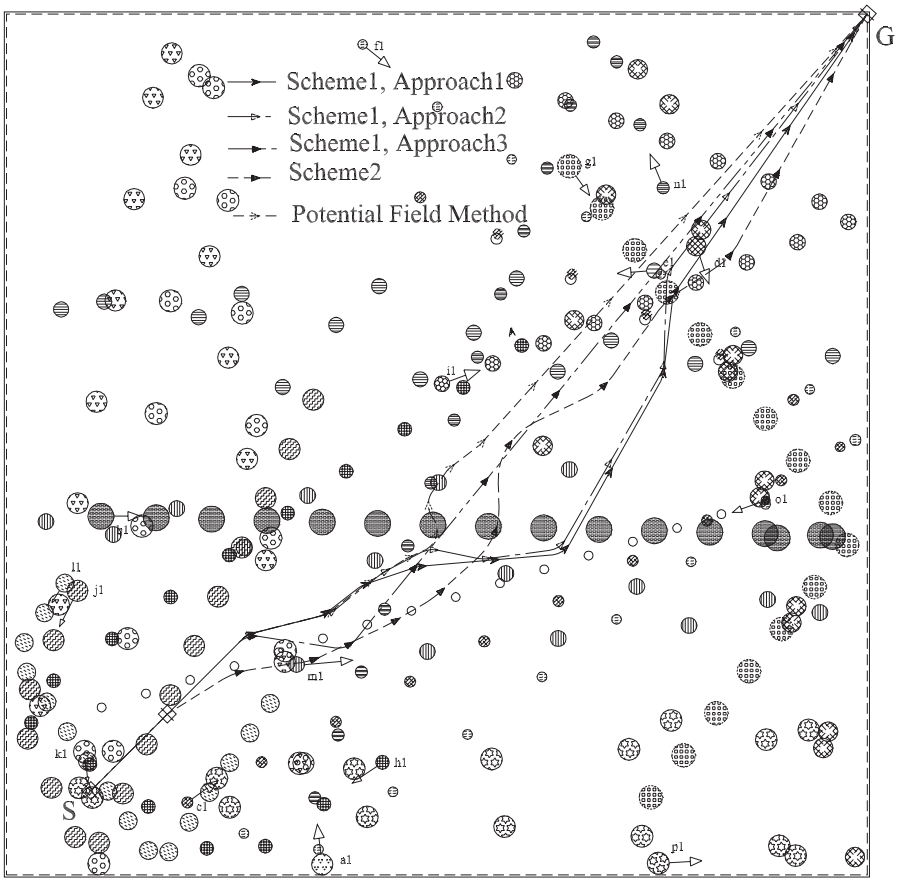

Fig. 3.10. Navigation of a robot among 16 moving obstacles

traditional motion planning scheme (potential field method) [36]. The initial position, size, velocity and direction of movement of the obstacles are created at random. The planning robot starts from the point S and reaches the goal G, by avoiding collisions with the obstacles. Soft computing-based approaches have proved their supremacy over the potential field method. It could be due to the reason that there is a chance that the solutions of the potential field method will get trapped at the local minima. On the other hand, the chance of the solutions of GA-tuned fuzzy logic controller for getting trapped into the local minima is less and it could be due to an exhaustive search carried out by the GA. Moreover, the GA is able to inject adaptability to the FLC, which has been observed from the performances of Approaches 2 and 3 of Scheme 1. Approach 3 of Scheme 1 is found to be the best of all approaches. It could be due to the fact that using this approach, a good KB of the FLC is evolved by the GA, after carrying the search in a wider space.

3.3.4 Optimization of FLC Using Ant Colony Optimization [37]

In Ant Colony Optimization (ACO) algorithm, an optimization problem is represented in the form of a graph – $G = (C, L)$. Here, C is the set of components of the problem and L indicates the possible connection or transition among the elements C. The solutions are expressed in terms of feasible paths on the graph G, after satisfying a set of constraints. Thus, the Fuzzy Rule Learning Problem (FRLP) using the ACO, is formulated as a combinatorial optimization problem. Its operational mode is composed of two stages: in the first stage, the number and antecedents of the linguistic rules are defined, and a set of consequent candidates is assigned to each rule. In the second stage, a combinatorial search is carried out to find the best consequent of each rule, according to a global error measure over the training set.

