On the Synthesis of Fuzzy Neural Systems

A Thesis Submitted to
the Department of Electronic Engineering
The Chinese University of Hong Kong

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

By

Chung, Fu Lai

May, 1995

QA
76.87
C472
1995
ult

To My Wife, Jenny

ACKNOWLEDGEMENT

It is a pleasure to express my gratitude to my supervisor, Dr. Tong Lee, who spent so much effort on me throughout this research. I am deeply indebted to him for his constant guidance, valuable advice, and critical comment contributed immeasurably to the research effort and completion of the thesis.

I am also grateful to my lovely brothers in Christ who give me encouragement throughout the study, namely, Tommy Chau, Winky Lai, Po Li, Anthony Ng, and Amos Chan. Thanks also go to Patrick Tang, Daniel Leung, Thomas Choi, Terence Chan, Alex Lee, T.L. Cheng, Shirley Chang, Michael Tso and Keith Chow with whom I have had many enlightening discussions on various research issues.

Last, but not the least, I would like to express my deepest appreciation to my wife Jenny who has suffered so many days and nights of "home alone". Her constant support and encouragement make my research easier and more enjoyable. She deserves the best gift I could ever give to her — this thesis.

ABSTRACT

During the past several years, fuzzy neural system (FNS), an integration of fuzzy systems and neural networks, has emerged as an active area of research. Its goal is to combine their individual strength and eliminate their weakness such that better results can be obtained. Three areas that are essential to the development of FNS have been chosen to study in this research. They are fuzzification of competitive learning (CL) networks, capacity analysis of fuzzy associate memory (FAM) and fuzzy relational neural system (FRNS) models, and development of a FRNS identification algorithm capable for both structure determination and parameter estimation.

The effectiveness of the FNS approach to overcome shortcomings of conventional learning algorithm has been demonstrated through fuzzifying the competitive learning methodologies. By fuzzification of the concept *win* in the competition process, a fuzzy competitive learning paradigm is developed and three CL algorithms have been fuzzified accordingly. The main idea is to let every neuron learn in a graded manner via the fuzzy win membership which is formulated as a non-increasing function of the closeness information. As demonstrated by the experimental results, the proposed FCL networks are superior to the crisp counterparts in avoiding neuron underutilization, training and testing performances, particularly in overlapping data sets, and interpretation of the trained networks.

The second area of research was motivated by the poor understanding on the storage property of two existing FNS models, FAM and FRNS. It has been assumed that the capacity of a FAM matrix is one rule only while that of a fuzzy relation is bounded by a pairwisely disjoint condition. Rigorous analysis of their storage capacity showed that the capacity of a FAM matrix can be more than one rule. By generalizing the usual max-min composition of the FAM model to a max-bounded-product one, the capacity can be enhanced to store up a set of semi-overlapped fuzzy rules, which is typical in general applications. By performing similar analysis of the FRNS model, it can be shown that the current understanding on the storage capacity is too conservative; again the model is, too capable of storing a set of semi-overlapped fuzzy rules. The impact of this study is that the storage requirement of the FAM model can be substantially reduced and both models consisting of one matrix/relation only are good enough in most applications.

A common application of FRNS is to model nonlinear dynamic system and a numerical algorithm is usually called for identification purposes. While the system identification process should determine both the structure and the parameters of FRNS, most works are devoted to parameter estimation with the structure determined by a trial-and-error process and only little work has attempted to determine the structure from the training data. In view of the success of evolutionary computation methods in determining the topology of neural networks, an evolutionary identification (EVIDENT) algorithm for FRNS is developed. Through the experiments with various benchmarking data sets, the effectiveness of EVIDENT in both structure and parameter identifications is demonstrated.

TABLE OF CONTENTS

Chapter 1

Introduction

1.1 Integration of Fuzzy Systems and Neural Networks

Over the last decade or so, significant advances have been made in two distinct fields of interest : fuzzy systems and neural networks. Fuzzy system (FS) goes by names like fuzzy logic system, fuzzy logic controller, fuzzy control system, and fuzzy model which make it tightly-coupled with control applications. In fact, fuzzy control has been one of the most active and fruitful areas of research in the application of fuzzy set theory [71]. Its applications have been found with impressive success in areas ranging from industry process control to consumer electronic product design [104]. Unlike classical control methodology which requires a precise mathematical model to describe the system, the fuzzy approach utilizes a set of linguistic rules in the form of IF-THEN logical statement, having its antecedents and consequents encoded with fuzzy concepts such as *fairly small*, *medium high*, etc. Thus human-oriented control rules which appear very useful in controlling complex and ill-posed processes can be incorporated into the fuzzy systems conveniently. Thus, fuzzy systems are rule-based and are usually viewed as early examples of expert systems.

The operations of fuzzy systems are usually described by a fuzzy inference process consisting of the following steps.

1. *Fuzzification* : Compare the crisp inputs with the antecedent fuzzy terms of the rules to obtain the membership values of each fuzzy term.
2. *Fuzzy Matching* : Combine the membership values rulewisely using a specific *t*-norm operator [35], usually minimum or algebraic product, to get the degree of match for each rule.
3. *Fuzzy Aggregation* : Clip the consequent fuzzy term of each rule to the degree of match and aggregate the qualified consequents using a *s*-norm operator [35], e.g., maximum, to form the overall output fuzzy set.
4. *Defuzzification* : Produce a crisp output that best represents the overall output fuzzy set.

Fuzzification and defuzzification are employed because in most practical applications crisp inputs and outputs are used. Such a fuzzy inference mechanism has been devised to model human's knowledge representation scheme, i.e., fuzzy terms, and logical reasoning via fuzzy matching and aggregation. It processes information in a parallel and distributed manner and allows the fuzzy systems to be interpreted linguistically.

Neural networks (NN) have been studied for many years in the hope of achieving human-like perceptual and recognition performance. The progress has been slow until the last decade, new network models and learning algorithms have been developed and the field becomes active again. Neural networks also go by many names, e.g., connectionist systems, neural information processing systems, and parallel distributed processing systems. They typically consist of many simple computational elements called *neurons* or *nodes* that are highly interconnected with each other as biological neural systems do. Input patterns are processed in a parallel and distributed fashion where each node computes its outputs according to a certain nonlinear activation function. There exists a number of ways to classify NN models. According to their learning mechanisms, NNs can be roughly categorized as supervised, self-organizing, and memory models. Among the supervised models, Rumelhart *et al.*'s multilayer feedforward network [101], also known as back-propagation network, may well be recognized as the most widely-used one. It is characterized by a layered feedforward architecture where layers of neurons are connected hierarchically. It maps sets of input patterns into sets of output patterns and the desired input-output mapping is usually determined by external, supervised adjustment of the network's interconnecting *links* or *weights* via the well-known back-propagation algorithm [101]. Such networks have demonstrated the ability to learn from examples and adapt to changes in the environment. On the other hand, self-organizing NN models such as the Kohonen map [63] and adaptive resonance theory (ART) one [46,47] attempt to "cluster" or average portions of the training data into representative groups in an unsupervised manner. Neurons in the network compete with each other to be specific detectors of different patterns and hence the involved learning algorithms are usually known as competitive learning. These models have appeared to be very useful in problems like feature detection [102], vector quantization [1] and pattern classification [64] where the labels of training patterns are unknown. Memorizing and learning are intricately connected [49]. In the first place, an activity pattern must be stored in memory through a learning or *encoding* process. When a particular activity pattern is learned, it can be recalled later when the network is presented with a stimulus that may be the original version, a noisy version or an incomplete description of the key pattern. As such, models of this category are usually termed as associative memories and the Hopfield model [52] and Kosko's BAM model [66] are undoubtedly the most well-known ones. As all these NN models process data rather than symbols or rules, interpretation of the networks is still a non-trivial task. Nevertheless, NNs have evolved a new paradigm of computation that contrasts sharply with the traditional sequential and programming approach.

Table 1.1 Differences of Fuzzy Systems and Neural Networks

Fuzzy Systems	***Neural Networks***
❒ tightly coupled with control applications	❒ tightly coupled with pattern recognition
❒ rule based systems	❒ pattern based systems
❒ high level information processing tools	❒ low level information processing tools
❒ imitating human's logical inference and knowledge representation abilities	❒ imitating human's brain structure and learning abilities
❒ no learning at all	❒ can learn from experience
❒ easy to interpret	❒ not easy to interpret

Table 1.2 Similarities of Fuzzy Systems and Neural Networks

❍ parallel and distributed processing of information
❍ model-free estimator
❍ universal approximator
❍ generalize to unseen information
❍ fault-tolerance

Although FS and NN are quite different from each other as discussed and summarized in Table 1.1, there exists a lot of similarities between them equally. As listed in Table 1.2, both models process information in a parallel and distributed manner [75]. Both are model-free estimators, i.e., they estimate a function without a mathematical model of how outputs depend on inputs [67]. Both are universal approximators, i.e., they are capable of approximating any nonlinear function over a compact set to any degree of accuracy [112]. Both generalize to unseen information and have good fault-tolerance capability [65]. In summary, fuzzy systems are generally advantageous in knowledge representation and logical reasoning under cognitive uncertainty while neural networks are distinguished for their learnability, parallelism, and adaptability to changing environments. Consequently, they are indeed complementary techniques to realize intelligent systems for various applications. In fact, the integration of these two fields has given birth to a new class of intelligent system models called fuzzy neural system (FNS). It is generally defined as models synthesized by the theoretical, conceptual, and/or functional components of fuzzy systems and neural networks. Its goal is to combine their strengths and eliminate their individual weaknesses in different problem domains.

Many FNS models have been proposed in the last few years, for instance, see the collected papers in [7]. To name a few, the fuzzy associative memory (FAM) [67] is one of the earliest attempts to use layered network architecture to implement fuzzy systems. The model is characterized by a two-layer heteroassociative feedforward network that stores discrete fuzzy rule pattern pair in its weight matrix using correlation-minimum or correlation-product encoding method. It has been successfully applied to problems like backing up a truck-and-trailer [65], target tracking [67, Ch.11], and voice cell control in asynchronous transfer mode (ATM) networks [81] where distinctive features like modularity, robustness, and adaptability have been demonstrated. The other well-known example is the fuzzy ART model

[11]. It replaces the crisp intersection operator used originally in the choice, search, and learning laws of the ART 1 model [46] by a fuzzy one, i.e., minimum. Such a modification has led to an enhanced version of ART 1. As a result, fuzzy ART can learn stable categories in response to either analog or binary input patterns rather than just the binary ones.

Owing to its mathematical richness and intimate connections with applications, fuzzy relational equation has been one of the most important areas in fuzzy set theory [32]. In [95,97], it has been shown that the equation is structurally similar to a two-layer feedforward network and hence can be implemented by parallel hardware and incorporated with learning. Furthermore, the resultant model, termed, here, by a "fuzzy relational neural system" (FRNS), can be interpreted logically [95]. There exist many other FNS models exploiting different integration methodologies and a systematic overview will be given in Chapter 3. In brief, they can be divided into two major categories, namely, fuzzification of neural networks and layered network implementation of fuzzy systems. The former has been shown to be effective in enhancing the performance and information handling capability of conventional neural networks while the latter is generally advantageous in capturing and articulating fuzzy IF-THEN rules from numerical input-output training data and implementing them in an efficient manner.

1.2 Objectives of the Research

Interests in the synthesis of fuzzy neural systems have been diverse. Three of them have been identified in this study, namely, fuzzification of competitive learning algorithms, storage capacity analysis of FAM and FRNS models, and development of a FRNS identification algorithm capable for both structure determination and parameter estimation. They are essential to the development of FNS because the first corresponds to demonstrating the advantages of the new FNS's over the conventional NN models, the second contributes to the consolidation of the theoretical understandings of two existing FNS models, and the third addresses an important but not-yet-explored learning problem in FNS.

1.2.1 Fuzzification of Competitive Learning Algorithms

Competitive learning has been frequently used in self-organizing NN models. Its basic idea is to let the neurons compete according to some sort of distance metric for the current input pattern $\vec{x}_k$. If the *j*th neuron wins, its parametric vector $\vec{m}_j$ is updated additively by some proportion of the difference vector $\vec{x}_k - \vec{m}_j$. In other words, the winner takes all the responsibility for learning the current input pattern. This learning mechanism however suffers from two major shortcomings. One is the neuron underutilization problem [1,46,102]. A frequently cited example is that neurons with parametric vectors far away from the training vectors never win, and therefore never learn. Thus the available network resources are not fully utilized. The other shortcoming of competitive learning is that information concerning the closeness of input patterns and competing neurons is not used during the winner-take-all training process but it contains rich information in enhancing the learning process [28]. It is expected that the quality of cluster centroids found by competitive learning will be improved if both shortcomings can be overcome [58]. Thus, the objective of this part of the research is to develop an effective FNS solution by fuzzifying the learning rule such that these two shortcomings can be addressed.

1.2.2 Capacity Analysis of FAM and FRNS Models

Central to a fuzzy system is a set of fuzzy IF-THEN rules that relates the antecedent fuzzy sets to the consequent fuzzy sets. A fuzzy system or a fuzzy neural system is therefore expected to encode the rules such that the set-to-set mapping defined by them can be realized. Despite its effectiveness using layered network architecture, the FAM model suffers from very low storage capacity, i.e., one rule pattern pair per weight matrix [67]. It results in consuming a large amount of hardware and computations (if serial machine is used) when the number of rules is large. Hence, the FAM model is limited to applications with small rulebases.

The fuzzy relational equation, the mathematical framework of the FRNS model, is another tool for encoding fuzzy rules. Each equation corresponds to a single rule and a system of equations results when there is a set of rules. If the system of equations can be solved, then only one fuzzy relation is enough for implementing the set of rules properly. In that case, the storage capacity of the FRNS would be very high. Unfortunately, according to the current understandings [33,44,92,97], this requires the input fuzzy sets to be normal and pairwise disjoint which is almost impossible to satisfy in practical applications. In this part of the research, the storage capacity of these two FNS models is re-examined rigorously and high capacity encoding of fuzzy rules is sought where possible. The results would lead to a better theoretical understanding of the models and a reduction of their storage requirements.

1.2.3 Structure and Parameter Identifications of FRNS

Both structure determination and parameter estimation are required when the FRNS model is applied to nonlinear dynamic system identification. While parameter identification is referred to the learning of the fuzzy relation elements, structure identification is the determination of the system orders, time delays, and composition operators of the systems being modelled [91]. Little work has been done in the later area. Existing identification algorithms usually fix the system structure beforehand and identify the fuzzy relation elements subsequently. If the performance is not acceptable, the identification process is repeated with different system structures. Hence, the system structure is determined by a trial-and-error process. In view of such a deficiency, a FRNS identification algorithm capable for both structure determination and parameter estimation is developed in the last part of the research. The new algorithm would identify the full systems being modelled without the intervention of human beings.

1.3 Outline of the Thesis

The thesis is made up of eight chapters. In Chapter 2, we first review the conceptual and mathematical background of fuzzy set theory. A brief description of the basic components of a fuzzy system is then followed.

In Chapter 3, a categorization of existing fuzzy neural system models is presented. We focus on two major categories : i) fuzzification of neural networks; and ii) layered network implementation of fuzzy systems. Representative models of each category are reviewed and their advantages and integration methodologies are highlighted. The significance of the three areas of research is then elaborated.

With the background provided in Chapters 1-3, the achievements of this research are reported in the subsequent chapters. In Chapter 4, the fuzzification of three existing competitive learning algorithms is described. The superiority of the new algorithms in avoiding neuron underutilization, classification and their generalization performance is then demonstrated through various data sets. Issues in controlling the fuzziness of the algorithms and interpreting the trained networks are also addressed.

Chapters 5 & 6 are devoted to report the storage capacity analysis of the FAM and FRNS models. In Chapter 5, after a brief review of the FAM model, a perfect recall theorem for encoding multiple rules instead of a single rule by a single weight matrix is derived. We then extend the results to higher capacity encoding and state the capacity. The chapter ends with a discussion on rule modifications and inference performance.

In Chapter 6, backgrounds on the subject of fuzzy relational equations and their mapping to layered network architecture are introduced first. We demonstrate that the usual understandings of the solvable conditions of a set of max-t equations have been too conservative. Furthermore, the results are extended to the min-s type equations and applied to existing approximate resolution methods. The theorems are then elaborated from a system capacity perspective and the inference performance is reported finally.

In Chapter 7, an evolutionary identification algorithm called EVIDENT for the FRNS model as applied to nonlinear dynamic system modelling is developed. Its effectiveness in identifying nearly optimal system structure and a better parametric solution, as compared with existing identification algorithms, is demonstrated through three benchmarking data sets. The last chapter concludes the thesis by summarizing the contributions of the present work, and gives suggestions on potential areas for further investigations.

Chapter 2

A Fuzzy System Primer

In this chapter, we briefly review some of the basic concepts and theories of fuzzy sets and systems which will be referred in the rest of the thesis. A more detailed discussion can be found in [34,60,123].

2.1 Basic Concepts of Fuzzy Sets

In classical set theory, a set A of a universe of discourse **U** is characterized by an indicator function

$$I_A(u \in \mathbf{U}) = \begin{cases} 1 & \text{if } u \in A \\ 0 & \text{if } u \notin A \end{cases} \tag{2.1}$$

However, such a crisp approach to model physical phenomena in the real world is not always adequate. It is particularly so when the definition of a concept or the meaning of a word is encountered. For examples, what constitutes a "tall person". In view of this, Zadeh [118] introduced the idea of fuzzy sets, where the transition between full membership and no membership is gradual rather than abrupt. A formal definition goes as follows.

Definition 2.1

Fuzzy Set : Let **U** be a collection of objects and be called the universe of discourse. It can be discrete, e.g., $\mathbf{U} = \{u_1, u_2, \cdots, u_l\}$ or continuous, e.g., $\mathbf{U} = \Re$. A fuzzy set A in **U** is characterized by a membership function

$$\mu_A : \mathbf{U} \rightarrow [0,1] \tag{2.2}$$

which associates with each element of **U** a real number in [0,1]. Thus, membership of a (fuzzy) set is no longer either/or, but is a matter of degree. When **U** is continuous, the fuzzy set A is usually written concisely as

$$A = \int_{u \in \mathrm{U}} \mu_A(u) / u \tag{2.3}$$

to denote A consisting ($\int$) of all the associations (/) of $\mu_A(u)$ and u for all $u \in \mathbf{U}$. When **U** is discrete, it is represented as $A = \{(\mu_A(u_i), u_i) | i = 1, 2, \cdots, l\}$ or

$$A = \sum_{i=1}^{l} \mu_A(u_i) / u_i \tag{2.4}$$

where the consisting symbol is replaced by Σ.

Let's take the concept "tall" as an example of fuzzy set A. With **U** defined by the range of height of 150cm to 200cm, the degree to which someone's height x cm can be called "tall" is $\mu_{tall}(x)$ and the membership function μ_{tall} is depicted in Fig.2.1. Similarly, the concept "short" can be represented in a discrete way as Fig.2.2.

