Hence, though hybrid systems have a tremendous potential to solve problems, an inappropriate use of the technology can backfire. For, it is improper to expect that if the individual technologies are good then hybridization of technologies should turn out to be even better. In fact, it is not unlikely for a hybrid technology to exhibit most of the weaknesses of the participating technologies and less of their strengths.

10.1 HYBRID SYSTEMS

Hybrid systems are those for which more than one technology is employed to solve the problem. Hybrid systems have been classified as (Refer Gray and Kilgour, 1997) 1. Sequential Hybrids, 2. Auxiliary Hybrids, and 3. Embedded Hybrids.

10.1.1 Sequential Hybrid Systems

As the name indicates, *sequential hybrid systems* make use of technologies in a pipeline-like fashion. Thus, one technology's output becomes another's input and so on. Figure 10.1 illustrates the schema for a sequential hybrid.

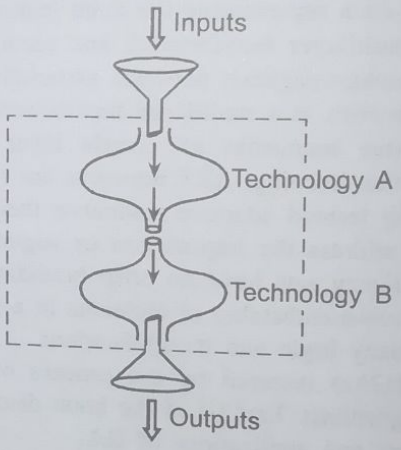

Fig. 10.1 A sequential hybrid system.

This is one of the weakest forms of hybridization since an integrated combination of the technologies is not present.

An example is a GA preprocessor which obtains the optimal parameters for different instances of a problem and hands over the 'preprocessed' data set to an NN for further processing.

10.1.2 Auxiliary Hybrid Systems

In this, one technology calls the other as a "subroutine" to process or manipulate information needed by it. The second technology processes the information provided by the first and hands it over for further use. Figure 10.2 illustrates an auxiliary hybrid system. This type of hybridization though better than sequential hybrids, is considered to be of intermediary level only.

An example is a neuro-genetic system in which an NN employs a GA to optimize its structural parameters, i.e. parameters which defines its architecture.

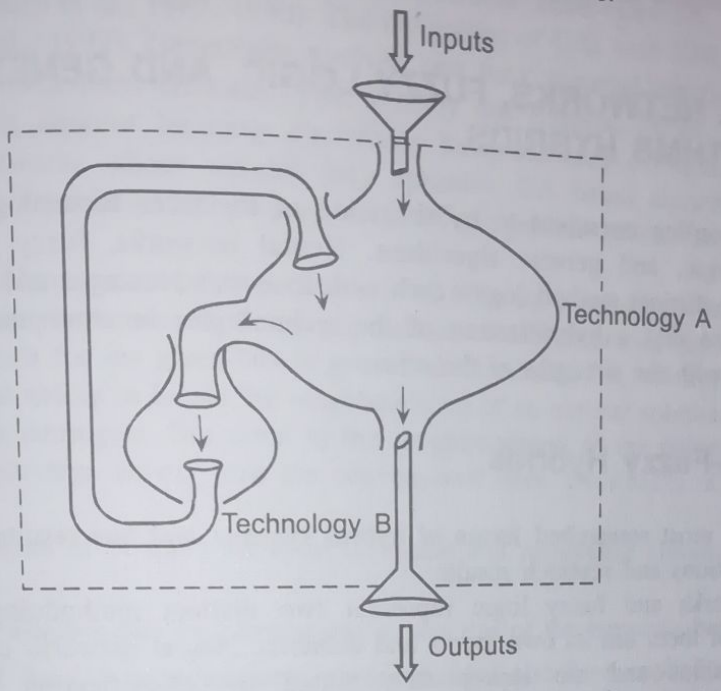

Fig. 10.2 An auxiliary hybrid system.

10.1.3 Embedded Hybrid Systems

In *Embedded hybrid systems*, the technologies participating are integrated in such a manner that they appear intertwined. The fusion is so complete that it would appear that no technology can be used without the others for solving the problem. Figure 10.3 illustrates the schema for an embedded hybrid system. Here, the hybridization is absolute.

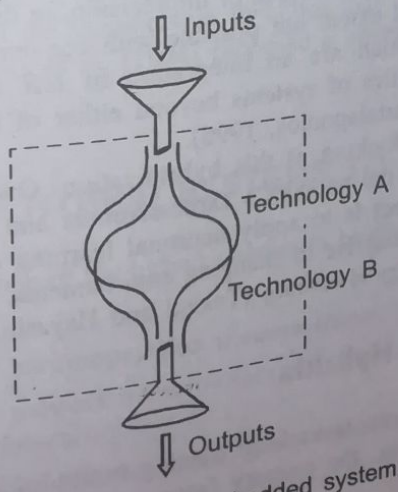

Fig. 10.3 An embedded system.