The fitness of a solution consists of two parts, namely the *functional fitness* and the *objective fitness*. The functional fitness deals with the functionality of the solutions. That is how good is the solution. The objective fitness is the measure of the quality of the solution, in terms of optimization objectives, such as area, delay, gate count, power consumption, and others.

To apply ACO algorithm to a specific problem, the following steps need to be followed:

- Represent the problem in the form of a graph or a similar easily covered structure,
- Define the way of assigning a heuristic preference to each choice that needs to be taken in each step in order to generate the solution,
- Establish an appropriate way of initializing the pheromone,
- Define the fitness function to be optimized,
- Select an ACO algorithm to determine the optimal solutions.

The Fuzzy Rule Learning Problem (FRLP) aims to obtain the rules combining the labels of the antecedents and to assign a specific consequent to

each antecedent combination. This problem is interpreted as a way of assigning consequents to the rules with respect to an optimality criterion. An ant iteratively goes over each rule and chooses a consequent with a probability that depends on the pheromone trail τ_{ij} and the heuristic information η_{ij}.

3.3.5 Tuning of FLC Using Reinforcement Learning

Fuzzy rules for control can be effectively tuned by means of reinforcement learning. In this approach, the rules with their associated antecedent and consequent fuzzy sets are represented with the help of a fuzzy-neural network. For this an action selection network (ASN) is used. This network provides continuous action value and records the state of the environment and also determines the next action required. Thereafter, the actions are evaluated by means of a critic element (CE), which is a two-layer feed forward action evaluation network (AEN). It predicts the reinforcements associated with different input states and whether or not a failure has occurred. If a failure occurs, it identifies the steps leading to the failure and modifies the fuzzy sets associated with the rules. A gradient descent technique in conjunction with an average reward is used to train both the action selection network (ASN) and the action evaluation network (AEN) over a set of trials. During training, a reward is provided until a failure occurs and then a high value penalty is given.

This approach had been used by Berenji and Khedkar [38], to solve the problem of a cart-pole balancing system.

3.3.6 Optimization of FLC Using Tabu Search

Denna et al. [39] presented an approach for automatic definition of the fuzzy rules based on the Tabu Search (TS) algorithm. To determine the most appropriate rule base for solving the problem, they employed the reactive form of TS algorithm. To apply the Reactive Tabu Search (RTS) algorithm in determining the rules of a fuzzy controller, the consequent of each rule is expressed with a binary string. The learning procedure is shown in Figure 3.11. The learning begins with an initial rule base, chosen randomly at each iteration. Initial states can also be selected by following a uniform distribution over the entire state space. In such conditions, regions of interest are assigned a higher probability during the learning procedure. Performance of the rule base is then evaluated by using an error function $E(\bullet)$, over a set of typical control rules. It is important to mention that during the evaluation of $E(\bullet)$, some rules, with a smaller contribution to the system are not used. This procedure continues until a termination criterion is reached. The termination condition for each execution may be based on the following parameters:

- The number of iterations carried out,
- The Current State of the error function,
- The properties of the solution found.

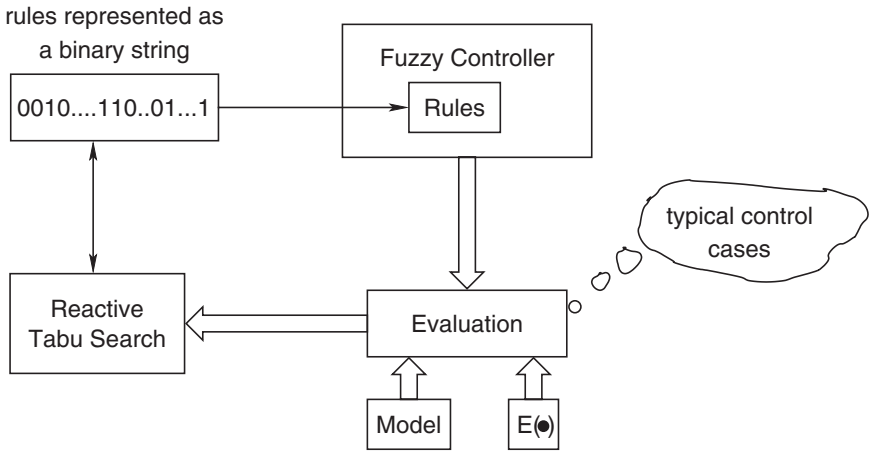

Fig. 3.11. A schematic diagram showing the learning of fuzzy rules using Tabu search [39]

Bagis [40] described a method for the determination of optimum fuzzy membership function distribution used for controlling a reservoir system of dams during floods.