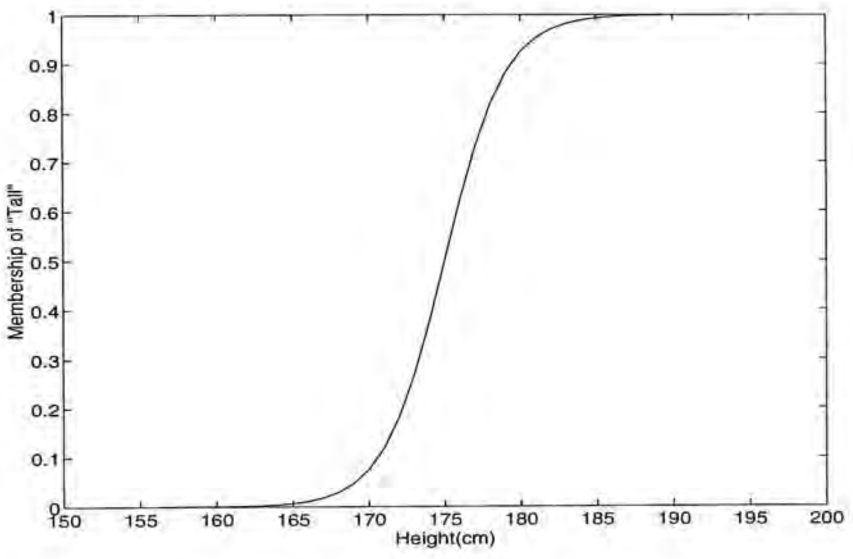


Figure 2.1 Continuous Fuzzy Set "Tall"

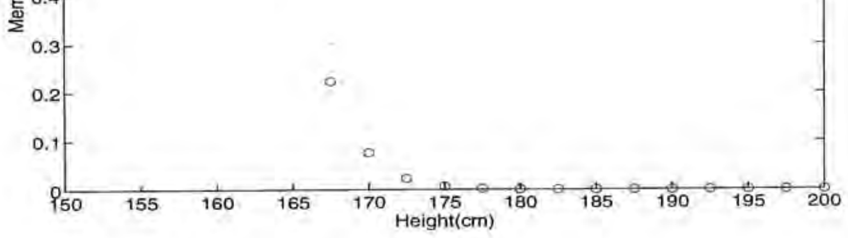


Figure 2.2 Discrete Fuzzy Set "Short"

Here are the some of the properties of fuzzy sets.

Definition 2.2

Support, Nucleus and Height : The support of a fuzzy set A is the crisp set that contains all elements of A with non-zero membership degree. Formally,

$$Support(A) = \{u \in \mathbf{U} | \mu_A(u) > 0\} \tag{2.5}$$

Closely connected to the support of a fuzzy set is nucleus. The nucleus of a fuzzy set A is the crisp set that contains all values with full membership degree. Formally,

$$Nucleus(A) = \{u \in \mathbf{U} | \mu_A(u) = 1\} \tag{2.6}$$

The height of a fuzzy set A is defined as

$$Height(A) = \sup_{u \in \mathbf{U}} \mu_A(u) \tag{2.7}$$

Definition 2.3

Normal and Singleton Fuzzy Set : A fuzzy set A is called normal if *Height*(A)=1. If the support of a fuzzy set A is a single point in $\mathbf{U}$ at which $\mu_A = 1$, A is called a singleton fuzzy set.

Fuzzy sets can also be defined in the product space of multiple universes of discourse and they are usually called fuzzy relations.

Definition 2.4

Fuzzy Relation : Let $\mathbf{U}$ and $\mathbf{V}$ be two universes of discourse. A fuzzy relation R is a fuzzy set in the product space $\mathbf{U}\times\mathbf{V}$; that is, R has the membership function $\mu_R(u,v)$, where $u \in \mathbf{U}$ and $v \in \mathbf{V}$. Similarly, an n-ary fuzzy relation is a fuzzy set in the product space $\mathbf{U}_1 \times \cdots \times \mathbf{U}_n$ and is expressed as

$$R_{\mathbf{U}_1 \times \cdots \times \mathbf{U}_n} = \{(\mu_R(u_1,\cdots,u_n),(u_1,\cdots,u_n)) | (u_1,\cdots,u_n) \in \mathbf{U}_1 \times \cdots \times \mathbf{U}_n\} \tag{2.8}$$

2.2 Fuzzy Set-Theoretic Operators

In classical set theory, the union, intersection and complement of sets are simple operations that are unambiguously defined. Due to the graded transition between full membership and no membership in fuzzy sets, the interpretations of fuzzy union, intersection and complement are not so simple.

Definition 2.5

Intersection, Union, and Complement : Let A and B be two fuzzy sets in **U**. Their intersection $A \cap B$ and union $A \cup B$ are fuzzy sets in **U** with membership functions defined for all $u \in \mathbf{U}$ by

$$\mu_{A\cap B}(u) = \min\{\mu_A(u), \mu_B(u)\} \tag{2.9}$$

and

$$\mu_{A\cup B}(u) = \max\{\mu_A(u), \mu_B(u)\} \tag{2.10}$$

respectively. Usually, the fuzzy intersection operator is denoted by $\wedge$ while the fuzzy union operator is denoted by $\vee$. The complement $\overline{A}$ of A is a fuzzy set in **U** defined for all $u \in \mathbf{U}$ by

$$\mu_{\overline{A}}(u) = 1 - \mu_A(u) \tag{2.11}$$

These are very simple extensions of the classical operators. The generalized operators for fuzzy intersection and union are *t*-norm and *s*-norm respectively.

Definition 2.6

t-norm : A two place function t:[0,1]×[0,1]→[0,1] is called *t*-norm iff for any $p,q,r \in [0,1]$:

(i) $t(p,1) = p$;

(ii) $p' \le p'' \Rightarrow t(p',q) \le t(p'',q)$ (monotonicity);

(iii) $t(p,q) = t(q,p)$ (commutativity);

(iv) $t(p,t(q,r)) = t(t(p,q),r)$ (associativity); and

(v) $t(0,p)=0$ (existence of 0).

Examples of *t*-norm operators include minimum, algebraic product, bounded product, and drastic product and their operations are defined for $p,q \in [0,1]$ as

$$p \wedge q = \min\{p,q\} \tag{2.12}$$

$$p \cdot q = pq \tag{2.13}$$

$$p \otimes q = \max\{0, p+q-1\} \tag{2.14}$$

$$p \hat{\times} q = \begin{cases} p & if\ q=1 \\ q & if\ p=1 \\ 0 & if\ p,q<1 \end{cases} \tag{2.15}$$

respectively. It has been pointed out that minimum is the largest *t*-norm, followed by algebraic product, bounded product, and drastic product [80], i.e.,

$$\mu_{p\wedge q} \geq \mu_{p\cdot q} \geq \mu_{p\otimes q} \geq \mu_{p\hat{\times} q} \tag{2.16}$$

This relationship has been depicted in Fig.2.3.

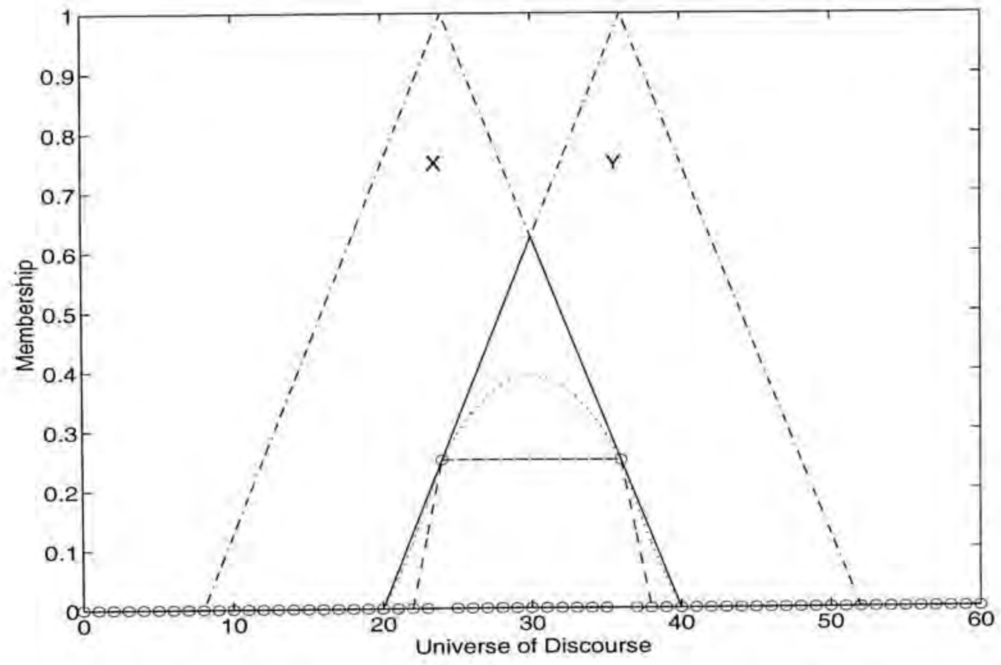


Figure 2.3 Fuzzy Intersection of X and Y. Solid line (—) : minimum, dotted line (...) : algebraic product, dash line (--) : bounded product, open circle (oo) : drastic product.

Definition 2.7

s-norm : A two place function s:[0,1]×[0,1]→[0,1] is called s-norm iff for any $p,q,r \in [0,1]$:

(i) $s(p,0) = p$;

(ii) $p' \le p'' \Rightarrow s(p',q) \le s(p'',q)$ (monotonicity);

(iii) $s(p,q) = s(q,p)$ (commutativity);

(iv) $s(p,s(q,r)) = s(s(p,q),r)$ (associativity); and

(v) $s(1,p) = 1$ (existence of 1).

Examples of s-norm operators includes maximum, algebraic sum, bounded sum, and drastic sum and their operations are defined for $p,q \in [0,1]$ as

$$p \vee q = \max\{p,q\} \tag{2.17}$$

$$p \dot{+} q = p + q - pq \tag{2.18}$$

$$p \oplus q = \min\{1, p+q\} \tag{2.19}$$

$$p \hat{+} q = \begin{cases} p & if\ q=0 \\ q & if\ p=0 \\ 1 & if\ p,q>0 \end{cases} \tag{2.20}$$

respectively. As shown in Fig.2.4, the following relationship [80]

$$\mu_{p \vee q} \le \mu_{p \dot{+} q} \le \mu_{p \oplus q} \le \mu_{p \hat{+} q} \tag{2.21}$$

holds true.

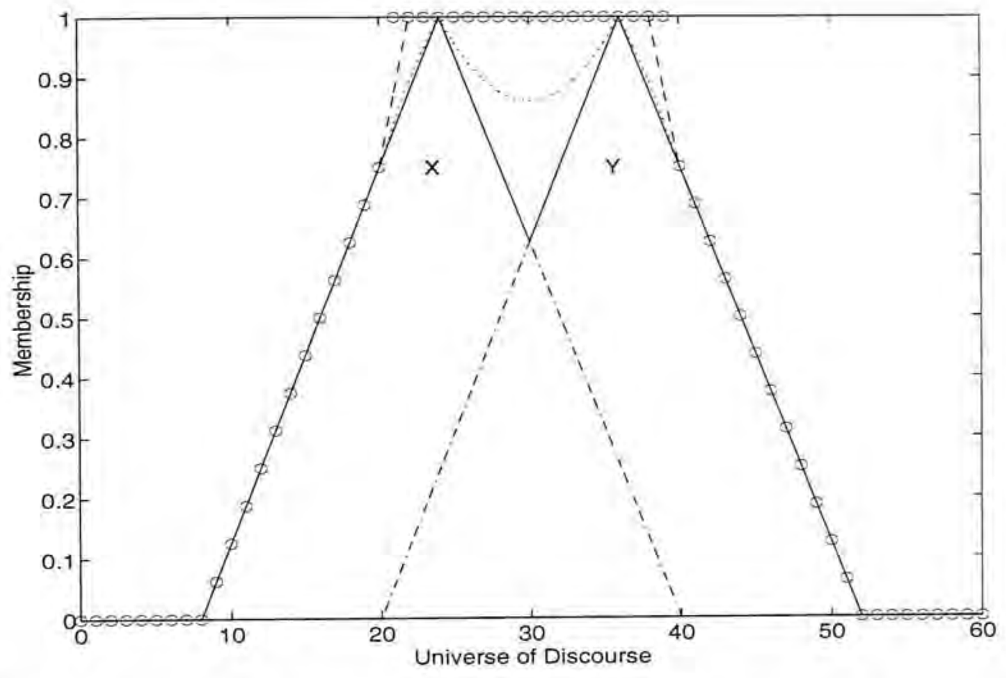


Figure 2.4 Fuzzy Union of *X* and *Y*. Solid line (—) : maximum, dotted line (...) : algebraic sum, dash line (--) : bounded sum, open circle (oo) : drastic sum.

Definition 2.8

Cartesian Product : If $A_1, A_2, \cdots, A_n$ are fuzzy sets in $\mathbf{U}_1, \mathbf{U}_2, \cdots, \mathbf{U}_n$ respectively, the Cartesian product of $A_1, A_2, \cdots, A_n$ is a fuzzy set in the product space $\mathbf{U}_1 \times \mathbf{U}_2 \cdots \times \mathbf{U}_n$ with membership function

$$\mu_{A_1 \times A_2 \cdots \times A_n}(u_1, u_2, \cdots, u_n) = \min\left\{\mu_{A_1}(u_1), \mu_{A_2}(u_2), \cdots, \mu_{A_n}(u_n)\right\} \tag{2.22}$$

or

$$\mu_{A_1 \times A_2 \cdots \times A_n}(u_1, u_2, \cdots, u_n) = \mu_{A_1}(u_1) \cdot \mu_{A_2}(u_2) \cdots \mu_{A_n}(u_n) \tag{2.23}$$

or

$$\mu_{A_1 \times A_2 \cdots \times A_n}(u_1, u_2, \cdots, u_n) = \mu_{A_1}(u_1)\, t\, \mu_{A_2}(u_2)\, t \cdots t\, \mu_{A_n}(u_n) \tag{2.24}$$

2.3 Linguistic Variable, Fuzzy Rule, and Fuzzy Inference

Based upon the concepts of fuzzy sets, the notion of a *linguistic variable* which is the fundamental knowledge representation unit in fuzzy systems was established. In [120], Zadeh states :

"By a *linguistic variable* we mean a variable whose values are words or sentences in a natural or artificial language. For example, *age* is a linguistic variable if its values are linguistic rather than numerical, i.e., *young, not young, very young, quite young, old, not very old, and not very young,* etc., rather than 20, 21, 22, 23, ..."

A linguistic variable is characterized by a quadruple

$$\langle \mathrm{X}, \mathrm{T(X)}, \mathbf{U}, \mathrm{M_X} \rangle$$

where X denotes the symbolic name of a linguistic variable, e.g., height, age, speed, temperature, error, and change-of-error, etc. T(X) is the set of linguistic values or terms, also known as fuzzy terms, that X can take and is also called the term set of X, e.g., T(temperature)=$\{$*cold, cool, comfortable, warm, hot*$\}$. **U** is the actual physical domain over which the linguistic variable X takes its quantitative crisp value, i.e., the universe of discourse. M_{X} is a semantic function which gives a "meaning" of a linguistic value in terms of the quantitative elements of X, i.e.,

$$M_{\mathrm{X}}: \mathrm{V_X} \in \mathrm{T(X)} \rightarrow A \tag{2.25}$$

where A denotes a fuzzy set defined over **U**. In other word, M_{X} associates the linguistic terms with appropriate fuzzy sets.

With the linguistic variables defined, human (expert) knowledge can then be expressed by fuzzy IF-THEN rules in the form of

$$\text{IF} \langle \textit{fuzzy proposition} \rangle \text{ THEN } \langle \textit{fuzzy proposition} \rangle$$

For example in fuzzy control, the relationship between process state and control output variable can be described by

IF error is *negative big (NB)* and change-of-error is *positive big (PB)*

THEN control is *negative small (NS)*

which stands for the linguistic description rule

"If it is the case that the current error is negative big and the current change-of-error is positive big, then this is a cause for a small decrease in the previous value of the control output."

Here, error, change-of-error, and control are the linguistic variables and *NB*, *PB*, and *NS* are the corresponding linguistic terms. Such a rule is usually implemented by a fuzzy implication (relation) *R* and is defined as follows :

$$\begin{aligned}\mu_R &\equiv \mu_{NB\ and\ PB \to NS}(u, v, w) \\ &= [\mu_{NB}(u)\ and\ \mu_{PB}(v)] \to \mu_{NS}(w)\end{aligned} \tag{2.26}$$

where "*NB and PB*" is a fuzzy set in $\mathbf{U}\times\mathbf{V}$; $R \equiv (NB\ and\ PB) \to NS$ is a fuzzy implication (relation) in $\mathbf{U}\times\mathbf{V}\times\mathbf{W}$; and $\to$ denotes a fuzzy implication function. There are may ways in which a fuzzy implication can be defined. Indeed, nearly 40 distinct fuzzy implication functions have been described in the literature [72]. To name a few, the fuzzy relations

$$\begin{aligned}R_a &= (\overline{A}\times\mathbf{V})\oplus(\mathbf{U}\times B) \\ &= \int_{\mathbf{U}\times\mathbf{V}} \min\{1, 1-\mu_A(u)+\mu_B(v)\} / (u, v)\end{aligned} \tag{2.27}$$

$$\begin{aligned}R_c &= A\times B \\ &= \int_{\mathbf{U}\times\mathbf{V}} \mu_A(u)\wedge\mu_B(v) / (u, v)\end{aligned} \tag{2.28}$$

$$\begin{aligned}R_p &= A\times B \\ &= \int_{\mathbf{U}\times\mathbf{V}} \mu_A(u)\mu_B(v) / (u, v)\end{aligned} \tag{2.29}$$

where *A* is a fuzzy set in **U** and *B* is a fuzzy set in **V** were proposed by Zadeh [121], Mamdani [77], and Larsen [70] respectively. In fuzzy control applications, Mamdani's minimum rule of fuzzy implication is undoubtly the most well-known one and will be revisited in next section.