For example, an NN-FL hybrid system may have an NN which receives fuzzy inputs, processes it and extracts fuzzy outputs as well.

10.2 NEURAL NETWORKS, FUZZY LOGIC, AND GENETIC ALGORITHMS HYBRIDS

In this book, we confine ourselves to hybridization of the three technologies, namely neural networks, fuzzy logic, and genetic algorithms. Neural networks, fuzzy logic, and genetic algorithms are three distinct methodologies each with its own advantages and disadvantages. It is therefore appropriate that a hybridization of the technologies is attempted to overcome the weaknesses of one with the strengths of the other.

10.2.1 Neuro-Fuzzy Hybrids

This is one of the most researched forms of hybrid systems and has resulted in a stupendous quantity of publications and research results.

Neural networks and fuzzy logic represent two distinct methodologies to deal with uncertainty. Each of them has its own merits and demerits. Neural networks can model complex nonlinear relationships and are appropriately suited for classification phenomenon into predetermined classes. On the other hand, the precision of outputs is quite often limited and does not admit zero error but only minimization of least squares errors. Besides, the training time required for an NN can be substantially large. Also, the training data has to be chosen carefully to cover the entire range over which the different variables are expected to change.

Fuzzy logic systems address the imprecision of inputs and outputs directly by defining them using fuzzy sets and allow for a greater flexibility in formulating system descriptions at the appropriate level of detail.

Neural networks and fuzzy logic though different technologies, can be used to accomplish the specification of mathematical relationships among numerous variables in a complex dynamic process, perform mappings with some degree of imprecision, in different ways, and can be used to control nonlinear systems to an extent not possible with conventional linear control systems.

Neuro-Fuzzy systems which are an integration of NN and FL have demonstrated the potential to extend the capabilities of systems beyond either of these technologies when applied individually (Haykin, 1994; Kartalapoulos, 1996).

There are two ways of looking at this hybridization. One is to endow NNs with fuzzy capabilities, thereby increasing the network's expressiveness and flexibility to adapt to uncertain environments. The second aspect is to apply neuronal learning capabilities to fuzzy systems to make the fuzzy systems more adaptive to changing environments. This approach is also known, in the literature, as *NN driven fuzzy reasoning* (Takagi and Hayashi, 1991).

10.2.2 Neuro-Genetic Hybrids

Neural networks can learn various tasks from training examples, classify phenomena, and model nonlinear relationships. However, the primary features that are of concern in the design of the

network are problem specific. Despite the availability of some guidelines, it would be helpful to have a computational procedure in this aspect, especially for the optimum design of an NN. Genetic algorithms have offered themselves as potential candidates for the optimization of parameters of NN (Harp et al., 1989, 1990). The integration of GAs with NNs has turned out to be useful (Schaffer et al., 1992). Genetically evolved nets have reported comparable results against their conventional counterparts (Kitano, 1990; Whitley and Hanson, 1989).

While gradient descent learning algorithms have reported difficulties in learning the topology of the networks whose weights they optimize, GA based algorithms have provided encouraging results especially with regard to face recognition, animation control, and other problems.

Genetic algorithms encode the parameters of NNs as a string of the properties of the network, that is, chromosomes. A large population of chromosomes representing the many possible parameter sets for the given NN is generated. Combined GA-NN technology also known as 'GANN' have the ability to locate the neighbourhood of an optimal solution quicker than other conventional search strategies. But once in the neighbourhood of the optimal solution GANN strategies tend to converge slower than the conventional ones. Drawbacks of GANN algorithms are:

The large amount of memory required to handle and manipulate chromosomes for a given network, and

The question whether this problem scales as the size of the networks becomes large.

Parallel versions of the GA computational paradigm have also been applied on NN (Maniezzo, 1994).

10.2.3 Fuzzy-Genetic Hybrids

Fuzzy systems have been integrated with GAs. Kosko (1992) has shown that fuzzy systems like NNs (feedforward) are universal approximators in the fact that they exhibit the capability to approximate general nonlinear functions to any desired degree of accuracy. The adjustment of system parameters that is called for in the process, so that the system output matches the training data, has been tackled using GAs. Several parameters which a fuzzy system is involved with, namely input/output variables and the membership functions that define the fuzzy systems, have been optimized using GAs. Nomura et al. (1994) proposed a genetic approach to the problem of fuzzy system adaptation.