3.3.7 Design of a Fuzzy Controller using the Taguchi Method

The Taguchi Method determines the parameter settings, which maximize the signal to noise (S/N) ratio in each problem by systematically performing the designed experiment. The designed experiment is composed of an inner array and an outer array. The inner array is a designed experiment using the control factors and the outer array consists of the noise factors. To design an FLC using the Taguchi method, control factors are considered as the membership parameters and different system conditions are assumed to be the noise factors. If the inner array is made up of m rows and the outer array contains n rows, then each of the m rows can obtain n performance characteristics. These n data are used to calculate the S/N ratio, for each row of the inner array. The optimal parameter settings are determined by analyzing the S/N ratio data. To check the adequacy of the model, Analysis of Mean (ANOM) and Analysis of Variance (ANOVA) are carried out. Later, a verification experiment is conducted to test the performance of the model. Kim and Rhee [41] utilized the Taguchi method, to design a suitable fuzzy logic controller, in which the following steps were used:

- Identify the performance characteristic to be observed,
- Identify important noise factors and their ranges,
- Identify the control factors and their levels,
- Construct the inner array and the outer array,

– Conduct the designed experiment,
– Analyze the data and determine optimal levels for the control factors,
– Conduct the verification experiment.

3.3.8 Fuzzy Logic Controller Tuned by Simulated Annealing

Simulated Annealing (SA) is one of the most popular non-traditional methods of optimization, in which the cooling process of a molten metal has been artificially modeled. Alfaro and Garcia [42] described a method for development of a fuzzy logic controller applied to path planning and navigation of mobile robots, by using a simulated annealing. Most of the researchers tried to optimize the membership function distributions of the FLC by utilizing the SA. In this approach,the cost function was defined as follows:

$$F = \frac{1}{N} \sum_{k}^{N} (y_k - \hat{y_k})^2 , \tag{3.12}$$

where $k = 1, 2, \ldots, N$. N is the number of learning samples, (x_k, y_k) is the k^{th} learning sample and $\hat{y_k}$ is the output of the fuzzy system corresponding to the input vector x_k. The optimization algorithm tunes the parameters (spread and shape) of membership function distributions. This is in order to minimize the cost function.

Consider the membership functions of the input variables to be Gaussian in nature, as shown below.

$$Gaussian(x; \sigma; c) = e^{-(\frac{x-c}{\sigma})^2} \tag{3.13}$$

where c and σ indicate the Center and Width, respectively, of the membership function distribution. In SA, the following steps are to be considered in order to optimize the Gaussian membership function distribution:

1. Set an Initial Temperature T to a high value and generate initial parameters c_j^i and σ_j^i, randomly and compute the cost function (F_{old}).
2. Generate a set of new parameters c_j^i and σ_j^i and compute the new cost function (F_{new}). Obtain the change in the cost function $\delta = F_{new} - F_{old}$. If $\delta < 0$, memorize the new set of membership functions and proceed until the termination criterion is reached. Otherwise, go to Step 3.
3. If $\delta > 0$ and probability of accepting the new set of membership functions $P(\delta) = exp(-\delta/T) \leq random[0, 1])$, the center and width values are not changed. Now, go to Step 2 by reducing the temperature T to the half of its previous value.
4. Repeat Steps 2 and 3 until an acceptable solution has been found or until a specified number of iterations has been reached.

3.4 Summary

Fuzzy logic controllers have proved their worth and are popular nowadays to solve real-world complex problems. As the performance of an FLC depends on its KB, several attempts had been made to design a suitable KB. Several methods had been tried by various investigators, to solve the problem. There is a chance of further improvement and much further work is necessary.

Both the Linguistic as well as Precise Fuzzy Modeling have been used separately, to solve a variety of problems and some satisfactory results have been obtained. Linguistic fuzzy modeling ensures better interpretability, but precise fuzzy modeling aims to achieve higher accuracy. It is obvious that as interpretability of the fuzzy model increases, its accuracy will decrease and vice-versa. Thus, depending on the physical problem, a particular type of fuzzy modeling is chosen. It is challenging to obtain a proper balance between interpretability and accuracy of a fuzzy model. These two properties are inversely related and it is important to investigate as to whether a pareto-optimal front exists.