Approximate reasoning is the best-known form of fuzzy logic on which fuzzy system is based [34]. It covers a variety of inference rules whose premises contains fuzzy propositions. There exist two important fuzzy inference rules, namely, generalized modus ponens (GMP) and generalized modus tollens (GMT), and they can be defined respectively by the inference procedures

premise #1 : X is A'

premise #2 : If X is A then Y is B

consequence : Y is B'

and

premise #1 : Y is B'

premise #2 : If X is A then Y is B

consequence : X is A'

where X and Y are linguistic variables and A, A', B, and B' are the corresponding fuzzy terms. The GMP is closely related to forward data-driven inference which is particularly useful in fuzzy systems. On the other hand, the GMT is closely related to backward goal-driven inference which is commonly used in expert systems, especially in the realm of medical diagnosis [71]. With the implementation of premise #2 by a fuzzy implication (relation), Zadeh proposed a fuzzy inference mechanism called *compositional rule of inference* [119]. Let R be a fuzzy relation in $\mathbf{U}\times\mathbf{V}$ for premise #2 of the GMP and A' be a fuzzy set in $\mathbf{U}$. The compositional rule of inference asserts that the fuzzy set B' in $\mathbf{V}$ induced by A' is given by

$$B' = A' \circ R \tag{2.30}$$

where "$\circ$" denotes the sup-min composition. Thus,

$$\mu_{B'}(v) = \sup_{u\in\mathbf{U}}(\mu_R(u,v) \wedge \mu_{A'}(u)) \tag{2.31}$$

If $\mathbf{U}$ is discrete, eq.(2.31) becomes

$$\mu_{B'}(v) = \max_{u\in\mathbf{U}}(\mu_R(u,v) \wedge \mu_{A'}(u)). \tag{2.32}$$

and "$\circ$" denotes the max-min composition.

2.4 Basic Structure of a Fuzzy System

Having briefly reviewed the mathematical and conceptual backgrounds of fuzzy sets, we proceed to describe the basic structure of a fuzzy system. Basically, a fuzzy system consists of four principle components, namely, a fuzzifier, a fuzzy knowledge base, a fuzzy inference engine, and a defuzzifier, as shown in Fig.2.5. Their functions and operations are described as follows.

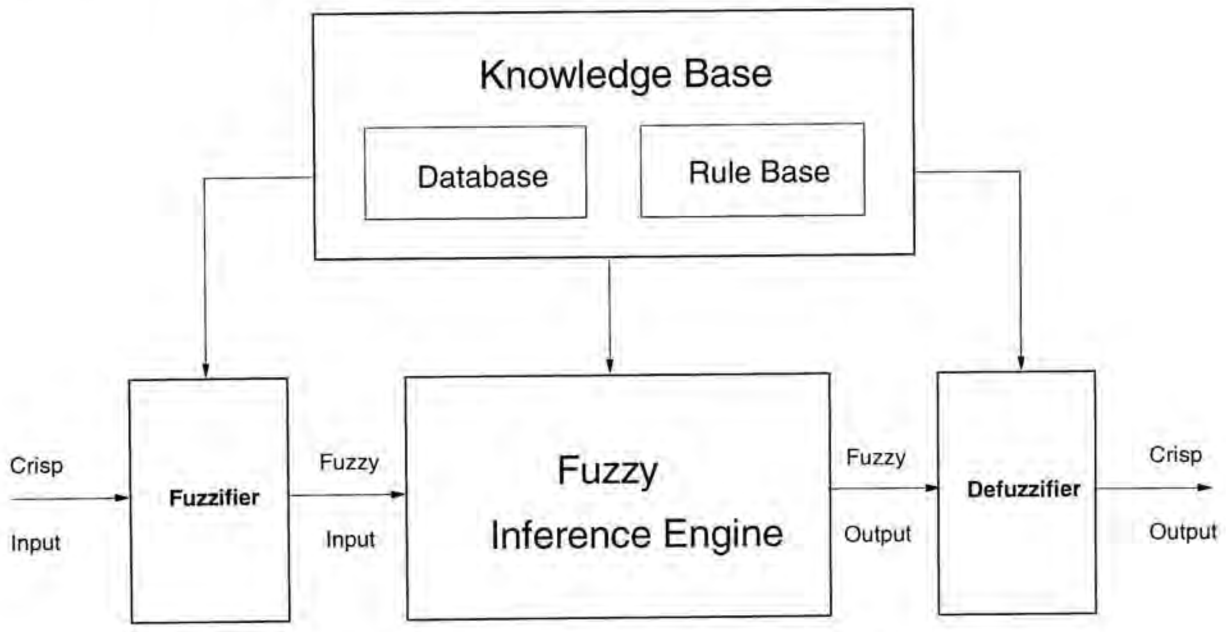


Figure 2.5 Basic Structure of a Fuzzy System

2.4.1 Fuzzifier

Since the observed data in most fuzzy system applications are crisp, they must be fuzzified before passing to the inference engine for further processing. Hence, the main function of a fuzzifier is to convert input point-wise data into suitable linguistic representations in the input universes of discourse. Depending on the inference mechanism adopted, two types of fuzzifiers have been proposed. One has already been mentioned in Chapter 1. It computes the membership values of the crisp input(s) to each linguistic term of the rule antecedent, i.e., $f : u' \in \mathbf{U} \rightarrow \mu(u')$. The second type of fuzzifiers on the other hand performs a mapping $f : u' \in \mathbf{U} \rightarrow A \ in \ \mathbf{U}$. There are at least two possible choices of this mapping [112].

1. Singleton Fuzzifier : A is a singleton fuzzy set with support u', i.e.,

$$\mu_A(u) = \begin{cases} 1 & if \ u = u' \\ 0 & \forall \ u \neq u' \end{cases} \tag{2.33}$$

2. Non-singleton Fuzzifier : A is a normal fuzzy set with

$$\mu_A(u) = \begin{cases} 1 & \text{at } u = u' \\ \text{decreasing from 1} & \text{as } u \text{ moves away from } u' \end{cases} \tag{2.34}$$

The triangular and bell-shaped membership functions fall into this category. Among them, the singleton fuzzifier is a more popular choice though the non-singleton one may be useful if the inputs are corrupted by noise [112].

2.4.2 Fuzzy Knowledge Base

As represented in Fig.2.5, the knowledge base provides the necessary information for fuzzifier, inference engine, and defuzzifier. It consists of a data base and a rule base. The data base consists of information such as

- *the universes of discourse* which are either continuous or discrete. If they are continuous, a process of discretization and normalization is usually required for effective digitial processing by computer [71].
- *the membership functions* of fuzzy sets taken by the linguistic terms of fuzzy rules. They can be represented as a vector of numbers or a function such as the triangular, trapezoidal, or bell-shaped type.

On the other hand, the rule base consists of a collection of fuzzy IF-THEN rules :

$$\begin{array}{l} Rule^{(1)}\text{: IF } \mathrm{X}_1 \text{ is } A_1^{(1)} \text{ and } \mathrm{X}_2 \text{ is } A_2^{(1)} \text{ and } \cdots \text{ and } \mathrm{X}_n \text{ is } A_n^{(1)} \text{ THEN Y is } B^{(1)} \\ Rule^{(2)}\text{: IF } \mathrm{X}_1 \text{ is } A_1^{(2)} \text{ and } \mathrm{X}_2 \text{ is } A_2^{(2)} \text{ and } \cdots \text{ and } \mathrm{X}_n \text{ is } A_n^{(2)} \text{ THEN Y is } B^{(2)} \\ \quad \vdots \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \vdots \\ Rule^{(L)}\text{: IF } \mathrm{X}_1 \text{ is } A_1^{(L)} \text{ and } \mathrm{X}_2 \text{ is } A_2^{(L)} \text{ and } \cdots \text{ and } \mathrm{X}_n \text{ is } A_n^{(L)} \text{ THEN Y is } B^{(L)} \end{array}$$

where $\mathrm{X}_1, \mathrm{X}_2, \cdots, \mathrm{X}_n$ are the linguistic input variables, Y is the linguistic output variable, and $A_1^{(i)}, A_2^{(i)}, \cdots, A_n^{(i)}, B^{(i)}$ are the linguistic terms of the *i*th fuzzy rule. It corresponds to the modelling of multiple-input-single-output (MISO) fuzzy systems and can be easily extended to cater for the multiple-input-multiple-output (MIMO) fuzzy systems. Furthermore, it is general enough to include other types of fuzzy rules, e.g., rules with "or" connectives, unless rules, non-fuzzy rules, and rules with incomplete IF-part [112]. Hence, we will focus on the fuzzy rules with "and" connectives depicted above in this thesis.

2.4.3 Fuzzy Inference Engine

In Section 2.3, we have already reviewed the fuzzy inference with a single rule. We continue in this section to describe the fuzzy inference with a set of fuzzy rules that characterizes a fuzzy system. There exist two types of fuzzy inference, namely, *composition based inference* and *individual-rule based inference* [34].

In composition based inference, the fuzzy relations representing the meaning of each individual rule are aggregated into one fuzzy relation describing the meaning of the overall set of rules. Then inference takes place by the composition between the fuzzy inputs (or fuzzified inputs from the second type of fuzzifiers) and the aggregated fuzzy relation. The output fuzzy set, as a result of the composition, describes the value of the system output. Let's take a look at an example. For the following set of L fuzzy rules

$Rule^{(1)}$: IF E is *zero* and ΔE is *zero* THEN C is *zero*

$\vdots \qquad \vdots$

$Rule^{(i)}$: IF E is $A_1^{(i)}$ and ΔE is $A_2^{(i)}$ THEN C is $B^{(i)}$

$\vdots \qquad \vdots$

$Rule^{(L)}$: IF E is *zero* and ΔE is *PS* THEN C is *NS*

the ith rule can be implemented by a fuzzy relation in terms of the Mamdani type implication function as

$$\mu_{R^{(i)}}(u_1,u_2,v) = \min\left\{\mu_{A_1^{(i)}}(u_1),\mu_{A_2^{(i)}}(u_2),\mu_{B^{(i)}}(v)\right\} \qquad \forall u_1 \in \mathbf{U_1}, u_2 \in \mathbf{U_2}, v \in \mathbf{V} \tag{2.35}$$

It is then aggregated with other fuzzy relations by fuzzy union to form a single fuzzy relation

$$\mu_R(u_1,u_2,v) = \max_{i=1}^{L} \mu_{R^{(i)}}(u_1,u_2,v) \quad \forall u_1 \in \mathbf{U_1}, u_2 \in \mathbf{U_2}, v \in \mathbf{V} \tag{2.36}$$

representing the meaning of the set of fuzzy rules. With the fuzzy inputs in $\mathbf{U_1} \times \mathbf{U_2}$ represented by their cartesian product as

$$\mu_{antecedent}(u_1,u_2) = \min\left\{\mu_{A_1^{(i)}}(u_1),\mu_{A_2^{(i)}}(u_2)\right\} \quad \forall u_1 \in \mathbf{U_1}, u_2 \in \mathbf{U_2} \tag{2.37}$$

the inference output according to the compositional rule of inference is

$$\mu_{consequent}(v) = \max_{u_1,u_2}\left\{\mu_{antecedent}(u_1,u_2) \wedge \mu_R(u_1,u_2,v)\right\} \tag{2.38}$$

The major disadvantage of composition based inference is to store all the relation elements $\mu_R(u_1,u_2,v)$. Also, the computational costs of computing $\mu_{antecedent}$ and performing the compositional rule of inference are very high if parallel hardware is not available. As such, the individual-rule based inference discussed in Section 1.1 is usually perferred traditionally, particularly in fuzzy control applications.

In individual-rule based inference, the rules are fired individually before their outputs are aggregated to form the overall output. The firing process can be described by two steps : i) computing the degree of match between the crisp input and the rule-antecedent's fuzzy sets via the first type of fuzzifiers; and ii) "clipping" the rule-consequent's fuzzy sets to the degree to which the rule-antecedent has been matched by the crisp input. The "clipped" output of each rule is then aggregated by some sort of fuzzy union operations to produce the final inference output. These two steps can be best illustrated by Fig.2.6 with respect to the example given for the composition based inference. It can be seen that the degrees of match between a crisp input E=6 and the *zero* fuzzy terms of $Rule^{(1)}$ and $Rule^{(L)}$ are both 0.5 while those between a crisp input $\Delta E = 1.5$ and the *zero* fuzzy term of $Rule^{(1)}$ and *PS* fuzzy term of $Rule^{(L)}$ are 0.7 and 0.3 respectively. By taking the minimum of the degrees of match for each rule, the "clipped" rule-consequent's fuzzy sets with heights 0.5 & 0.3 respectively are shown on the right side of Fig.2.6. They are then aggregated using the maximum operator to produce the output fuzzy set for these two rules. The crisp output is generated by a defuzzification process to be discussed in section 2.4.4. Such an inference process is usually implemented by a lookup table approach.

It has been pointed out that such kind of inference is equivalent to the composition based inference if Mamdani-type fuzzy implication is employed [34]. Furthermore, it is computationally efficient and requires little memory storage. Therefore, it is frequently adopted in many fuzzy system applications.

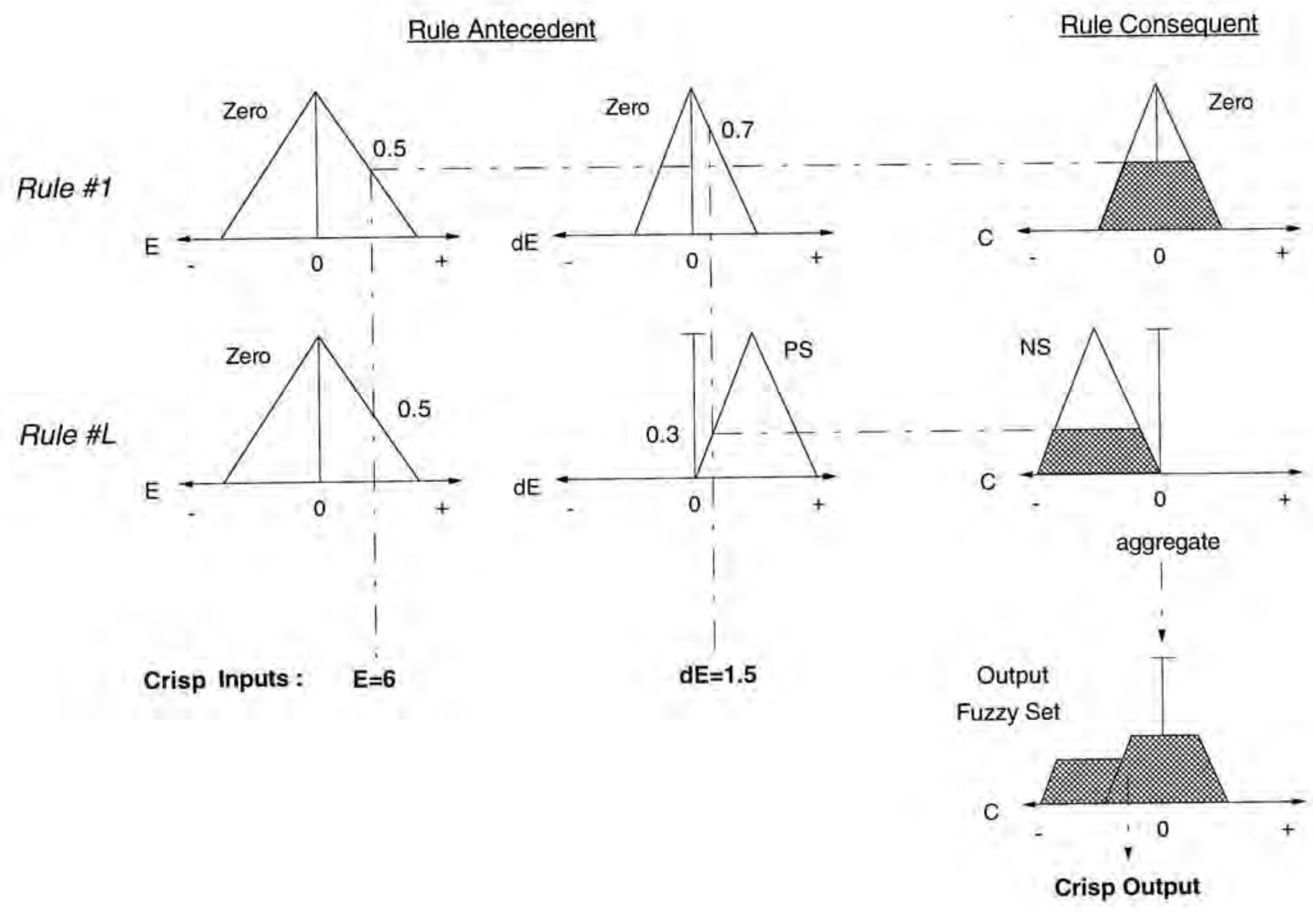


Figure 2.6 Individual-Rule Based Fuzzy Inference

2.4.4 Defuzzifier

With the output fuzzy set in hand, the defuzzifier performs a mapping from fuzzy set in $\mathbf{V}$ to a crisp point $v^* \in \mathbf{V}$, i.e., $f: C \in \mathbf{V} \rightarrow v^* \in \mathbf{V}$. Its goal is to produce a crisp output that best represents the output fuzzy set *C*. There exists a lot of defuzzification strategies, see, e.g. [34], and two popular choices are depicted below.

1. Center of Gravity (COG) Defuzzifier : The widely used COG method regards the center of the area below the output fuzzy set as the crisp output. In the case of a discrete universe of discourse, the COG defuzzifier yields

$$v^* = \frac{\sum_{j=1}^{l} \mu_C(v_j) \cdot v_j}{\sum_{j=1}^{l} \mu_C(v_j)} \tag{2.39}$$

   where l is the cardinality of $\mathbf{V}$, i.e., the number of quantization levels at the output space.

2. Height Defuzzifier : The height method on the other hand takes the nucleus value of each output fuzzy set and builds a weighted sum of them according to their fired values. This is a very simple method and is computationally efficient. Let c_i and f_i be the nucleus and fired values of the output fuzzy terms respectively in the ith rule. The height defuzzifier yields

$$v^* = \frac{\sum_{i=1}^{L} c_i \cdot f_i}{\sum_{i=1}^{L} f_i} \tag{2.40}$$

   where L is the total number of fuzzy rules.

2.5 Concluding Remarks

A brief introduction to fuzzy sets and systems is given in this chapter. We see that fuzzy set with its membership being a matter of degree is particularly advantageous in modelling concepts that are imprecise or vague in nature. This can be especially useful in problems like pattern classification where, frequently, objects are not clearly members of one class or another. We also see that fuzzy systems, as a result of a sequence of developments in fuzzy set theory, namely, fuzzy sets, linguistic variables, fuzzy rules, inference with a single rule, and inference with a system of fuzzy rules, provide an interface between human knowledge and numerical information processing and hence linguistic control/decision rules which are essential to the success of modelling complex or ill-posed systems can be incorporated into the model. There exists some variants of the fuzzy system described in Section 2.4. For example, Takagi and Sugeno [107] proposed to use the following fuzzy IF-THEN rules :

$$\begin{aligned}&Rule^{(i)}\text{: IF } \mathrm{X}_1 \text{ is } A_1^{(i)} \text{ and } \mathrm{X}_2 \text{ is } A_2^{(i)} \text{ and } \cdots \text{ and } \mathrm{X}_n \text{ is } A_n^{(i)},\\ &\qquad \text{THEN } y^{(i)} = a_0^{(i)} + a_1^{(i)}x_1 + \ldots + a_n^{(i)}x_n\end{aligned}$$

where $a_0^{(i)}, a_1^{(i)}, \cdots, a_n^{(i)}$ are real-valued parameters, $x_1, \cdots, x_n$ are the crisp inputs, and $y^{(i)}$ is the crisp system output due to the *i*th rule. The definition of the rule antecedent is the same as before. It can be seen that the rule consequent is not fuzzy, therefore it does not provide a natural framework to incorporate fuzzy rules from human experts. On the other hand, Pedrycz [88] proposed to use the fuzzy relational equations [103] to model fuzzy rules. We will discuss this variant in a greater detail in chapters 6 & 7.