10.3 PREVIEW OF THE HYBRID SYSTEMS TO BE DISCUSSED

In this section, we present a preview of the hybrid systems to be discussed in Part IV of the book. The systems discussed are:

1. Genetic algorithm based backpropagation network (*Neuro Genetic Hybrid*).
2. Fuzzy backpropagation network (*Neuro–Fuzzy Hybrid with Multilayer Feedforward Network as the host architecture*).
3. Simplified fuzzy ARTMAP (*Neuro-Fuzzy Hybrid with Recurrent Network as the host architecture*).

4. Fuzzy associative memory (*Neuro-Fuzzy Hybrid with Single layer Feedforward or Recurrent Network as its host architecture*).
5. Fuzzy logic controlled genetic algorithms (*Fuzzy-Genetic Hybrid*).

We now briefly review each of these hybrid systems.

10.3.1 Genetic Algorithm based Backpropagation Network

This is a Neuro-Genetic hybrid which makes use of GAs to determine the weights of a multilayer feedforward network with backpropagation learning. Conventional backpropagation networks make use of gradient descent learning to obtain their weights. However, there remains the problem of the network getting stuck in local minimum. On the other hand, the GA based backpropagation though not guaranteed to obtain global optimum solution, has been found to obtain 'acceptably good' solutions 'acceptably quickly'. Though several successful propositions regarding the GA-backpropagation integration have been implemented, the network discussed is a backpropagation network architecture in which the GA makes use of real-coded chromosomes, instead of the conventional binary chromosomes to determine its weights.

The hybrid system has been demonstrated on two problems, namely k-factor design and electrical load forecasting.

10.3.2 Fuzzy-Backpropagation Network

This is a Neuro-Fuzzy hybrid system in which the host is a multilayer feedforward network. Proposed by Lee and Liu (1994), the network maps fuzzy input vectors to crisp outputs making use of backpropagation like learning.

The architecture has been applied to the problems of knowledge-based evaluation and earthquake damage evaluation.

10.3.3 Simplified Fuzzy ARTMAP

Fuzzy ARTMAP (Carpenter et al., 1992) is a neuro-fuzzy hybrid in which the host is a recurrent network with a kind of competitive learning termed adaptive resonance theory by its authors (Carpenter and Grossberg, 1988).

Proposed by Kasuba (1992), simplified fuzzy ARTMAP is a vast simplification of fuzzy ARTMAP. It classifies inputs by its fuzzy set of features and unlike its predecessor has reduced computational overhead and architectural redundancy.

The application of simplified fuzzy ARTMAP to image recognition under noisy and noise-free conditions is elaborated.

10.3.4 Fuzzy Associative Memories

Fuzzy Associative Memory (FAM) is a neuro-fuzzy system with the host architecture as a recurrent or a single layer network. FAMs map fuzzy sets and can encode fuzzy rules. It is on account of this mapping capability that they behave like associative memories.

Fuzzy associative memories make use of graphical method of inference and correlation matrix encoding schemes for inference.

The application of FAM has been demonstrated on two problems, namely balancing an inverted pendulum and fuzzy truck backer-upper system.

10.3.5 Fuzzy Logic Controlled Genetic Algorithms

This is a Fuzzy-Genetic hybrid system applicable on fuzzy optimization problems. The system obtains optimal solution to problems with fuzzy constraints and fuzzy variables.

The hybrid system has been demonstrated on the problems of optimization of structures (civil/machine tool) and obtaining the optimal mix for high performance concrete.

SUMMARY

- Hybrid systems are those which employ integrated technologies to effectively solve problems. Hybrid systems are classified as sequential, auxiliary and embedded hybrids.
- The soft computing techniques of neural networks, fuzzy logic, and genetic algorithms have offered themselves as candidates for a healthy integration or hybridization of technologies for effective problem solving.
- The synergy of the three technologies has led to neuro-fuzzy, neuro-genetic, and fuzzy-genetic hybrids. In this book, five hybrid systems, namely genetic algorithm based back-propagation network (*Neuro-Genetic hybrid*), fuzzy backpropagation network, simplified fuzzy ARTMAP and fuzzy associative memories (*Neuro-Fuzzy hybrids*) and fuzzy logic based genetic algorithms (*Fuzzy-Genetic hybrid*), are presented.

REFERENCES

Carpenter, G.A. and S. Grossberg (1988), The Art of Adaptive Pattern Recognition by a Self-Organizing Neural Network, *IEEE Computer Magazine,* Vol. 21, No. 3, pp. 77–88.

Carpenter, G.A., S. Grossberg, Markuzon Natalya, J.H. Reynolds and D.B. Rosen (1992), Fuzzy ARTMAP: A Neural Network Architecture for Incremental Supervised Learning of Analog Multidimensional Maps, *IEEE Trans on Neural Networks,* Vol. 3, No. 5, pp. 698–713.