References

1. Zadeh, L.A.: Fuzzy sets. Information and Control **8** (1965) 338–353
2. Mamdani, E.H., Assilian, S.: An experiment in linguistic synthesis with a fuzzy logic controller. International Journal on Man-Machine Studies **7** (1975) 1–13
3. Wang, L.X.: Analysis and design of hierarchical fuzzy systems. IEEE Trans. on Fuzzy Systems **7** 5 (1999) 617–624
4. Lee, M.L., Chung, H.Y., Yu, F.M.: Modeling of hierarchical fuzzy systems. Fuzzy Sets and Systems **138** (2003) 343–361
5. Pratihar, D.K., Deb, K., Ghosh, A.: A genetic-fuzzy approach for mobile robot navigation among moving obstacles. International Journal of Approximate Reasoning **20** (1999) 145–172
6. Takagi, T., Sugeno, M.: Fuzzy identification of systems and its application to modeling and control. IEEE Trans. on Systems, Man and Cybernetics **SMC-15** (1985) 116–132
7. Pham, T.D., Valliappan, S.: A least square model for fuzzy rules of inference. Fuzzy Sets and Systems **64** (1994) 207–212
8. Bustince, H., Calderon, M., Mohedano, V.: Some considerations about a least square model for fuzzy rules of inference. Fuzzy Sets and Systems **97** (1998) 315–336
9. Goldberg, D.E.: Genetic algorithms in search, optimization, machine learning. Addison-Wesley, Reading, Mass, USA (1989)
10. Karr, C.: Genetic algorithms for fuzzy controllers. AI Expert (1991) 38–43
11. Thrift, P.: Fuzzy logic synthesis with genetic algorithms, Proc. of Fourth International Conference on Genetic Algorithms. (1991) 509–513
12. Cordon, O.: Gomide, F., Herrera, F., Hoffmann, F., Magdalena, L.: Ten years of genetic-fuzzy systems: current framework and new trends. Fuzzy Sets and Systems **141** (2004) 5–31