Chapter 3

Categories of Fuzzy Neural Systems

3.1 Introduction

Being motivated by the advances in theory and applications of fuzzy systems and neural networks, the interest in integrating these two fields has been rigorous and fascinating during the past few years. There are several reasons for this. First, the enormous success of commercial applications of fuzzy systems has led to a surge of curiosity about the utility of fuzzy technologies for scientific and engineering applications. Second, the synergism of fuzzy systems and neural networks has a sound technical basis, because these two techniques generally attack the design of intelligent systems from quite different aspects. As mentioned in Chapter 1, fuzzy systems are generally advantageous in knowledge representation and logical reasoning under cognitive uncertainty while neural networks are distinguished for their learnability, parallelism, and adaptability to changing environments. Consequently, they often complement each other. Third, there seems to be many ways to use either technology as a "tool" within the framework of a model based on the other. For example, neural network which is well-known for its ability to represent functions can be used to provide good approximations to the membership functions of fuzzy systems.

There have been many attempts to synthesize fuzzy neural system (FNS) models in the last few years and according to their integration methodologies, two major categories of FNS's can be identified. One is based on the fuzzification of conventional neural network models and the other is based on the implementation of fuzzy systems using neural networks. Based upon such a categorization of FNS models, the representative models in each category are briefly reviewed in this chapter. It is by no means an extensive survey of FNS models but rather serves as an overview of the major methodologies and merits of integrating fuzzy systems and neural networks. After that, the motivation of the proposed three areas of study is described.

3.2 Fuzzification of Neural Networks

Since its introduction in 1965, fuzzy set theory [118] has been continuously exploited by researchers to fuzzify (or generalize) existing algorithms or techniques such that their performances can be enhanced. In the field of pattern classification, for example, various well-known supervised and unsupervised training algorithms have been successfully fuzzified. In [57], a fuzzy perceptron algorithm was proposed to ameliorate the convergence problem of classical perceptron algorithm in nonseparable training data sets. The fuzzy *k*-nearest neighbor algorithm [56] assigns fuzzy membership values rather than a particular class to the pattern being classified according to its distance from *k* nearest neighbors and those neighbors' fuzzy memberships in the possible class. As reported, the algorithm not only outperforms the original *k*-nearest neighbor algorithm in classification rate, the resulting membership values also provide a confidence measure of the classification. The fuzzy *c*-means algorithm [6,36] perhaps is the most notable fuzzy algorithm for pattern classification. As pointed out in [105], the fuzzy *c*-means algorithm is especially useful in applications where clusters touch or overlap. Furthermore, it is superior to its crisp counterpart in yielding more often the global optimum [109].

In view of the success of fuzzy pattern classification algorithms and the close relationship between neural networks and pattern classification, the first category of FNS models has been devised to fuzzify conventional NN models by incorporating different fuzzy concepts at various stages of the networks such that their performances are improved and their abilities to handle uncertain information are enhanced. According to the fuzzy concepts being employed, it can be divided into three subcategories, namely, fuzzy membership driven models, fuzzy operator driven models, and fuzzy arithmetic driven models.

3.2.1 Fuzzy Membership Driven Models

- **A Fuzzy Version of Multilayer Feedforward Network**

In [85], Pal and Mitra proposed a FNS model based on the multilayered feedforward network. The fuzzification efforts include i) expressing the input features in terms of membership values to each of the three linguistic properties *low* (L), *medium* (M), and *high* (H); and ii) expressing the output vector in terms of fuzzy class membership values. Let $\vec{x}_k = [x_{1k}, x_{2k}, \cdots, x_{nk}]$ be an *n*-dimensional crisp input pattern belonging to the *i*th class of an *c*-class problem. The fuzzified input representation of $\vec{x}_k$ is an 3*n*-dimensional vector

$$\vec{x}_k = [\mu_L(x_{1k}), \mu_M(x_{1k}), \mu_H(x_{1k}), \mu_L(x_{2k}), \cdots, \mu_L(x_{nk}), \mu_M(x_{nk}), \mu_H(x_{nk})] \quad (3.1)$$

while the desired output representation of $\vec{x}_k$ is

$$\vec{o}_k = [\mu_1(\vec{x}_k), \cdots, \mu_i(\vec{x}_k), \cdots, \mu_c(\vec{x}_k)] \quad (3.2)$$

where $\mu_i(\bar{x}_k) \in [0,1]$. These modifications enable efficient modelling of fuzzy or uncertain patterns, with appropriate weights being assigned to the back-propagated errors employed during training. The fuzzified model as a result is distinctive in classifying fuzzy data with overlapping class boundaries and is able to perform fuzzy classification of input patterns [85].

- **A Fuzzy Version of Kohonen Map**

In [78], Mitra and Pal applied their fuzzification methodology for multilayered feedforward network to the Kohonen map and proposed a fuzzy version of it. Self-organization takes place with respect to the fuzzified training patterns which is a concatenation of the membership values in eq.(3.1) and the fuzzy class membership values in eq.(3.2), i.e.,

$$\bar{\chi}_k = [\bar{x}_k, 0] + [0, \bar{o}_k] = [\bar{x}_k, \bar{o}_k] \tag{3.3}$$

As demonstrated in [78], the proposed model is capable of producing fuzzy partitioning of the output space and can thereby provide a more faithful representation for ill-defined or fuzzy data with overlapping classes.

- **Another Fuzzy Version of Kohonen Map**

Tsao *et al.* [110] have proposed a different fuzzy version of Kohonen map called fuzzy Kohonen clustering network. Their motivation was to address some intrinsic problems of the crisp model, namely, neither termination nor convergence is guaranteed, no model is optimized by the learning strategy, and the performance is sensitive to the initial condition, training data sequence, and training parameters such as the learning rate and the size of update neighborhood. Fuzzification was accomplished by integrating the fuzzy *c*-means (FCM) algorithm into the Kohonen map. This yields an optimization problem related to the FCM algorithm and takes use of the membership values from FCM to control of both the learning rate distribution and update neighborhood. The proposed model can be considered as a Kohonen map implementation of the FCM algorithm and has been shown to improve the convergence as well as reduce the labelling errors of the crisp model.

3.2.2 Fuzzy Operator Driven Models

- **Fuzzy ART**

Fuzzy ART [11], as proposed by Carpenter, Grossberg, and Rosen, is one of the earliest attempts to incorporate fuzzy concepts into conventional neural network models. It represents a fuzzification of the ART 1 model [46] that can learn stable categories only in response to binary input patterns. By replacing the intersection operator ($\cap$) in ART 1 by the *min* operator ($\wedge$) of fuzzy set theory, the fuzzy ART model can learn stable categories in response to either binary or analog input patterns. It is indeed a generalization of ART 1 because it reduces to ART 1 when binary input patterns are presented to the network.

- **Logic-Based Neural Networks**

In [96], Pedrycz introduced a FNS model consisting of logic-based neurons for realizing mechanisms of pattern matching and aggregation. It can be considered as a structural fuzzification of three-layer feedforward networks where all computations in the networks are driven by *t*-norm and *s*-norm logic operations. Let $\vec{x}_k = [x_{1k}, x_{2k}, \cdots, x_{nk}] \in [0,1]^n$ be an input pattern and $\vec{r}_j = [r_{1j}, r_{2j}, \cdots, r_{nj}]$ be a reference pattern in $[0,1]^n$. The hidden node, termed as reference neuron, is described by

$$h_j = \overset{n}{\underset{i=1}{S}}\left[w_{ij}\, t(x_{ik} \equiv r_{ij}) \right] \tag{3.4}$$

where w_{ij} is the interconnecting weight from ith input to jth hidden node, $\equiv$ denotes the fuzzy matching operator introduced in [93], and the *s*-*t* composition corresponds to the fuzzy counterpart of the product-sum operation in conventional multilayer feedforward networks. The reference neuron is devised to realize matching of input pattern with regard to an associated reference pattern. The output neuron then summarizes the level of activations from the reference neurons using

$$y = \overset{p}{\underset{j=1}{S}}\left[v_j \, t \, h_j\right] \tag{3.5}$$

where the interconnecting weight v_j models the influence stemming from the *j*th reference neuron. As mentioned in [96], the model not only can cope with classification problems in which classes are distributed along several linearly nonseparable regions, it also provides a straight-forward interpretation regarding the shape of generated regions, the significance of particular features of the patterns, and the impact of the detected regions on the final classification.

3.2.3 Fuzzy Arithmetic Driven Models

A detail discussion of this subcategory of FNS models can be found in Buckley and Hayashi's recent survey paper [9]. It is referred to the layered feedforward network whose inputs, outputs, and/or interconnecting weights are modelled by fuzzy numbers [55], i.e., fuzzy sets defined in the real number space. Three types of such FNS models have been defined : (i) networks with fuzzy weights only; (ii) networks with fuzzy input-output only; (iii) networks with both fuzzy weights and fuzzy input-output. The corresponding learning algorithms, usually known as fuzzy back-propagation, have also been developed. This subcategory of FNS models has been so far studied theoretically, e.g., proving their universal approximation capability, though, it should have great potential in fuzzy regression analysis and fuzzy expert systems.

3.3 Layered Network Implementation of Fuzzy Systems

The second category of FNSs considers neural network as a learnable, parallel and distributed processing platform for the implementation of different fuzzy systems models. It takes use of a common feature of fuzzy systems and neural networks — distributed representation. In neural networks, as is well-known, knowledge is distributed among interconnecting weights and nodes, while in fuzzy systems, knowledge is distributively encoded by fuzzy rules, their fuzzy terms and the associated fuzzy sets. If appropriate mapping between fuzzy knowledge and network components exists, fuzzy systems can be realized by neural network architecture. Furthermore, learning ability can be introduced into the resulted FNS models. In what follows, the FNS models of this category are reviewed according to the types of fuzzy systems being implemented. They includes Mamdani's type which is essentially referred to systems using individual-rule based inference as depicted in Fig.2.6, Takagi and Sugeno's type which employs the crisp consequent fuzzy rules mentioned in Section 2.5, and fuzzy relation based type whose knowledge is encoded in fuzzy relation format.

3.3.1 Mamdani's Fuzzy Systems

In [74], Lin and Lee proposed a FNS model that implements Mamdani's fuzzy systems using multilayer feedforward networks. The model consist of five layers whose functions correspond to the inference steps depicted in Fig.2.6. Nodes in layer one, i.e., the input nodes, are *linguistic nodes* which represent the input linguistic variables. They buffer the crisp input values. Layer five as the output layer denotes the output linguistic variables. It has two linguistic nodes for each output variable. One is to feed the desired crisp output into the network for training and the other is to produce the defuzzified system outputs. Nodes in layers two and four are *term nodes* which act as membership functions to represent the fuzzy terms of the respective linguistic variables. Each node in layer three is a *rule node* which represents one fuzzy rule. Its incoming and outgoing links define the preconditions and consequents of the IF-THEN rules. Hence, for each rule node, there is at most one link (maybe none) from the term nodes of each linguistic node. Thus, all the layer-three nodes form a fuzzy rulebase. With such network model, the transfer functions of the nodes in each of the five layers are linear function, bell-shaped membership function, minimum function (computing the degree of match), minimum function (clipping the rule-consequent's fuzzy sets), and defuzzification function respectively.

Based on the well-known back-propagation algorithm, a two-phase hybrid learning algorithm has been developed for the proposed FNS model. The algorithm determines the optimal centers and widths of the bell-shaped membership functions of term nodes in layers two and four. It also learns the fuzzy rulebase by deciding the existence of links between layers two and three and layers three and four. The model can be contrasted with the traditional fuzzy systems with its network structure and learning ability. It can be constructed from training examples by neural network learning techniques, and its structure can be trained to develop an appropriate fuzzy rulebase and find optimal input/output membership functions in an autonomous manner.

3.3.2 Takagi and Sugeno's Fuzzy Systems

The ANFIS fuzzy neural system model was proposed by Jang [54] to implement Takagi and Sugeno's fuzzy systems using layered network architecture. In the reported simulations, the model has yielded remarkable results as compared with Rumelhart *et al.*'s multilayered feedforward networks and Fahlman and Lebiere's cascade-correlation learning networks [37]. It also consists of five layers where layer one computes the membership values, layer two produces the degree of match for each rule by multiplying the incoming signals (i.e., taking the algebraic product *t*-norm operation), layer three calculates the normalized degree of match, layer four weights the rule's crisp output by the normalized degree of match, and layer five computes the overall output as the summation of all weighted crisp outputs. Such an architecture had been formulated according the inference steps of the Takagi and Sugeno's fuzzy systems. Consider a rulebase consisting of two rules

$Rule^{(1)}$: IF X_1 is $A_1^{(1)}$ and X_2 is $A_2^{(1)}$ THEN $y^{(1)} = a_0^{(1)} + a_1^{(1)}x_1 + a_2^{(1)}x_2$

$Rule^{(2)}$: IF X_1 is $A_1^{(2)}$ and X_2 is $A_2^{(2)}$ THEN $y^{(2)} = a_0^{(2)} + a_1^{(2)}x_1 + a_2^{(2)}x_2$

Let f_1 & f_2 be the degrees of match for rules 1 & 2 respectively. The inference and defuzzification continue by computing

$$y = \frac{f_1 y^{(1)} + f_2 y^{(2)}}{f_1 + f_2} = \bar{f}_1 y^{(1)} + \bar{f}_2 y^{(2)} \tag{3.6}$$

It can be seen that layer three corresponds to the computation of the normalized degrees of match $\bar{f}_1$ & $\bar{f}_2$, layer four determines the values of $\bar{f}_1 y^{(1)}$ & $\bar{f}_2 y^{(2)}$ and layer five adds them to produce the final crisp output. Again, learning has been introduced into the network model to generate a set of Takagi and Sugeno's fuzzy rules from the available input-output data pairs.

3.3.3 Fuzzy Relation Based Fuzzy Systems

- **Fuzzy Associative Memory (FAM)**

  As already mentioned in Chapter 1, Kosko's FAM model [67] is characterized by a two-layer heteroassociative feedforward network that stores the discrete input-output fuzzy set pair in its weight matrix. In order to avoid crosstalks, separate storage of the fuzzy pattern pairs was proposed. Therefore, the capacity of a FAM matrix has been assumed to be one rule only. Upon presenting an input fuzzy set, max-min or max-product compositions are carried out individually and the corresponding outputs are then superimposed to form the final output fuzzy set.

The overall FAM system therefore comprises a bank of FAM matrices followed by an appropriate superimposition operator, e.g., weighted sum or maximum. Hence, the FAM model is also based on individual-rule based inference but it encodes the fuzzy knowledge in fuzzy relation (matrix) format. Unlike the previous two FNS models, learning takes place to define the associations between input and output fuzzy sets by a clustering process. Once they are learned, they can be directly implemented with layered network architecture. The model has been successfully applied to problems like backing up a truck-and-trailer [65], target tracking [67, Ch.11], and voice cell control in asynchronous transfer mode (ATM) networks [81] where distinctive features like modularity, robustness, and adaptability have been demonstrated.

- **Fuzzy Relational Neural System (FRNS)**

  Fuzzy relational equations [103], in their generic form, are non-linear equations in which the relation R determines the mappings between the associated pairs of input and output fuzzy sets (X_i, Y_i). Each equation corresponds to a single rule and a system of equations results when there is a set of rules. In [95,97], it has been shown that such equation is structurally similar to a two-layer feedforward network and hence can be implemented by parallel hardware and incorporated with learning ability. The resulted FRNS model therefore offers an efficient platform to capture the set-to-set mapping defined by the available fuzzy rules. Unfortunately, the rule storage property of the model is still poorly understood and its ability to perform structure identification when it is applied to model nonlinear dynamic systems is still lacking.

3.4 Concluding Remarks

In this chapter, various FNS models under the categories of fuzzification of neural networks and layered network implementation of fuzzy systems have been briefly reviewed and their integration methodologies and advantages have been highlighted. It can be concluded that the current investigations along these two major approaches to synthesize FNS models have been quite extensive and the subject has reached a certain level of maturity. In order to advance its development, consolidating the theoretical understandings of existing FNS models seems to be an indispensable step. As such, the poorly established storage capacities of the FAM and FRNS models are re-examined in Chapters 5 & 6 respectively. The results not only have led to a better understanding of these two models, they have also contributed to the development of high capacity encoding schemes which in turn reduce the storage requirements of the models.

Despite the considerable efforts in synthesizing the second category FNS models, the learning algorithms derived are mainly based on the back-propagation algorithm which is excellent in learning the layered network's interconnection parameters but not the network structure itself. It is particularly insufficient when these models are applied to identify nonlinear dynamic systems where structural parameters like system orders, time delays, and inference operators, have to be determined. Thus, algorithms capable for both structure and parameter identifications are much desirable. In view of such a need, an identification algorithm possessing this capability for the FRNS model is developed in Chapter 7. The methodology developed is expected to be applicable to other FNS models.

Although there have been numerous attempts to fuzzify neural networks, such an approach to synthesize FNS models should be considered as a continuous effort in the development of FNS. It is because new neural network models are continuously proposed by researchers and there is always a possibility to fuzzify them such that their performances are enhanced. As such, we have included such kind of study in this research. In particular, due to their simplicity in implementation and popularity in applications, we work on the fuzzification of competitive learning networks, aiming at addressing two important problems of the model. The values and derivations of the fuzzy competitive learning algorithms are described in the next chapter.

Chapter 4

Fuzzification of Competitive Learning Networks

4.1 Introduction

Due to its simplicity, efficient hardware implementation, and adaptability to changing environment, competitive learning (CL) has been frequently suggested as an alternative approach to various sophisticated problems such as image segmentation [111], vector quantization [1], and pattern classification [61]. Generally speaking, the goal of competitive learning is to cluster or categorize the training patterns into representative groups such that patterns within a cluster are more similar to each other than patterns belonging to different clusters. It differs from traditional clustering algorithms that it is an on-line algorithm where adapting to changing environment is possible. Based on a "learn only if it wins", i.e. winner-take-all principle, neurons in a CL network compete to move to the centroids of similar patterns and consequently uncorrelated patterns will be encoded by different neurons. As mentioned in Chapter 1, such an "exclusive" learning mechanism suffers from two major shortcomings, namely, neuron underutilization [1,46,102] and inefficient use of closeness information [28]. If they can be overcome, better quality of cluster centroids found by competitive learning is expected. This part of the research was mainly motivated by two observations. One is that the closeness information can be formulated as fuzzy membership and incorporated in the learning process to achieve graded control rather than crisp control of the competitive process. The other observation is that the basic form of CL is nothing more than an on-line version of the hard *c*-means algorithm. In view of the superiority of the fuzzy *c*-means algorithm [6,36], a corresponding fuzzified CL procedure for this algorithm can be derived with the similar process and should perform better than its crisp counterpart.