Gray, Andrew and Richard Kilgour (1997), Hybrid Systems FAQ, Version Draft 1.00.

Harp, S., T. Samad, and A. Guha (1989), Towards the Genetic Synthesis of Neural Networks, *Proc of the Third International Conf on Genetic Algorithms,* Morgan Kaufmann, San Mateo, CA.

Harp, S., T. Samad, and A. Guha (1990), Designing Application Specific Neural Networks using the Genetic Algorithm, *Advances in Neural Information Processing Systems 2* D.S.Touretzky, Ed., Morgan Kaufmann, San Mateo, CA.

Haykin, S. (1994), Neural Networks: A Comprehensive Foundation, IEEE Computer Society Press Macmillan NY.

To start the algorithm, the initial population is created randomly. If the mix consists of five constituents and any individual in the population represents five bits, each constituent contains 25 bits. The objective function and constraints are calculated for every individual and genetic algorithm is applied as explained in Chapter 9 and the optimal mix is arrived at as

Cement = 1
Sand = 1.36
Coarse aggregate = 2.95
Water cement ratio = 0.4190
Silica fume = 21.3%
Superplasticizer = 4.03

The above mix gives the strength of 129 MPa and a slump of 125 mm and the cost of concrete mix/unit volume is given as Rs 3.50.

15.5 GA IN FUZZY LOGIC CONTROLLER DESIGN

For optimal control problems, fuzzy logic techniques are primarily applied since quick control strategy is needed and imprecise and qualitative definition of action plans are available. While designing an optimal fuzzy controller, one has to look for two primary activities.

1. Find optimal membership functions for control and action variable.
2. Find an optimal set of rules between control and action variable.

In the above two cases, GAs have been suitably used. Figure 15.5 shows typical membership functions for a variable (control or action) having three choices low, medium, and high. Since the maximum membership function value of these choices is always one, the abscissae marked by X_i are usually chosen by the user. These abscissae can be treated as variables in GA and an optimization problem can be posed to find these variables for minimizing or maximizing a control strategy such as time of variable operation, product quality, and others. Consider an example given by Deb (1999) to illustrate how GA can be uniquely applied to the above problem. Let us assume that there are two control variables (temperature and humidity) and there are three operations for each, low, medium, and high. There is one action variable (water jet flow rate) which also takes three choices low, medium, and high. Considering individual effect of each control variable separately, there are total of $4 \times 4 - 1 = 15$ combinations of control variables possible as shown in Table 15.1.

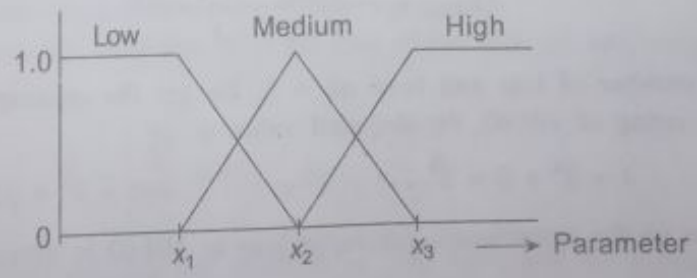

Fig. 15.5 Fuzzy membership function and typical variables used for optimal design.

Table 15.1 Action variable for a string representing a fuzzy rule base

Humidity	Temperature			
	Low	Medium	High	No action
Low	High	Medium	–	Medium
Medium	Low	–	Medium	Medium
High	Medium	High	–	–
No action	–	–	High	–

Thus, finding an optimal rule base is equivalent to finding the four operations (fourth operation—not known) or the action variable for each combination of the control variables. A GA with a string length of 15 and with a ternary coding can be used to represent the rule base for this problem. Considering real values 1 to 4 for representation as 1—Low, 2—Medium, 3—High, 4—No action, thereby signifying the absence of the corresponding combination of action variables in the rule base. Table 15.1 shows the rule base and this can be represented by the following string.

3 1 2 4 2 4 3 4 4 2 4 3 2 2 4

Although this rule base may not be the optimal one, GA can process a population of such rule bases and finally find the optimal rule base. Once the rows present in the rule base are determined from the string user, defined fixed membership functions can be used to simulate the underlying process. The objective function can be evaluated and the usual single point cross over and a mutation operator (one allele mutating to one of three other alleles) can be used with this coding. GA can find the optimal number of rules to solve the problem. If one wants to use binary strings instead of ternary strings and two bits are used to represent each of four operations, a total of 30 bits is necessary to represent a rule base. This kind of technique has been used to design fuzzy logic controller for mobile robot navigation among dynamic obstacles (Deb et al., 1998). Both optimum membership function determination and optimal rule base tasks can be achieved simultaneously by using a concatenation of two codings mentioned above. A part of the overall string will represent the abscissae of the control variables and the rest of the string will represent the rules present in the rule base. Fitness is calculated as explained above.