13. Hoffman F., Pfister, G.: Evolutionary design of a fuzzy knowledge base for a mobile robot. Intl. Jl. of Approximate Reasoning **17** 4 (1997) 447–469
14. Pham, D.T., Karaboga, D.: Optimum design of fuzzy logic controllers using genetic algorithms. Journal of Syst. Engg. **1** (1991) 114–118
15. Ishibuchi, H., Nakashima, T., Murata, T.: Performance evaluation of fuzzy classifier systems for multidimensional pattern classification problems. IEEE Trans. on Systems Man and Cybernetics **29** (1999) 601–618
16. Cordon, O., DeeJesus, M.J., Herrera, F., Lozano, M.: MOGUL: a methodology to obtain genetic fuzzy rule-based systems under the iterative rule learning approach. Intl. Jl. of Intelligent Systems **14** 11 (1999) 1123–1153
17. Gonzalez, A., Perez, R.: SLAVE: a genetic learning system based on an iterative approach. IEEE Trans. on Fuzzy Systems **7** 2 (2001) 176–191
18. Furuhashi, T., Miyata, Y., Nakaoka, K., Uchikawa, Y.: A new approach to genetic based machine learning and an efficient finding of fuzzy rules-proposal of Nagoya approach. Lecture notes on Artificial Intelligence **101** (1995) 178–189
19. Wang, L., Yen, J.: Extracting fuzzy rules for system modeling using a hybrid of genetic algorithms and Kalman filtering. Fuzzy Sets and Systems **101** (1999) 353–362
20. Farag, W.A., Quintana, V.H., Lambert-Torres, G.: A genetic-based neuro-fuzzy approach for modeling and control of dynamical systems. IEEE Trans. on Neural Networks **9** (1998) 576–767
21. Ishibuchi, H., Nil, M., Murata, T.: Linguistic rule extraction from neural networks and genetic algorithm-based rule selection, Proc. of IEEE Intl. Conf. on Neural Networks. Houston, TX (1997) 2390–2395
22. Abdessemed, F., Benmahammed, K., Monacelli, E.: A fuzzy-based reactive controller for a non-holonomic mobile robot. Robotics and Autonomous Systems **47** (2004) 1–22
23. Yupu, Y., Xiaoming, X., Wengyuan, Z.: Real-time stable self learning FNN controller using genetic algorithm. Fuzzy Sets and Systems **100** (1998) 173–178
24. Angelov, P.P., Buswell, R.A.: Automatic generation of fuzzy rule-based models from data by genetic algorithms. Information Science **50** (2003) 17–31
25. Nandi, A.K.: Pratihar, D.K.: Automatic design of fuzzy logic controller using a genetic algorithm-to predict power requirement and surface finish in grinding. Journal of Materials Processing Technology **148** (2004) 288–300
26. Nawa, N.E., Hashiyama, T., Furuhashi, T., Uchikawa, Y.: A Study on fuzzy rules discovery using pseudo-bacterial genetic algorithm with adaptive operator, Proc. IEEE Int. Conf. Evolutionary Computation. Indianapolis, USA, (1997) 13–16
27. Ishibuchi H., Nakashima, T.: Effect of rule weights in fuzzy rule-based classification systems. IEEE Trans. on Fuzzy Systems **9** 4 (2001) 506–515
28. Ghosh A., Nath, B.: Multi-objective rule mining using genetic algorithms. Information Sciences **163** (2004) 123–133
29. Pratihar, D.K., Deb, K., Ghosh, A.: Optimal path and gait generations simultaneously of a six-legged robot using a GA-Fuzzy approach. Robotics and Autonomous Systems **41** 1 (2002) 1–20
30. Marichal, G.N., Acosta, L., Moreno, L., Mendez, J.A., Rodrigo, J.J., Sigut, M.: Obstacle avoidance for a mobile robot: a neuro-fuzzy approach. Fuzzy Sets and Systems **124** (2001) 171–170
31. Song, K.T., Sheen, L.H.: Heuristic fuzzy-neuro network and its application to reactive navigation of a mobile robot. Fuzzy Sets and Systems **110** (2000) 331–340

32. Li, W., Ma, C., Wahl, F.M.: A Neuro-fuzzy system architecture for behavior based control of a mobile robot in unknown environments. Fuzzy Sets and Systems **87** (1997) 133–140
33. Hui, N.B., Mahendar, V., Pratihar, D.K.: Time-optimal, collision-free navigation of a car-like mobile robot using a neuro-fuzzy approach. Fuzzy Sets and Systems **157** 16 (2006) 2171–2204
34. Jang, J.S.R., Sun, C.T., Mizutani, E.: Neuro-Fuzzy and Soft Computing. Prentice-Hall of India Pvt. Ltd., New Delhi (2002)
35. Hui, N.B., Pratihar, D.K.: Automatic design of fuzzy logic controller using a genetic algorithm for collision-free, time-optimal navigation of a car-like robot. International Journal of Hybrid Intelligent Systems **5** 3 (2005) 161–187
36. Latombe, J.C.: Robot motion planning. Kluwer Academic Publishers (1991)
37. Casillas, J., Cordon, O., Herrera, F.: Learning fuzzy rules using ant colony optimization algorithms, Proc. of 2nd Intl. Workshop on Ant Algorithms. Brussels, Belgium (2000) 13–21
38. Berenji, H., Khedkar, P.: Learning and tuning fuzzy controllers through reinforcements. IEEE Transactions on Neural Networks **3** 5 (1992) 724–740
39. Denna, M., Mauri, G., Zanaboni, A.M.: Learning fuzzy rules with Tabu search– an application to control. IEEE Transactions on Fuzzy Systems **7** 2 (1999) 295–318
40. Bagis, A.: Determining fuzzy membership functions with tabu search – an application to control. Fuzzy Sets and Systems **139** (2003) 209–225
41. Kim, D., Rhee, S.: Design of a robust fuzzy controller for the arc stability of CO_2 welding process using the Taguchi method. IEEE Transactions on Systems, Man and Cybernetics–Part B **32** 2 (2002) 157–162
42. Alfaro, H.M., Garcia, S.G.: Mobile robot path planning and tracking using simulated annealing and fuzzy logic control. Expert Systems with Applications **15** (1998) 421–429