In this chapter, a fuzzy competitive learning (FCL) paradigm adopting a principle of "learn according to how well it wins" [19] is developed and based upon which three existing competitive learning algorithms, namely, the unsupervised competitive learning [49], learning vector quantization [61], and frequency sensitive competitive learning [1] are fuzzified to form a class of FCL algorithms. The three existing CL algorithms are firstly reviewed in the following section. Section 4.3 describes the FCL paradigm, derives the three FCL algorithms, and comments on their algorithmic advantages. The stability of FCL algorithms is discussed in Section 4.4 and a scheme to control the fuzziness parameter of the algorithms [16] is described in the subsequent section. Through the concept of "clusters as rules", the interpretation of FCL networks [18] is introduced in Section 4.6. The performance of the proposed FCL algorithms is reported, discussed and compared with the crisp CL performance through various data sets in Section 4.7.

4.2 Crisp Competitive Learning Algorithms

Depending on the nature of applications, the available training patterns could be unlabelled or labelled and hence unsupervised and supervised competitive learning algorithms were proposed accordingly. In this section, three different CL algorithms of which two are unsupervised and one is supervised are reviewed. Among the two unsupervised algorithms, one was chosen from the so-called win-rate dependent type CL [1,29] which has been frequently proposed to overcome the neuron underutilization problem. For the sake of clarity, a general CL algorithm is described below and then the three CL algorithms are described as variations.

```
General Crisp Competitive Learning Algorithm
Step 1 Initialization
    - Set the number of competing neurons $c$.
    - Initialize the neuron's parametric vectors $\vec{m}_j(0); j=1,...,c$.
Step 2 Distance Computation
    - For input pattern $\vec{x}_k$, compute the distances $D_{ki} = d(\vec{x}_k, \vec{m}_i(t))$ for all
      competing neurons $i$. Different algorithms may employ different distance
      metric to suit their applications. However, Euclidean distance metric is the
      most popular one.
Step 3 Competition
    - Determine the winning neuron $j$ having $D_{kj} = \min_i D_{ki}$
Step 4 Learning
    - Update the winning neuron's parametric vector as
          $\vec{m}_j(t+1) = \vec{m}_j(t) + \alpha(t) I_j(t)[\vec{x}_k - \vec{m}_j(t)]$
      where $\alpha(t)$ is the learning rate which is usually monotonically decreasing
      and $I_j(t)$ is a scaling function specifying the sign and magnitude of the
      difference vector being updated by neuron $j$. In fact, existing CL algorithms
      are mainly different in the formulation of $I_j(t)$
Step 5 Termination :
    - Repeat steps (2)-(4) until the terminating criterion is met.
```

Such a crisp CL algorithm is usually associated with a layered feedforward network. As shown in Fig.4.1, the bottom two layers are used to implement the distance computation step while the winner-take-all net is used to determine the winner, i.e., the neuron nearest to the input pattern.

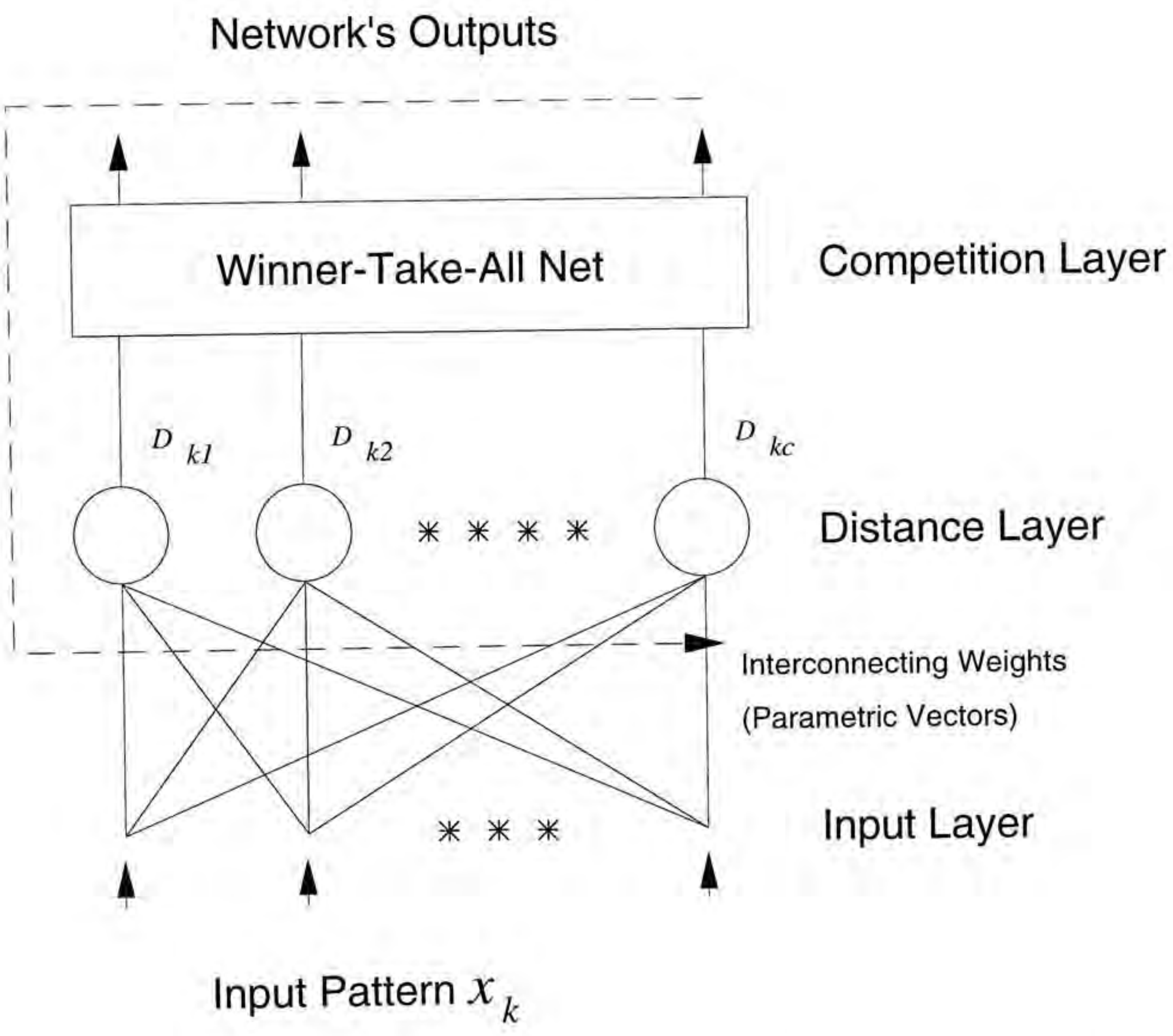


Figure 4.1 A Crisp Competitive Learning Network. The distance layer computes the distances D_{ki} between input pattern x_k and competing neuron's parametric vectors while the competition layer determines which neuron is nearest to the input pattern, i.e. smallest D_{ki}, and produces a +1 output for it and zero outputs for the losing neurons. The network outputs are used to adjust the winning neuron's parametric vector.

4.2.1 Unsupervised Competitive Learning Algorithm

The unsupervised competitive learning (UCL) algorithm is the simplest form of CL. As mentioned in the introductory section, it is only an on-line version of the hard c-means algorithm. Recall that the goal of hard c-means algorithm is to group the input patterns into c clusters such that the total Euclidean distances between input patterns and their nearest cluster centroids are minimized. Suppose that there are N training patterns $\{\vec{x}_k\}$ and c clusters with parametric vectors $V=\{\vec{m}_i | i=1,\cdots,c\}$. The problem can be formulated as minimizing the within groups sum of squared error

$$J(W,V)=\sum_{k=1}^{N} J_k(W,V)=\sum_{k=1}^{N}\sum_{i=1}^{c} w_{ki}D_{ki}^2 \tag{4.1}$$

subjected to the constraints

$$\sum_{i=1}^{c} w_{ki}=1 \quad \forall k \tag{4.2}$$

$$w_{ki}\in\{0,1\} \quad \forall k,i \tag{4.3}$$

where

$$W=\{w_{ki} | k=1,\cdots,N; i=1,\cdots,c\} \tag{4.4}$$

$$D_{ki}=\|\vec{x}_k-\vec{m}_i\| \tag{4.5}$$

$$w_{ki}=\begin{cases}1 & if\ D_{ki}=\min_{l} D_{kl}\\ 0 & \text{otherwise}\end{cases} \tag{4.6}$$

By evaluating $\partial J(W,V)/\partial \vec{m}_i=0$, the updating rule for cluster centroids $\vec{m}_i$ of the hard c-means algorithm would result. Here w_{ki} is the crisp membership of pattern $\vec{x}_k$ to cluster i and its binary value is determined by the nearest-neighbor rule in eq.(4.6). Consider now that the error function being minimized is half of the pattern based objective function, i.e., $\frac{1}{2}J_k(W,V)$. The UCL law can be readily obtained by the gradient descent method, viz

$$\begin{aligned}\vec{m}_j(t+1) &= \vec{m}_j(t) - \varsigma \frac{\partial \frac{1}{2} J_k(W,V)}{\partial \vec{m}_j(t)} \\ &= \vec{m}_j(t) + \varsigma w_{ki}\left[\vec{x}_k - \vec{m}_j(t)\right] \\ &= \begin{cases} \vec{m}_j(t) + \varsigma\left[\vec{x}_k - \vec{m}_j(t)\right] & \textit{if } D_{ki} = \min\limits_l D_{kl} \\ \vec{m}_j(t) & \textit{otherwise} \end{cases} \end{aligned} \tag{4.7}$$

Hence the UCL algorithm follows the general CL algorithm with the scaling function

$$I_j(t) = 1 \tag{4.8}$$

and distance function

$$D_{ki} = \|\vec{x}_k - \vec{m}_i\|. \tag{4.9}$$

As mentioned, the UCL algorithm suffers from the problem of neuron underutilization [1,46,102]. A simple example is shown in Fig.4.2 where neurons with parametric vectors far away from the training pattern as those in the upper-right corner never win and learn and therefore are underused.

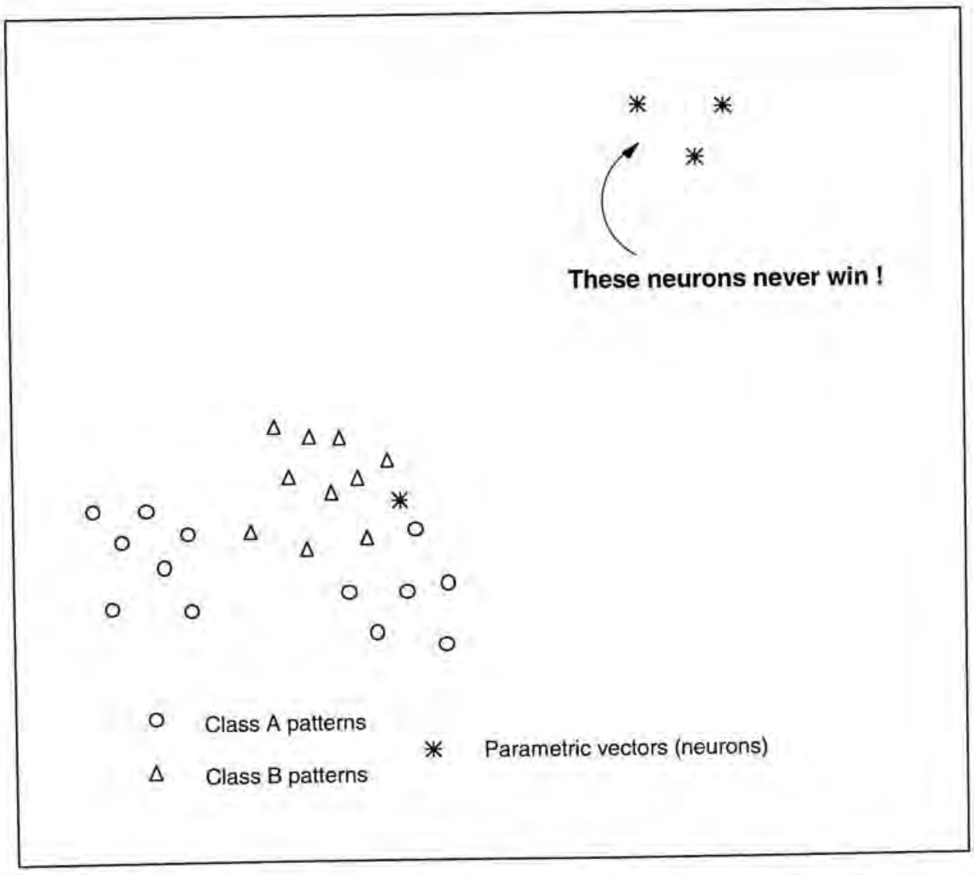


Figure 4.2 A Neuron Underutilization Example of UCL

4.2.2 Learning Vector Quantization Algorithm

Kohonen's learning vector quantization (LVQ) is a supervised type CL algorithm that works on labelled training patterns and aims at minimizing the misclassification rate. Competing neurons have to be labelled a prior to a specific class and the number of competing neurons should be equal to or greater than the number of classes considered. The LVQ algorithm may well be described by the general CL algorithm with the scaling function defined as

$$I_j(t) = \begin{cases} +1 & if\ Cl(\vec{m}_j) = Cl(\vec{x}_k) \\ -1 & if\ Cl(\vec{m}_j) \neq Cl(\vec{x}_k) \end{cases} \tag{4.10}$$

where $Cl(\cdot)$ denotes the *class of* operator. Thus, LVQ adopts a reinforce-or-punish learning principle in the competitive process so as to move the winning neuron's parametric vector $\vec{m}_j$ closer to the class centroid if the current training pattern $\vec{x}_k$ is correctly classified and to move $\vec{m}_j$ out of the misclassified region if $\vec{x}_k$ is wrongly classified. To determine the winner of competition, the Euclidean distance metric is usually adopted by LVQ. After training, the network is simply a nearest-neighbor classifier where the neuron's parametric vectors are the class prototypes.

The LVQ algorithm has been applied to numerous tasks. For examples, it has been used to fine tune the self-organizing feature map for pattern classification [63] and as a preprocessing stage for the probabilistic neural network [10] and the multilayer perceptron [108]. Despite its success, the algorithm is not free from problems. One is related to its convergence [4]. A simple example is depicted in Fig.4.3 where the parametric vectors $\vec{m}_1$ and $\vec{m}_2$ are near to the patterns of different class. According to eq.(4.10), $\vec{m}_1$ and $\vec{m}_2$ will be moved away from the patterns as training proceeds. Hence, they are pushed towards $+\infty$ and $-\infty$ respectively and divergence happens. In fact, the convergence of the LVQ algorithm requires the initial parametric vectors to be close to the solutions [4]. Therefore, the classical *c*-means algorithm and samples picked from the training set are frequently used to initialize the parametric vectors [62]. The other problem of LVQ is again neuron underutilization. As demonstrated by the example in Fig.4.4, all the class A parametric vectors are not close enough to win and learn the three class A training patterns on the right. Hence, those patterns will be misclassified and the network resource is underused. In fact, this is a problem common to winner-take-all type competitive learning algorithms and its occurrence essentially depends on the initial distribution of the parametric vectors.

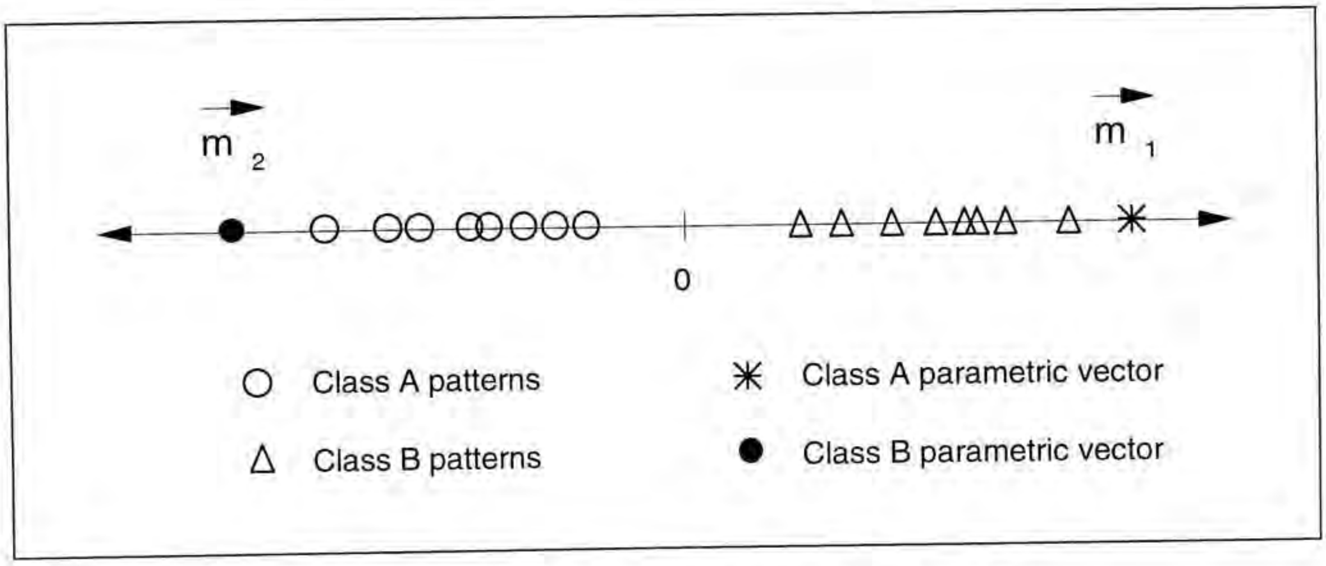


Figure 4.3 A Divergent Example of LVQ

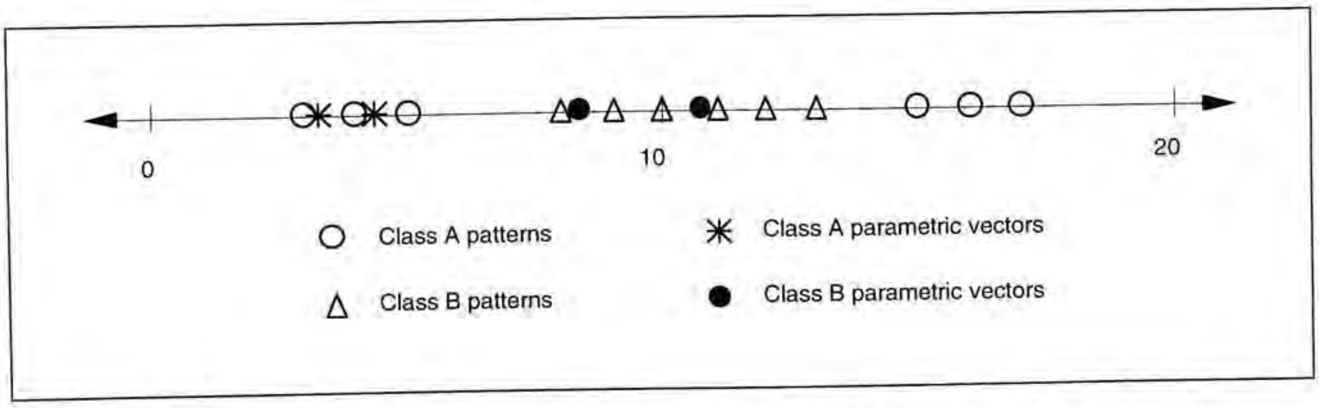


Figure 4.4 A Neuron Underutilization Example of LVQ

4.2.3 Frequency Sensitive Competitive Learning Algorithm

The frequency sensitive competitive learning (FSCL) algorithm [1] was designed to address the neuron underutilization problem of the UCL algorithm. The algorithm simply incorporates the win-rate information into the distance metric so as to ensure that every neuron has a fair chance to win and learn. Specifically, let $n_i(t-1)$ be the number of times that neuron *i* won in the past *t*-1 learning competitions. The modified distance function is defined as

$$D_{ki} = \|\bar{x}_k - \bar{m}_i(t)\| \cdot n_i(t-1) \tag{4.11}$$

Thus the more neuron *i* wins, the larger $n_i(t-1)$ is and the less likely it will win again. This gives other neurons a better chance to win in the coming competitions. Such a simple modification not only addresses the neuron underutilization problem, it also forces the competing neurons to learn approximately equal number of times. As pointed out by the authors of FSCL in [68], the algorithm is nothing but an implementation of Grossberg's conscience principle [46], and is similar to the conscience method of DeSieno [29].