15.6 FUZZY LOGIC CONTROLLER

Soh and Yang (1996), Yang and Soh (2000) have investigated fuzzy based structural optimization using GA. Using the approach of Soh and Yang as seen in Section 15.4, a fuzzy logic controller (FLC) a rule based system incorporating the flexibility of human decision making is used for fuzzy structural optimization. The fuzzy functions are intended to represent a human expert's conception of the linguistic terms, thus giving an approximation of the confidence with which precise numeric value is described by a linguistic label.

15.6.1 Components of Fuzzy Logic Controller (FLC)

As shown in Fig. 15.6, fuzzy logic controller process is divided into three stages.

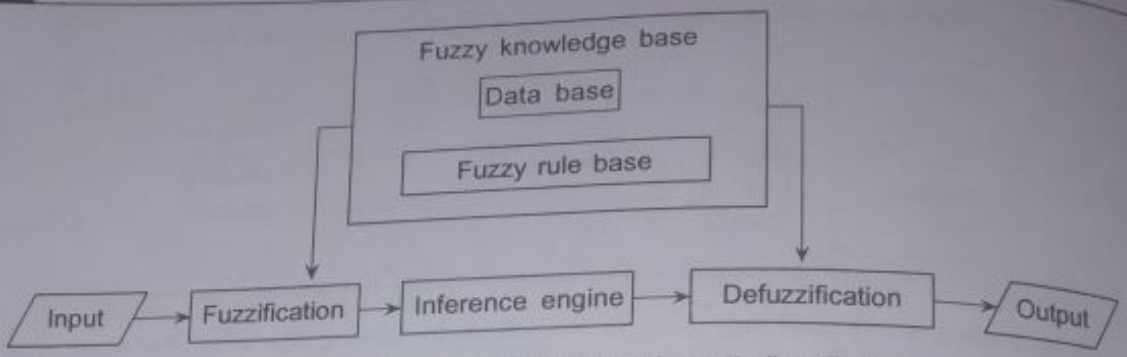

Fig. 15.6 Framework of fuzzy logic controller (FLC).

(a) Fuzzificataion—To calculate fuzzy input (i.e. to evaluate the input variables with respect to corresponding linguistic terms in the condition side).

(b) Fuzzy inference—To calculate fuzzy output (i.e. to evaluate the activation strength of every rule base and combine their action sides).

(c) Defuzzification—To calculate the actual output (i.e. to convert the fuzzy output into precise numerical value).

15.6.2 Fuzzy IF-THEN Rules

Fuzzy rules take the form IF (conditions) and THEN (actions), where conditions and actions are linguistic variables, respectively. An example of Fuzzy IF-THEN rule is given below.

Increase interest rates slightly if unemployment is low and inflation is moderate.

Increase interest rates sharply if unemployment is low and inflation is moderate but rising sharply.

Decrease interest rates slightly if unemployment is low but increasing and inflation rate is low and stable.

The primary format of IF-THEN rules is given in Fig. 15.7

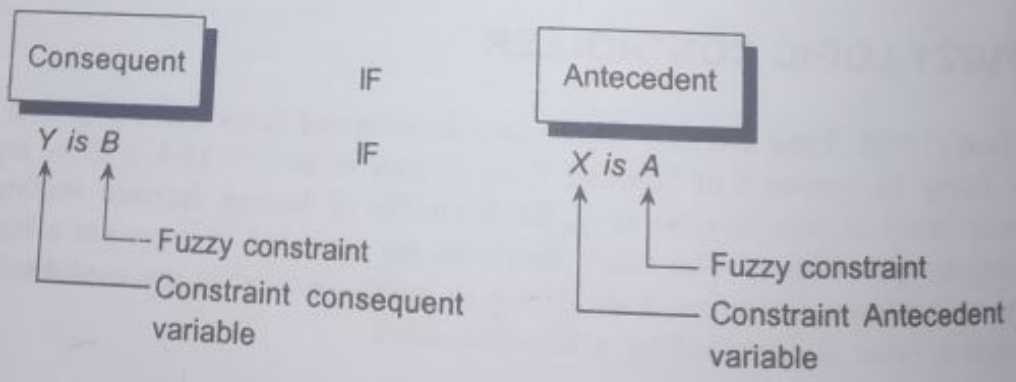

Fig. 15.7 Format of IF-THEN rule.