4.3 Fuzzy Competitive Learning

As described in the previous section, the rationale of CL is "learn only if it wins". Under competition, only one neuron will win and learn the current training pattern. Obviously, the concept *win* in this setting is a crisp one and has a very clear cut boundary. By considering *win* as a fuzzy set, every neuron to a certain degree wins, depending on its distance to the current training pattern. Hence it has to learn according to its *win* membership during the competition. In this way, a "learn according to how well it wins" fuzzy competitive learning (FCL) paradigm [19] results, based upon which a class of FCL algorithms can be developed. In this section, we present the derivation of a fuzzy version of the three CL algorithms described in Section 4.2. Furthermore, we comment on the advantages of the derived fuzzy algorithms. Again for the sake of clarity, a general FCL algorithm is first described below and then the three FCL algorithms are described as variations.

General Fuzzy Competitive Learning Algorithm

```
Step 1 Initialization
        - Set the number of competing neurons c.
        - Initialize the neuron's parametric vectors $\vec{m}_j(0)$; $j=1,\dots,c$.
Step 2 Distance Computation
        - For input pattern $\vec{x}_k$, compute the distances $D_{ki}=d(\vec{x}_k,\vec{m}_i(t))$ for all
          competing neurons i. Different algorithms may employ different distance
          metric to suit their applications.
Step 3 Fuzzy Competition
        - Based on the computed distances $D_{ki}$, determine the win membership $\mu_{ki}$ for
          each competing neuron i
Step 4 Fuzzy Learning
        - Update EACH competing neuron's parametric vector as
              $\vec{m}_i(\mu_{ki})=\vec{m}_i(t)+\alpha(t)Z_i(\mu_{ki})[\vec{x}_k-\vec{m}_i(t)]$
          where $\alpha(t)$ is the learning rate and $Z_i(\mu_{ki})$ is a fuzzy scaling function of $\mu_{ki}$
          specifying the sign and magnitude of the difference vector being updated by
          neuron i
Step 5 Termination
        - Repeat steps (2)-(4) until the terminating criterion is met.
```

The implementation of the above algorithm is depicted in Fig.4.5. It can be seen that the distance layer is the same as that of the crisp competitive learning network in Fig.4.1 while the fuzzy competition layer here takes use of the computed distances D_{ki} to determine the *win* memberships μ_{ki} in [0,1] of the competing neurons. The membership outputs would be fed back to the corresponding neurons to update the parametric vectors.

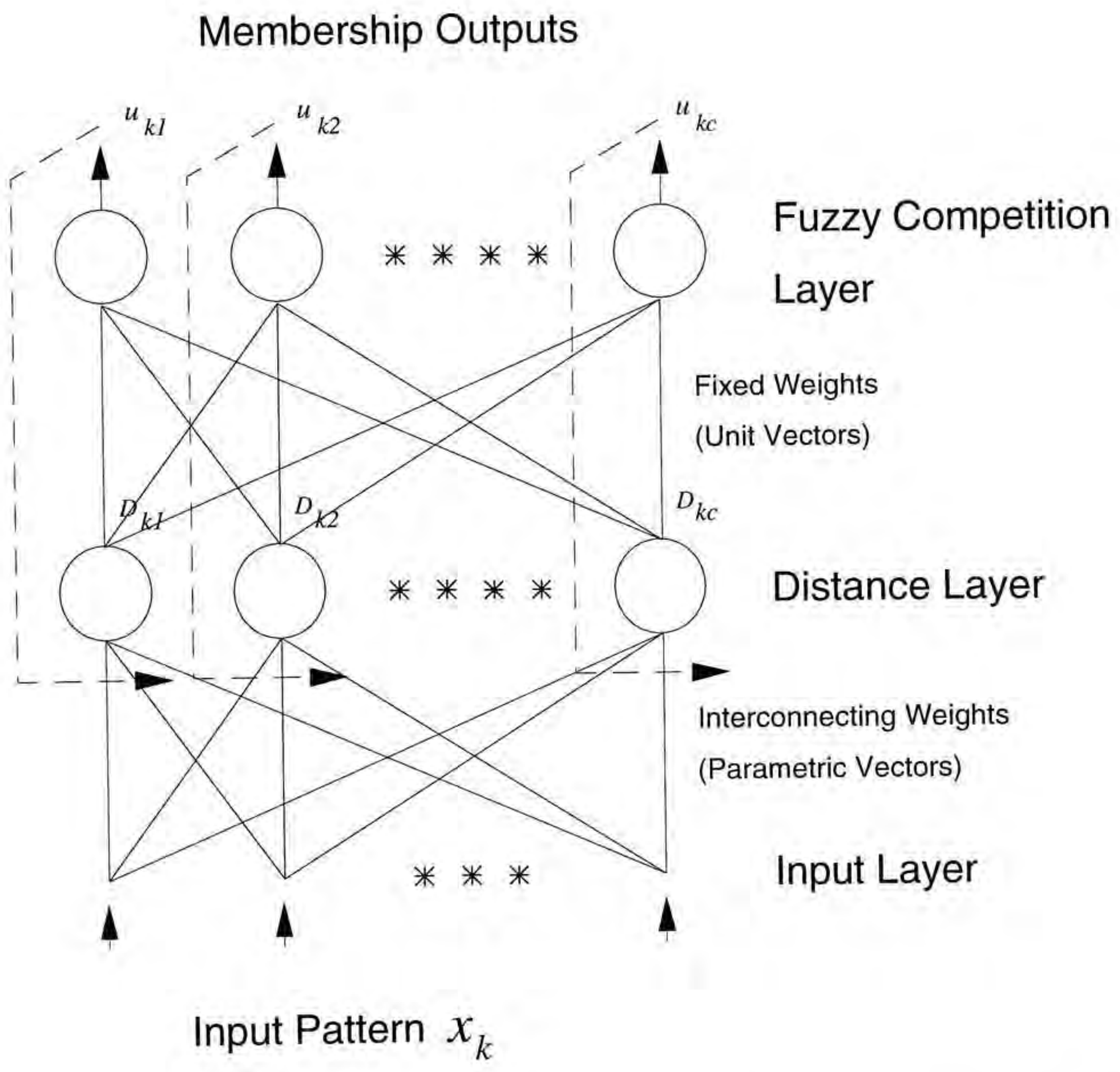


Figure 4.5 A Fuzzy Competitive Learning Network. The distance layer is the same as that of the crisp competitive learning network in Fig.4.1 and the fuzzy competition layer determines the *win* memberships μ_{ki} in [0,1] of the competing neurons from the computed distances D_{ki}. The membership outputs would be used to update the parametric vectors of the corresponding neurons.

4.3.1 Unsupervised Fuzzy Competitive Learning Algorithm

In Section 4.2.1, we have already shown that the UCL law can be derived from the objective function of hard *c*-means algorithm using the steepest descent technique. Likewise, a fuzzy version of UCL law can be obtained from the fuzzy *c*-means algorithm as follows. It will be shown that the derived unsupervised fuzzy competitive learning (UFCL) algorithm implements the FCL paradigm described earlier. Recall from [6] that the fuzzy *c*-means algorithm is to find optimal cluster centers $V = \{\vec{m}_i | i = 1, \cdots, c\}$ and fuzzy *c*-partition $U = \{\mu_{ki} | k = 1, \cdots, N; i = 1, \cdots, c\}$, i.e., membership function, which minimize the sum of fuzzy membership weighted squared error

$$J^m(U,V) = \sum_{k=1}^{N} J_k^m(U,V) = \sum_{k=1}^{N} \sum_{i=1}^{c} (\mu_{ki})^m D_{ki}^2 \tag{4.12}$$

subjected to

$$\sum_{i=1}^{c} \mu_{ki} = 1 \quad \forall k \tag{4.13}$$

$$\mu_{ki} \in [0,1] \quad \forall k, i \tag{4.14}$$

where

$$D_{ki} = \|\vec{x}_k - \vec{m}_i\| \tag{4.15}$$

and *m* is a real quantity greater than one and its value specifies the nature of clustering, ranging from absolute hard clustering at *m*=1 to increasingly fuzzier clustering as *m* increases. Here, the objective function consists of variables *U* and *V*. By fixing *U* and applying the gradient descent method to the pattern based objective function $J_k^m(V)$, the UFCL law

$$\Delta\vec{m}_i(t) = \varsigma(\mu_{ki})^m [\vec{x}_k - \vec{m}_i(t)] \tag{4.16}$$

is resulted. Similarly, by fixing *V* and applying the method of Lagrangian multiplier to *U*, the membership updating rule

$$\mu_{ki} = \frac{\left(\frac{1}{D_{ki}}\right)^{\frac{2}{m-1}}}{\sum_{l=1}^{c}\left(\frac{1}{D_{kl}}\right)^{\frac{2}{m-1}}} \tag{4.17}$$

is obtained. Hence, the UFCL algorithm follows the general fuzzy CL algorithm with membership μ_{ki} defined by eq.(4.17) and

$$Z_i = (\mu_{ki})^m \tag{4.18}$$

$$D_{ki} = \left\|\vec{x}_k - \vec{m}_i(t)\right\| \tag{4.19}$$

From the derived algorithm, it can be seen that every neuron is responsible for learning the current training pattern. However, closer the neuron is to the pattern, larger its win membership μ_{ki} is and more it has to learn. With such UFCL algorithm, the two shortcomings of crisp CL paradigm are addressed. Far away neurons can now have a chance to move to the pattern regions. This is similar in spirit to Rumelhart and Zipser's leaky learning [102] which allows the losing neurons to learn a small amount of the difference vector unconditionally. On the other hand, closeness information has already been employed by the proposed algorithm to compute the membership values which in turn are used to control the learning process. It is expected that the UFCL algorithm will converge to better solutions than the UCL algorithm. These two advantages will be demonstrated by the simulation results reported in Section 4.7.

4.3.2 Fuzzy Learning Vector Quantization Algorithm

Being inspired by the UFCL algorithm, we have opted for an objective function minimization approach to derive a fuzzy LVQ (FLVQ) algorithm [17]. Although LVQ was developed without resorting to minimize any objective function, two objectives of the algorithm could be identified. One is to maximize the classification rate and the other is to minimize the distances between training patterns and neuron's parametric vectors. Taking these two criteria into considerations, we have defined the FLVQ objective function as follows. For the sake of clarity, we assume here that the number of competing neurons c is equal to the number of pattern classes. Then, μ_{ki} is the class i membership of the kth input pattern. The goal of the FLVQ algorithm therefore is to minimize the following fuzzy objective function

$$Q^m(U,V)=\sum_{k=1}^{N} Q_k^m(U,V)=\sum_{k=1}^{N}\sum_{i=1}^{c}[t_{ki}-(\mu_{ki})^m]D_{ki}^2 \tag{4.20}$$

subjected to the constraints

$$\sum_{i=1}^{c}\mu_{ki}=1 \quad \forall k \tag{4.21}$$

$$\mu_{ki}\in[0,1] \quad \forall k,i \tag{4.22}$$

where $t_{ki}\in\{0,1\}$ is the target membership value of input pattern $\vec{x}_k$ for class i neuron. As in the case of UFCL, the first constraint is essential for the validity of the objective function and to avoid the trivial solution where Q^m is minimized by assigning all memberships to 1. Similarly, without the fuzziness parameter m, the solution of applying Lagragian multiplier method to μ_{ki} will be a null one. Repeating the derivation steps of the UFCL algorithm, we have the learning law and membership updating rule for the FLVQ algorithm as

$$\Delta\vec{m}_i(t)=\varsigma[t_{ki}-(\mu_{ki})^m][\vec{x}_k-\vec{m}_i(t)] \tag{4.23}$$

and

$$\mu_{ki}=\frac{\left(\dfrac{1}{D_{ki}}\right)^{\frac{2}{m-1}}}{\sum_{l=1}^{c}\left(\dfrac{1}{D_{kl}}\right)^{\frac{2}{m-1}}} \tag{4.24}$$

respectively. It can be seen that every neuron i has to learn in each pattern presentation and graded correction is introduced. Conforming to the general fuzzy CL procedure, the FLVQ algorithm adopts the membership function μ_{ki} defined by eq.(4.24), the distance computation D_{ki} defined by the squared Euclidean one, and the fuzzy scaling function

$$Z_i(\mu_{ki}) = t_{ki} - (\mu_{ki})^m \tag{4.25}$$

In fact, the derived algorithm implements the FCL paradigm as well as the LVQ's reinforce-or-punish learning principle. Specifically, when $t_{ki} = 1$, Z_i will be positive and the quantity that neuron i will move to the training pattern will depend on the discrepancy between the target and membership values to the training pattern. Similarly, when $t_{ki} = 0$, Z_i will be negative and neuron i will move away from the training pattern accordingly. Thus, the differences $[t_{ki} - (\mu_{ki})^m]$ reduce as learning proceeds until the neurons converge to the class centroids where the neuron's membership outputs would be large for patterns of the same class and small for patterns of difference class.

Recall that the number of competing neurons has been assumed to be equal to the number of pattern classes. Such an assumption can be removed by defining the distance between the input pattern and the class i neurons as

$$D_{ki} = \min_{j \in S_i} \left\| \vec{x}_k - \vec{m}_{ij} \right\| \tag{4.26}$$

where S_i is the index set of the competing neurons for pattern class i and $\vec{m}_{ij}$ is the jth competing neuron's parametric vector for pattern class i. Thus, the FLVQ learning law, corresponding to the new objective function

$$Q^m(U,V) = \sum_{k=1}^{N} Q_k^m(U,V) = \sum_{k=1}^{N} \sum_{i=1}^{c} [t_{ki} - (\mu_{ki})^m] \min_{j \in S_i} \left\| \vec{x}_k - \vec{m}_{ij} \right\|^2 \tag{4.27}$$

would become

$$\vec{m}_{ij}(t+1) = \vec{m}_{ij}(t) + \alpha(t) I_{ij} [t_{ki} - (\mu_{ki})^m][\vec{x}_k - \vec{m}_{ij}(t)] \tag{4.28}$$

where

$$I_{ij} = \begin{cases} 1 & if \left\| \vec{x}_k - \vec{m}_{ij} \right\| \le \left\| \vec{x}_k - \vec{m}_{il} \right\| \forall l \in S_i \\ 0 & otherwise \end{cases} \tag{4.29}$$

This modification can be realized by introducing a MIN layer between the distance computation layer and the fuzzy competition layer of Fig.4.5. As shown in Fig.4.6, among the pool of competing neurons labelled for pattern class *i*, only the one closest to the input pattern is chosen by the *i*th MIN neuron to undergo the fuzzy competitive learning process, i.e. class membership computations and parametric vector adaptations. Thus the pool of competing neurons would learn the corresponding class patterns in a cooperative manner. With the proposed algorithm, the divergent example in Fig.4.3 can be solved because the punishment force caused by different class patterns is always smaller than the reinforcement force caused by same class patterns. Furthermore, the FLVQ algorithm is excellent in avoiding neuron underutilization, as demonstrated by the simulation results reported later.