Example

Volume is small if pressure is high.

Usually in civil engineering, most of these specifications in codes and the functional requirements set by the users must be given in natural language to describe the expert's knowledge of design modifications.

In usual structural optimization, the stress constraint written as

$$\sigma_i^L \le \sigma_i \le \sigma_i^U \tag{15.11}$$

is a member stress constraint, where σ_i is the stress in member 'i' and σ_i^L and σ_i^U are the lower and upper bounds of the allowable stress. But in case of fuzzy optimization, the Eq. (15.11) is replaced as

$$\tilde{\sigma}_i^L - \tilde{\sigma}_i^{Le} \tilde{\le} \sigma_i \tilde{\le} \tilde{\sigma}_i^U + \tilde{\sigma}_i^{Ue} \tag{15.12}$$

with relevant α membership degree similar to Fig. 15.2 as in Fig. 15.8.

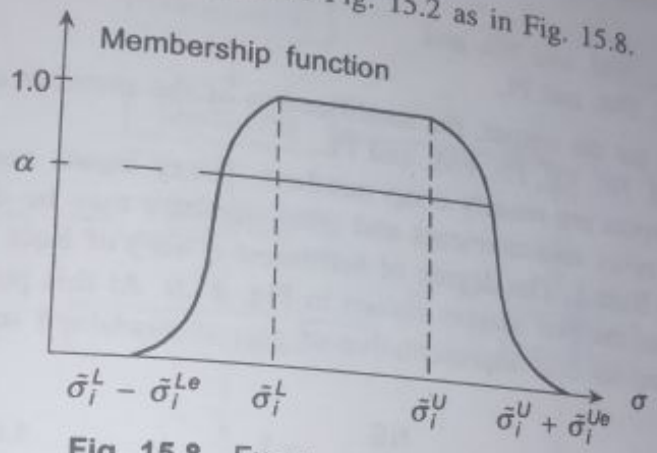

Fig. 15.8 Fuzzy constraint for stress.

Here, the symbol $\tilde{\le}$ means fuzzy variable operator and α represents a series of linguistic variables that means "very very small", "small", "medium", "large", "very large", and "very very large" and so forth. According to Zadeh (1987), the following seven fuzzy variables are usually in the study of structural optimization as negative large (NL), negative medium (NM), negative small (NS), zero (ZE), positive small (PS), positive medium (PM), and positive large (PL). They are defined by the membership functions as shown in Fig. 15.9. For the convergence of implementation, seven fuzzy variables are assigned seven integer reference numbers, namely -3, -2, -1, 0, 1, 2, 3 respectively.

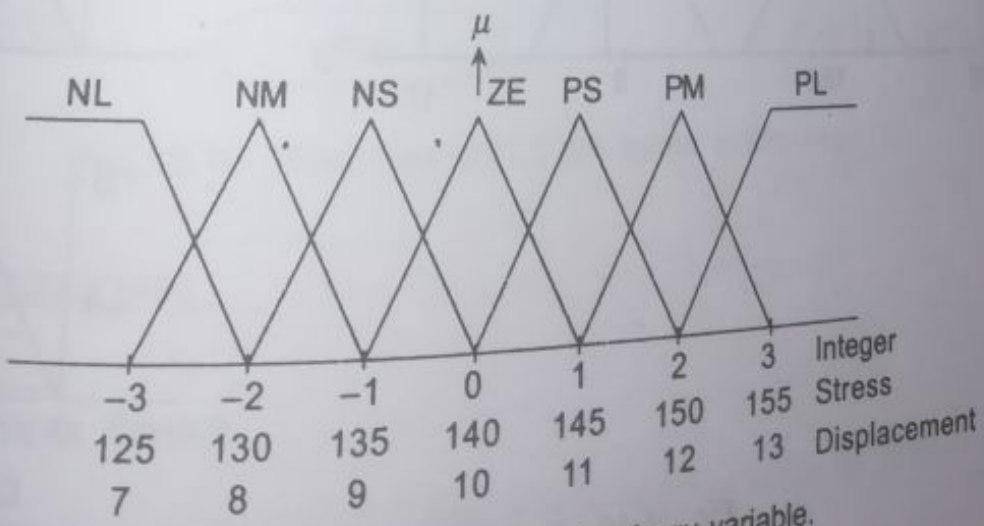

Fig. 15.9 Membership function for fuzzy variable.

If the allowable stress is 140 MPa and the tolerance for each unit is 5Mpa, X axis is also given in terms of stress values. Similarly, one can define seven fuzzy membership functions for displacements as well as for any other variable.

As explained before, heuristic fuzzy rules can be written as

Rule 1: IF the maximum of violated member stress constraints is PS and all the displacement constraints are inactive
THEN the change of the corresponding member cross sectional area is PS.