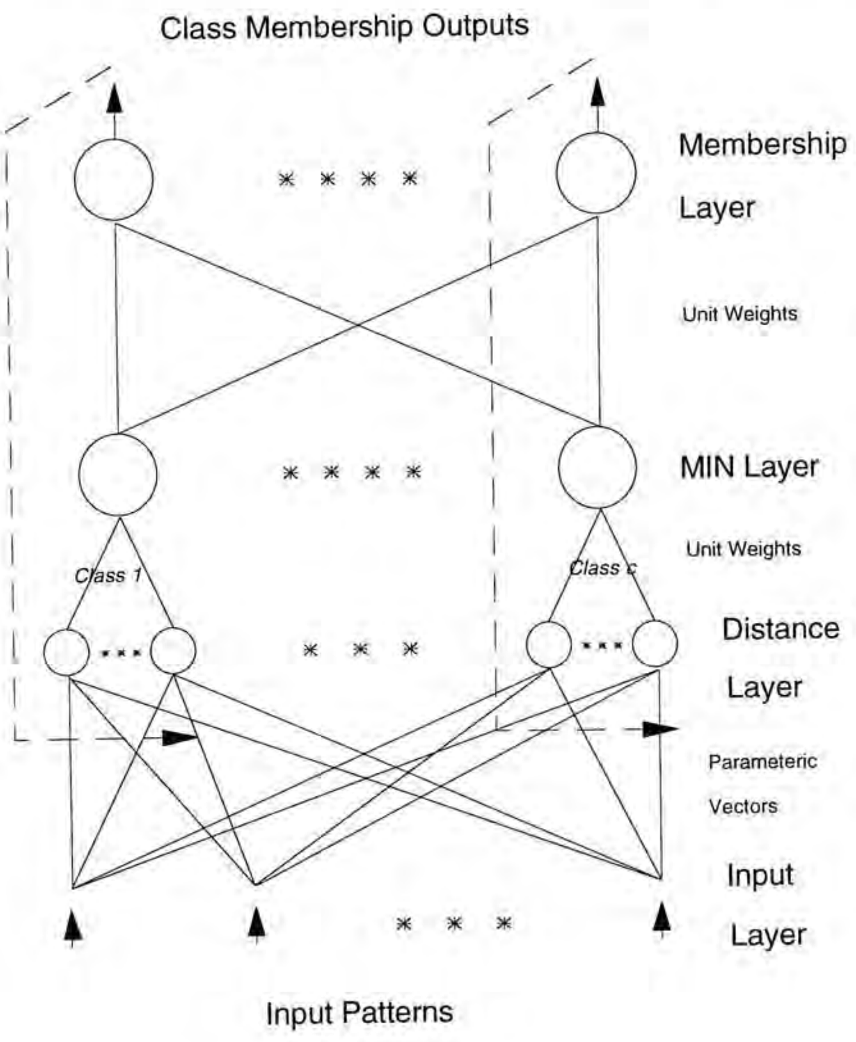


Figure 4.6 A FLVQ Network with Multiple Neurons per Pattern Class

4.3.3 Fuzzy Frequency Sensitive Competitive Learning Algorithm

Central to the FSCL algorithm is the use of win-rate to bias the competition such that it is fairer. With the concept *win* being fuzzified, the fuzzy FSCL (FFSCL) algorithm proposed here takes use of the corresponding win frequency obtained from the UFCL algorithm to modulate the competition. Specifically, the FFSCL algorithm employs a modified distance function for the UFCL algorithm which is defined as

$$D_{ki} = \left\| \vec{x}_k - \vec{m}_i(t) \right\| \cdot u_i(t-1) \tag{4.30}$$

where

$$u_i(t-1) = \sum_{l=1}^{t-1} (\mu_{li})^m \tag{4.31}$$

is the accumulated win membership of neuron i for the past t-1 competitions. Obviously, the proposed algorithm is simply a conceptual extension of the original FSCL algorithm. In fact, one may argue the motivation to fuzzify the FSCL algorithm as we have already mentioned that the UFCL algorithm does have the ability to resolve the neuron underutilization problem. Simulation results reported later however will demonstrate that the two approaches are complementary rather than redundant in addressing the problem.

4.4 Stability of Fuzzy Competitive Learning

As for the crisp CL algorithms and most on-line neural network learning algorithms, one may suspect the convergence and stability of the FCL algorithms when the input presentation order is biased. This could be true if the order is extremely biased. In this regard, two alternatives to implement the FCL algorithms are suggested. One is to apply the learning law in batch mode, i.e., accumulating the changes $\Delta\vec{m}_i(t)$ for each of the available training patterns $\vec{x}_k$ before actually updating the parametric vectors, viz.

$$\Delta \vec{m}_i(T) = \varsigma \sum_{k=1}^{N} Z_i(\mu_{ki})[\vec{x}_k - \vec{m}_i(T)] \tag{4.32}$$

where T denotes the iteration index. In fact, eq.(4.32) is just the results of applying gradient descent to the training set based objective functions J^m and Q^m in eqs.(4.12) & (4.27) respectively. Hence, reaching a stable equilibrium (i.e., a local minimum) could be expected. Another implementation of the FCL algorithms for biased input presentation order is to include momentum update as in the back-propagation algorithm [101] to the learning law, i.e.

$$\begin{aligned} \Delta m_i(t) &= \varsigma Z_i(\mu_{ti})[\vec{x}_t - \vec{m}_i(t)] + \eta \Delta m_i(t-1) \\ &= \varsigma \sum_{\tau=0}^{t-1} \eta^{\tau} Z_i(\mu_{(t-\tau)i})[\vec{x}_{t-\tau} - \vec{m}_i(t-\tau)] \end{aligned} \tag{4.33}$$

where $0 \leq \eta < 1$ is the momentum term that determines the relative contribution of the current and past learning actions, the original index k of training patterns has been changed to t to refer to the training sequence. Thus the parametric vectors tend to change in the decending direction of the exponentially weighted sum of current and past gradients. Such kind of "filtering" operation is particularly suitable for irregular input presentation order. In fact, with appropriate momentum, eq.(4.33) provides a close approximation to the batch mode implementation which is a truly gradient descent procedure. Thus, the FCL algorithms would not be very sensitive to the input presentation order. These two alternatives of which the momentum implementation is suitable for on-line training and the batch mode implementation is more appropriate for off-line training can be adopted when the order of input presentation is extremely biased. Otherwise, the standard implementation is sufficient to obtain good performance.

4.5 Controlling the Fuzziness of Fuzzy Competitive Learning

A fuzziness parameter m, as in most fuzzy pattern recognition algorithms, has been introduced in the proposed FCL algorithms. From the membership function in eq.(4.17), it can be observed that for fixed distances D_{ki} the variation of membership μ_{ki} for different neuron i decreases (i.e. fuzzier) as the fuzziness parameter m increases. In the extreme case of $m \to \infty, \mu_{ki} \to 1/c$ for all k & i and therefore all the competing neurons no matter how far they are from the input pattern will be equally active in learning that pattern. Thus adopting larger fuzziness values, forcing every neuron to learn actively, should be more beneficial in avoiding neuron underutilizations. Unfortunately, adopting larger fuzziness value on the other hand may not lead to effective clustering of training data. A study [13] for the fuzzy c-means algorithm has shown that it is better to choose fuzziness value between 1.25 and 1.75. In view of such a conflict, a monotonically decreasing implementation scheme for the fuzziness parameter is proposed for the UFCL algorithm [16]. The basic idea is to use large fuzziness value (say 2.5) to avoid neuron underutilizations in the initial training stage and then gradually decrease the fuzziness to an appropriate value (say 1.5) during the training process. In [16], the following schedule has been adopted :

$$m(n) = m_f + (m_i - m_f)\exp(-\frac{5n}{n_{\max}}) \tag{4.34}$$

where m_i and m_f are the initial and final fuzziness values, n is the current training iteration index and $n_{\max}$ is the maximum number of training iterations allowed.

4.6 Interpretations of Fuzzy Competitive Learning Networks

Through the concept of "clusters as rules" [67], the trained FCL networks can be interpreted linguistically. Let's take the FLVQ network as an example [18]. Assume that there are c pattern classes and the size of the trained network is p neurons per pattern class. The ith classification rule extracted directly from the trained network has the form

$Rule^{(i)}$: If $\vec{x}_k$ is near to $\vec{m}_{i1}$ or $\vec{x}_k$ is near to $\vec{m}_{i2}$ or ... or $\vec{x}_k$ is near to $\vec{m}_{ip}$,

then $\vec{x}_k$ belongs to class i

It can be seen that the clusters characterized by the parametric vectors $\vec{m}_{ij}$ form the rule. The membership function of the antecedent has been described by eq.(4.24) where the computed membership value will be the truth value of the consequent. The logical "or" operations correspond to the MIN neurons in Fig.4.6. In fact, the MIN (of distances) neurons can be regarded as MAX (of memberships) neurons which conform to the usual implementations of the fuzzy "or" operations.

Such kind of rules however does not involve any linguistic term. (say *small*, *medium fast*, *high*) for the parametric vectors $\vec{m}_{ij} \in R^n$ and therefore does not provide true linguistic interpretation. In order to do so, the classification rules extracted from the trained network should have the form

$Rule^{(i)}$: If $\vec{x}_{k1}$ is $A_1(\vec{m}_{i1})$ and $\vec{x}_{k2}$ is $A_2(\vec{m}_{i1})$ and ... $\vec{x}_{kn}$ is $A_n(\vec{m}_{i1})$ or

$\vec{x}_{k1}$ is $A_1(\vec{m}_{i2})$ and $\vec{x}_{k2}$ is $A_2(\vec{m}_{i2})$ and ... $\vec{x}_{kn}$ is $A_n(\vec{m}_{i2})$ or . . .

$\vec{x}_{k1}$ is $A_1(\vec{m}_{ip})$ and $\vec{x}_{k2}$ is $A_2(\vec{m}_{ip})$ and ... $\vec{x}_{kn}$ is $A_n(\vec{m}_{ip})$,

then $\vec{x}_k$ belongs to class i

where $\vec{x}_{kn}$ is the nth feature of $\vec{x}_k \in R^n$ and $A_n(\vec{m}_{ip})$ is the linguistic label attached to the nth dimensional space for the parametric vector $\vec{m}_{ip}$. To obtain $A_n(\vec{m}_{ip})$, we propose to replace the constrained fuzzy membership function [69] in eq.(4.24), rewritten here as

$$\mu_{ki} = \left[\sum_{l=1}^{c} \left(\frac{D_{ki}}{D_{kl}} \right)^{\frac{2}{m-1}} \right]^{-1} \tag{4.35}$$

by the possibilistic one [69], i.e,

$$\mu_{ki} = \left[1 + \left(\frac{D_{ki}^2}{\eta_i} \right)^{\frac{1}{m-1}} \right]^{-1} \tag{4.36}$$

where η_i are positive constants to determine the squared distance at which the membership value becomes 0.5. It was adopted because the constrained fuzzy membership function, in spite of its mathematical tractability, does not provide intuitive interpretation of the resulting membership [69]. For example in Fig.4.7, it will produce very different *class A* membership values for the points *i* & *j*, even though they are equally typical of this class, i.e., equidistant from the representative neuron '*'. On the other hand in Fig.4.8, it will assign equal membership value (0.5) for the points *i* & *j* to both classes, even though point *i* should have smaller memberships than point *j* intuitively. These problems arise from the probabilistic constraint in eq.(4.21) which makes a point's membership value in a fuzzy set depend on its membership values in other fuzzy sets defined over the same universe of discourse. The memberships of the possibilistic membership functions, however, are solely a function of the distance of point from a prototypical member and therefore provide a better model for interpretation. Furthermore, the function is rotational symmetric in the multidimensional space and can be easily projected to the 1-D space for linguistic labelling. Such an interpretation process will be exemplified in the next section.

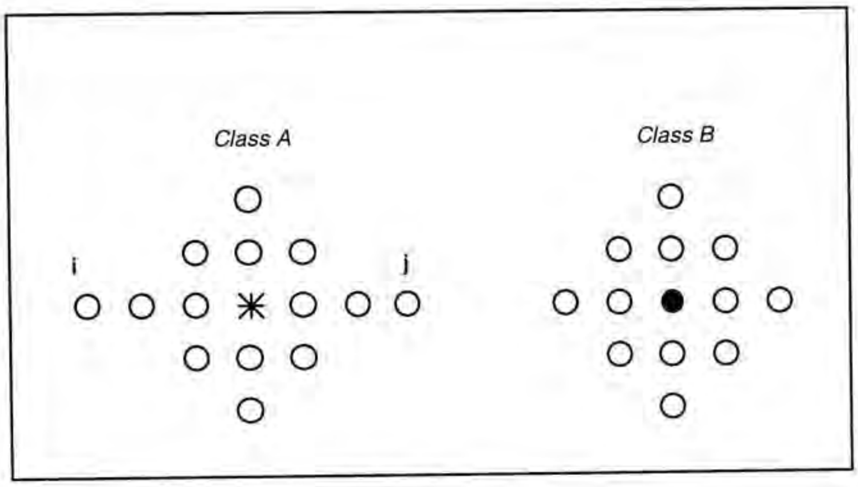


Figure 4.7 Example of data points 'o' forming two clusters with parametric vectors '*' & '●' in which the constrained fuzzy membership for points *i* and *j* are different, even though they are equally typical to *class A*.

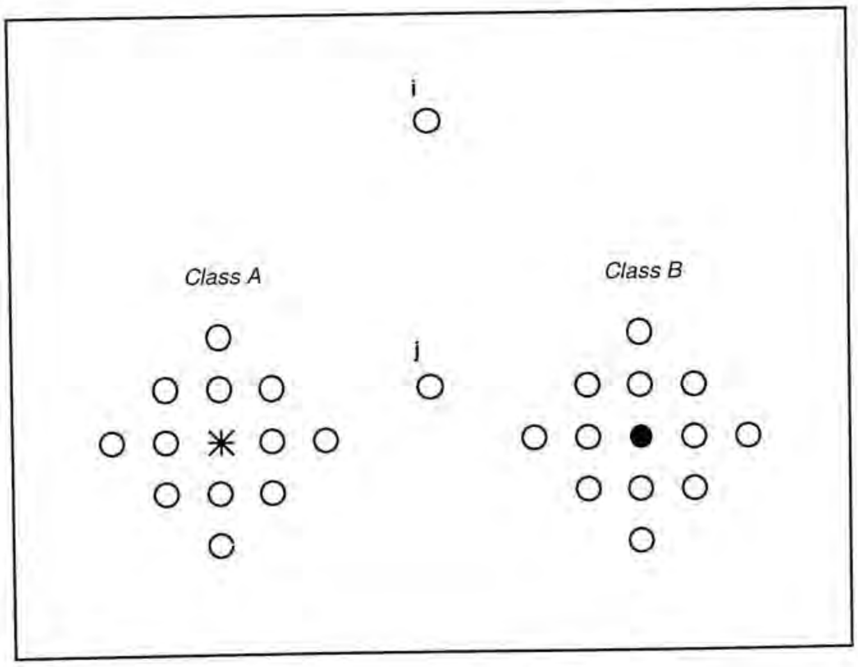


Figure 4.8 Example of data points 'o' forming two clusters with parametric vectors '*' & '●' in which the constrained fuzzy membership for two noise points *i* and *j* are equal, even though point *i* is much less representative of either cluster than point *j*.

4.7 Simulation Results

In this section, the performances of the proposed FCL algorithms as compared with those of the crisp algorithms in three pattern classification data sets are reported. To demonstrate its effectiveness, the fuzziness control scheme is adopted by the UFCL algorithm and tested on two data sets. The interpretation of trained FLVQ networks is then described via a 2-D model data set. In all simulations, the available training vectors were randomly chosen as the initial parametric vectors. Furthermore, a monotonic decreasing learning rate defined as

$$\alpha(t) = \alpha(0)\left(1 - \frac{t}{t_{\max}}\right) \tag{4.37}$$

was used, where $\alpha(0)$ is the initial learning rate and $t_{\max}$ is the maximum number of (pattern based) trainings allowed. Unless otherwise stated, $t_{\max}$ was set to 30 times the number of available training data. The fuzziness parameter values m=1.2, 1.5, 2.0 were employed to test on its sensitivity to the algorithm's performance. To ensure fair comparisons, all the training conditions, particularly the initialization step, of the fuzzy and crisp CL algorithms are exactly the same.

4.7.1 Performance of Fuzzy Competitive Learning Algorithms

***Experiment #1** : 2-D Gaussian Distributed Data Set with Distant Clusters*

In this data set, 400 training patterns are located at four gaussian clusters of which centroids are (-15,0), (-9,0), (3,3), & (3,-3) and variances are fixed to one. No clusters are overlapped and the two left clusters are far apart from the right ones. Such data set was initially designed to test on the performances of FCL algorithms in non-overlapping data set. However, simulation results showed that neuron underutilizations were frequently encountered. A typical case of it is depicted in Fig.4.9 where the UCL algorithm cannot learn the four cluster centroids by four neurons, but instead, it has converged to group the two left clusters by one neuron and encode the lower right cluster by two neurons. This is due to the random initialization cases that three neurons are initialized from the two right clusters and only one neuron is initialized from the two left clusters. With such initialization, the single competing neuron at the left will always win and learn those patterns in the two left clusters. This has left no chance for the other competing neurons to learn them.