Rule 2: IF all constraints are inactive and the minimum of member stress constraints is NL
THEN the change of the corresponding member cross-sectional area is NS.

As an input, the constraint C_{ij} is usually classified as

1. active for ZE,
2. inactive for NL, NM, and NS, and
3. violated for PS, PM, and PL.

On the other hand for the output, the modification of the member cross-sectional areas has the possibilities, NL, NM, NS, ZE, PS, PM, and PL.

Fuzzy controller inputs are usually crisp numbers. Fuzzy inputs may also be considered in the case of uncertain or noisy measurements and crisp numbers may be defuzzified. Figure 15.10 shows the application of Rule 1. The degree of fulfilment (DOF) of Rule 1 is 0.4. The total fuzzy output μ_{out} is the union of the two outputs shown in Fig. 15.9. At this point we need to defuzzify and obtain the crisp value for ΔA^* representative of μ_{out} as explained in Chapters 6 and 7.

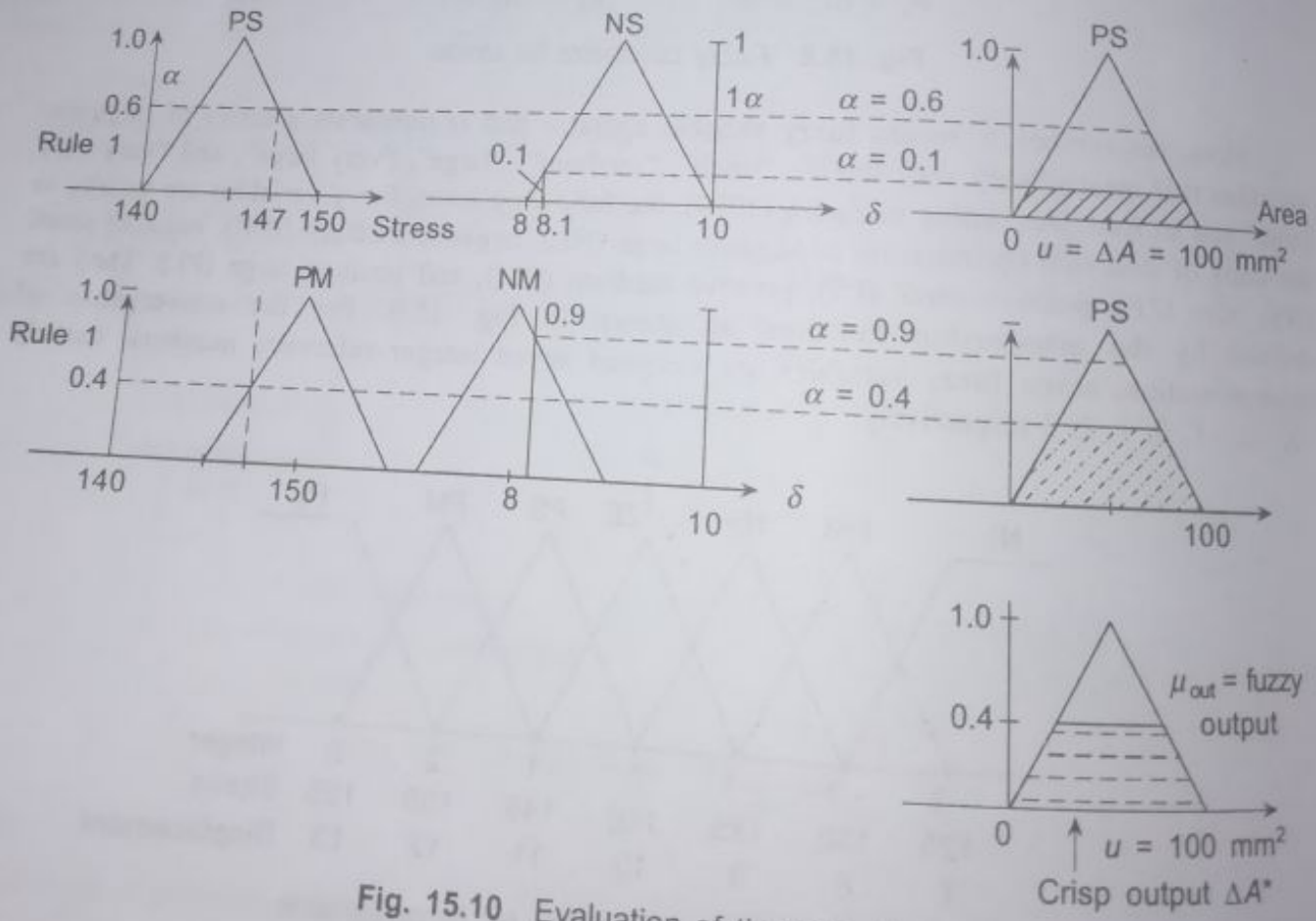

Fig. 15.10 Evaluation of the Rule 1.

15.7 FLC-GA BASED STRUCTURAL OPTIMIZATION

First, coding scheme is to be defined and the initial population is produced. The computation with genetic operators is used to evaluate fitness function with respect to the objective function. Figure 15.11 shows the FLC-GA based optimization procedure. Using FLC we can get the expert's experience in fuzzy rule base of FLC. Hence, the search can react optimum solution quickly. As a result, computing time is very much reduced. The predefined probability and fuzzy representation of design constraints causes FLC to reduce the risk of premature problem solution caused by improper rule.

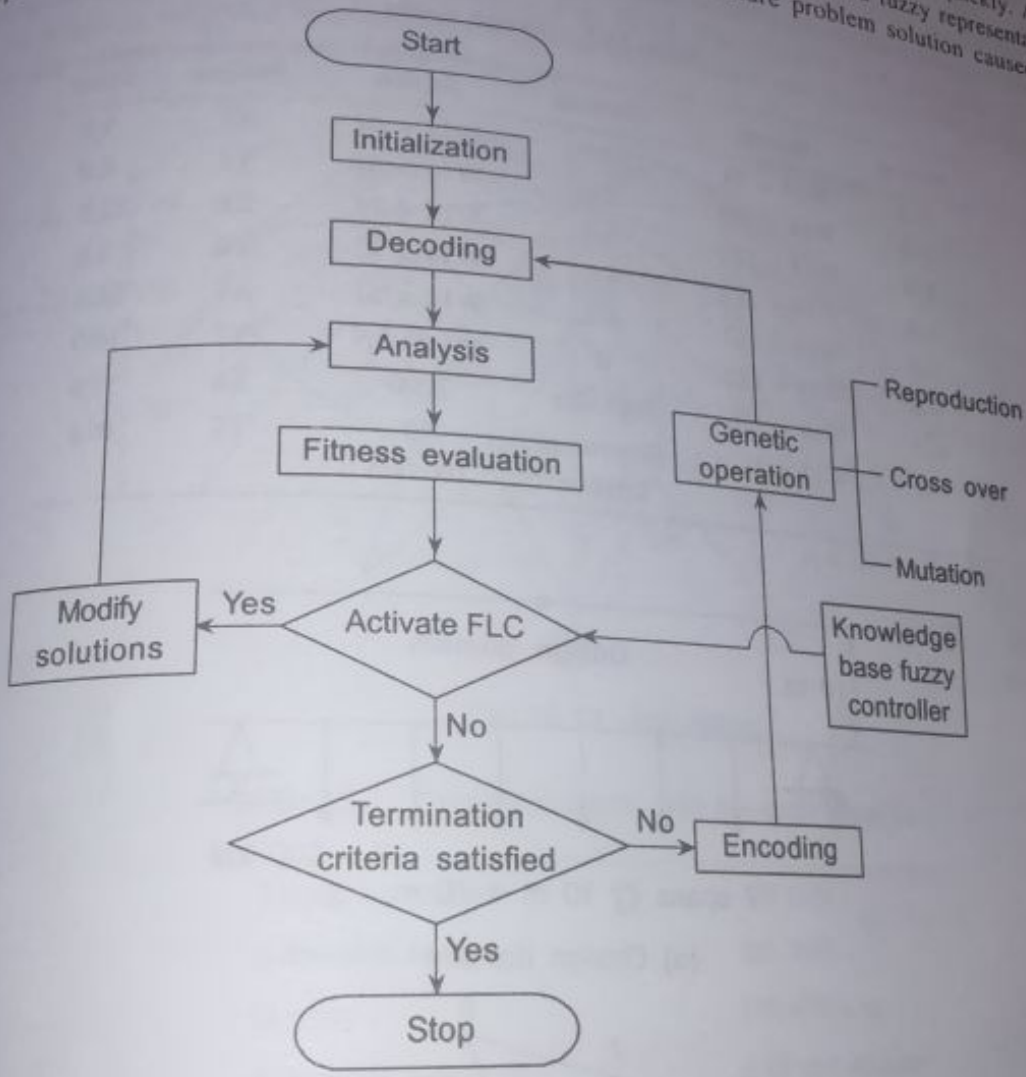

Fig. 15.11 Flow chart of FLC GA based optimization.

15.8 APPLICATIONS

15.8.1 Optimum Truss

Yang and Soh (2000) have found out the optimum truss structure within the given design domain