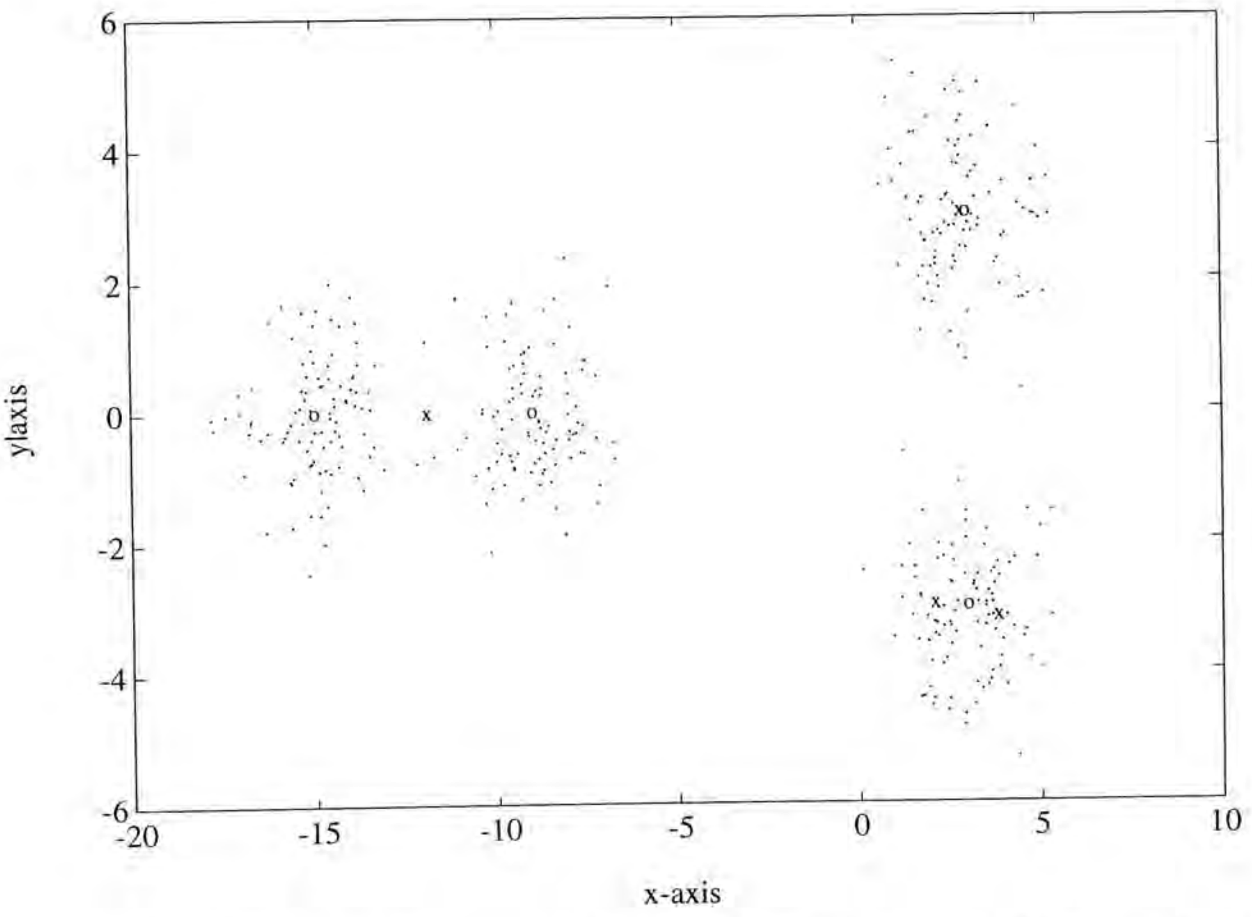
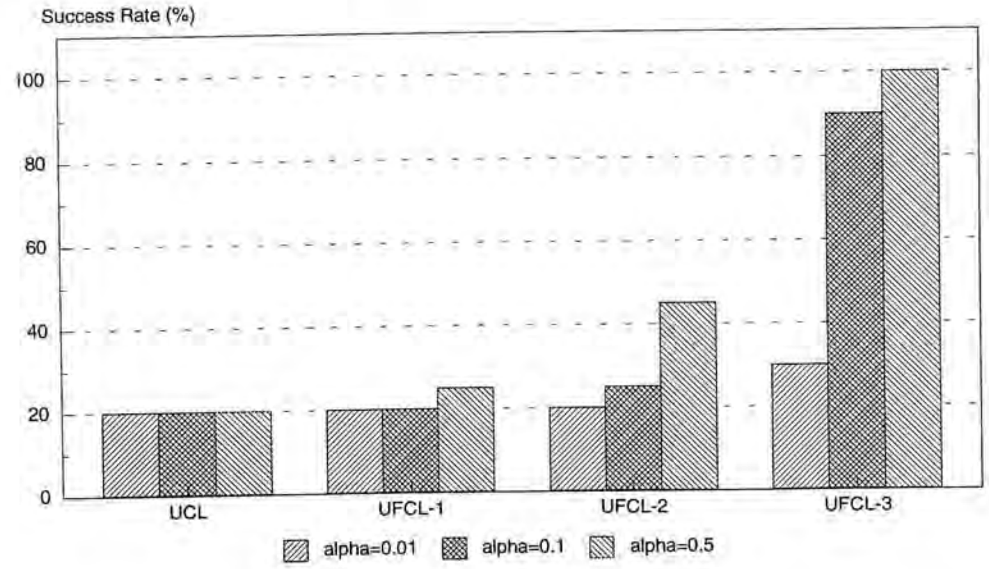
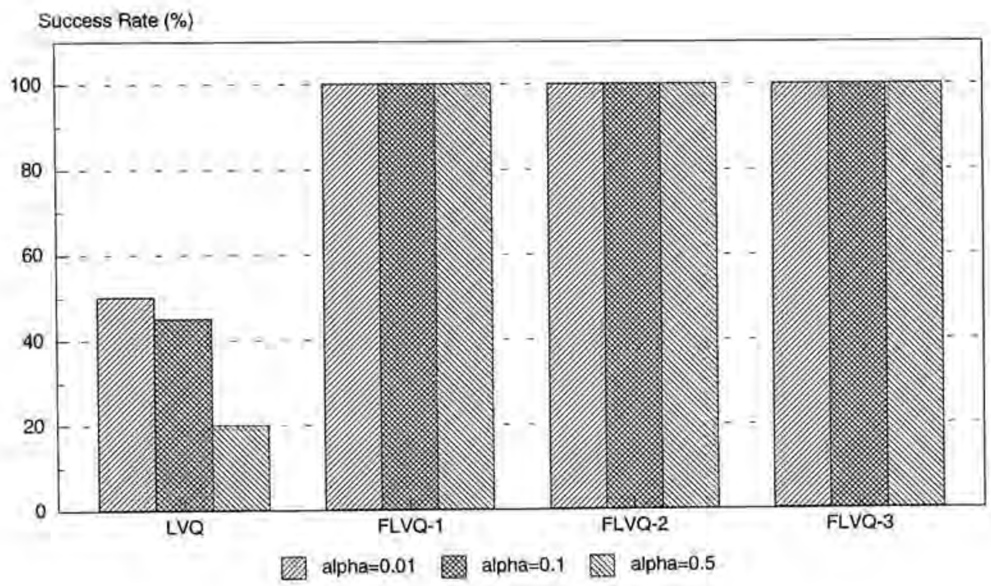


Figure 4.9 Typical Neuron Underutilization Case. The unsupervised competitive learning algorithm failed to converge to the four cluster centroids (denoted by "o"). But instead, it has converged to encode the 200 patterns in the two left clusters by one neuron (denoted by "x") and the 100 patterns in the lower right cluster by two neurons.

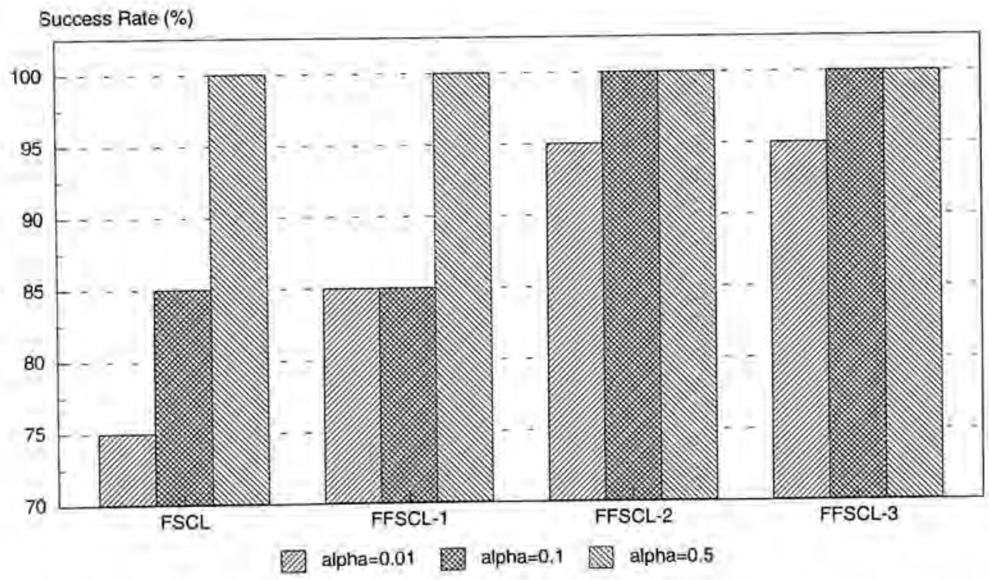
The performances, in terms of the success rate (i.e. the percentage of 20 trials converged to approximately the four cluster centroids by four competing neurons), of the three different types of CL algorithms are shown in Fig.4.10(a)-(c). Fuzzy algorithms using m=1.2, 1.5, 2.0 are symbolized by extensions "-1", "-2", and "-3" respectively. Also, three initial learning rates $\alpha(0)$=0.01, 0.1, 0.5 representing slow, medium and fast learning were used. Obviously, the proposed fuzzy algorithms converged more often to the desired solutions than the crisp ones. In particular, the FLVQ algorithm has attained 100% success rate in all cases, very robust to the parameters used. For the other two types of fuzzy algorithms, simulation results showed that adopting larger m values would have better performance. As discussed in Section 4.5, larger m values specify fuzzier clusterings which in turn will weight more on larger distances in the membership computation and produce higher membership values. Thus, far away neurons are more active to learn the current input pattern and hence neuron underutilization cases are less likely to occur. Therefore, fuzzier FCL algorithms are more beneficial in resolving the neuron underutilization problem. As shown by the learning trajectories of four competing neurons (particularly the solid line one) in Fig.4.11, the UFCL algorithm using $\alpha(0) = 0.1$ and m=2.0 has successfully escaped from the undesirable neuron underutilization case depicted previously in Fig.4.9. Although the FSCL algorithm was proposed to address the same problem, Fig.4.10(c) shows that such capability has been further enhanced by the fuzzification of the algorithm.



(a)



(b)



(c)

Figure 4.10 Success Rates of Crisp and Fuzzy Competitive Learning (CL) Algorithms in Converging to the Desired Solutions : 2-D Gaussian Data Set with Distant Cluster Locations. Parameter alpha is the initial learning rate used. The depicted crisp CL algorithms include UCL, LVQ and FSCL. The corresponding fuzzy CL algorithms are the UFCL, FLVQ, and FFSCL. Fuzzy algorithms using fuzziness parameter value m=1.2, 1.5, & 2.0 are symbolized by extensions "-1", "-2", & "-3" respectively.

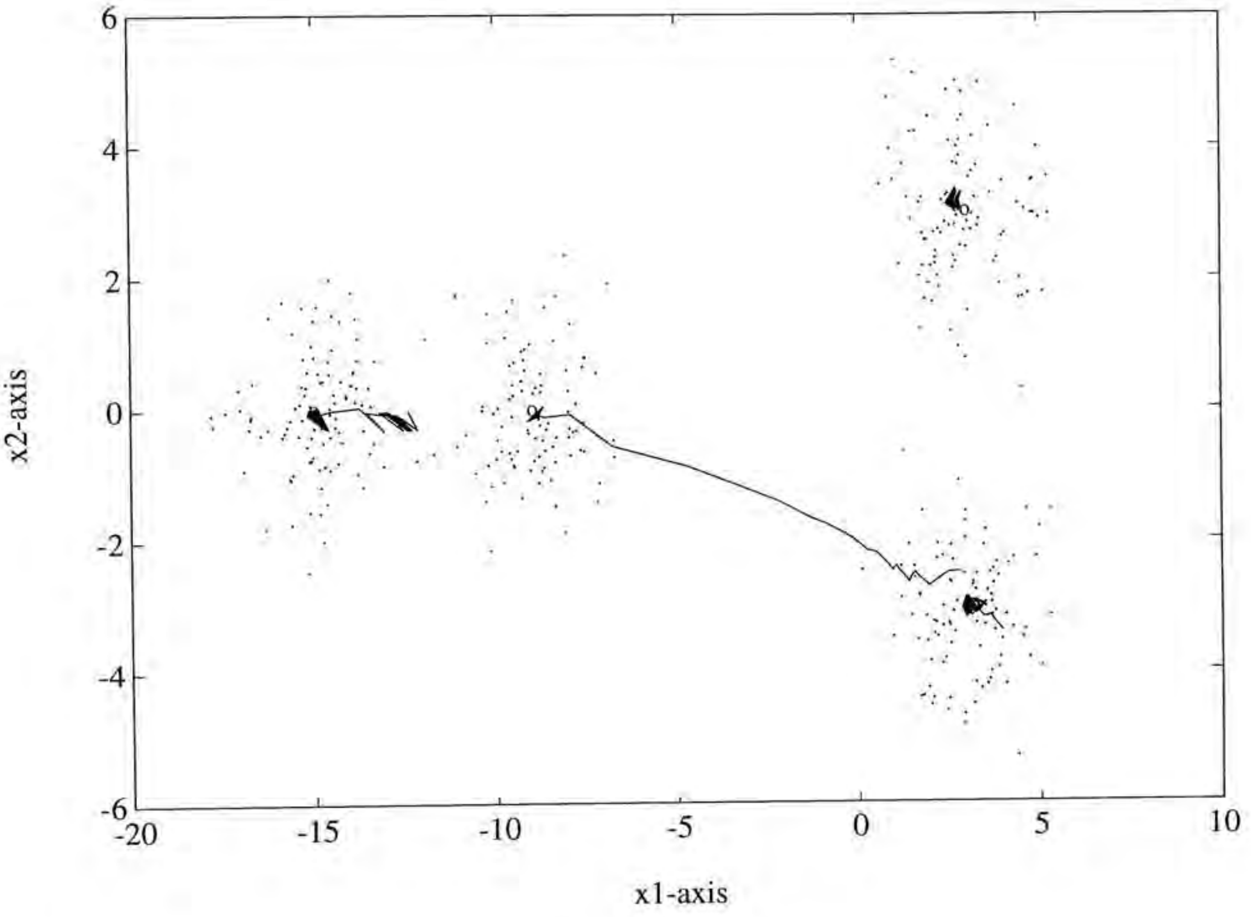
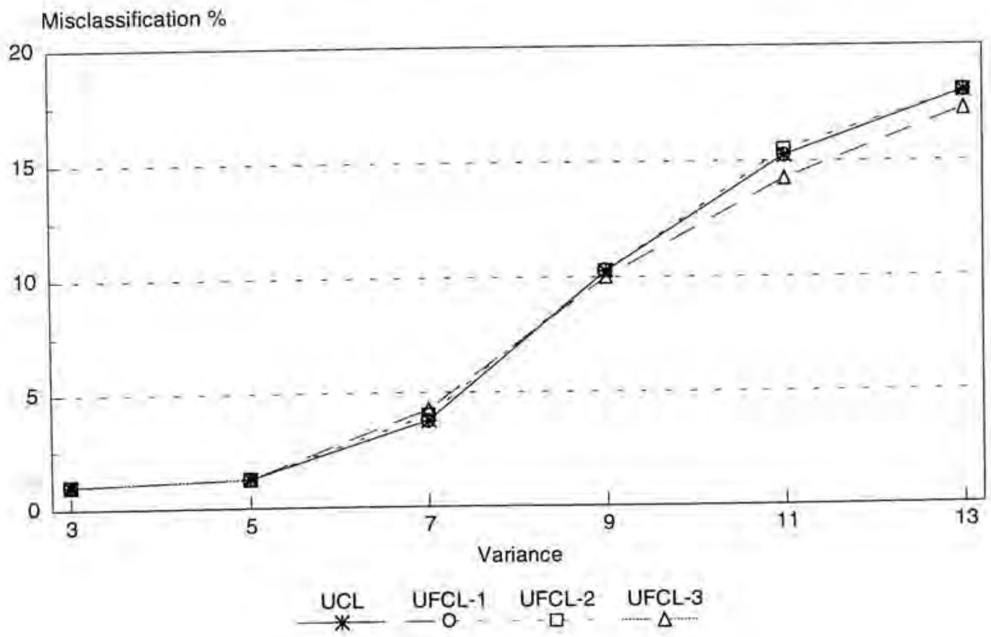


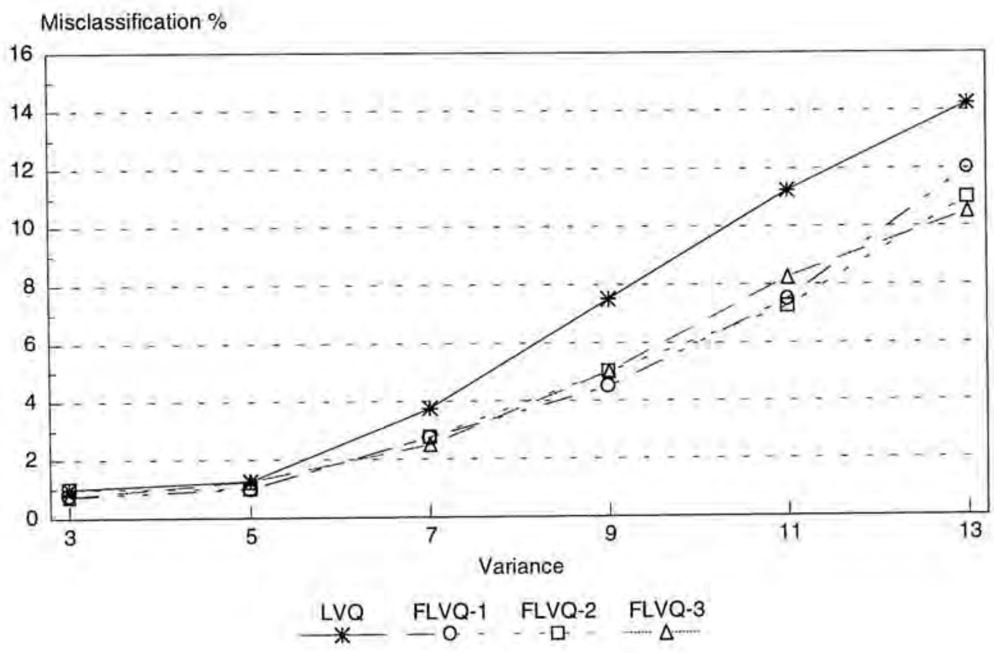
Figure 4.11 Escaping from Neuron Underutilization Case of Fig.4.9 by Unsupervised Fuzzy Competitive Learning. The solid lines are the learning trajectories of four competing neurons. After a few iterations of training, the "excess" neuron in the lower right cluster has been moved to the second left cluster and hence the four cluster centroids are learned. Note also the zigzagging trajectory of the left neuron which is the result of wandering between the two left clusters during the time the "excess" neuron has not been moved away from the lower right cluster.

***Experiment #2** : 2-D Gaussian Distributed Data Sets with Overlapping Clusters*

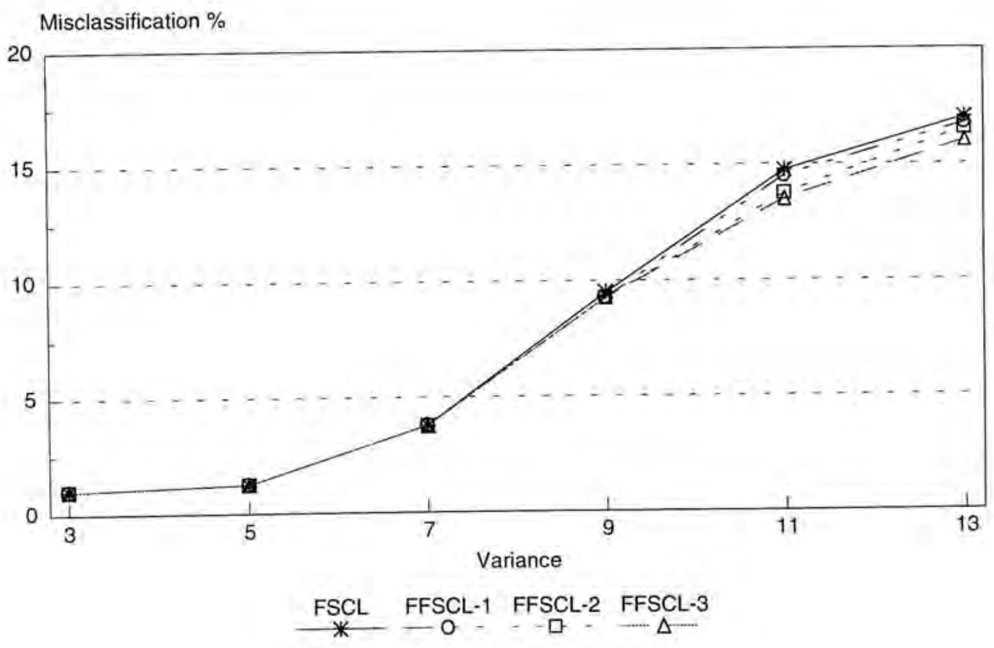
In contrast to previous data set, the 400-pattern gaussian distribution data sets here consist of two overlapping pattern clusters and they were generated from centroids (10,0) & (-10,0) with different variances. Specifically, we have fixed the variance of left centroid to 5 and varied that of the right centroid from 3 to 13 at 2 unit interval. Thus different degrees of overlapping simulating different fuzzy environments were resulted. In Fig.4.12, the algorithms' classification performances in terms of the averaged misclassification rate of 20 trials using $\alpha(0) = 0.1$ and networks of two competing neurons are depicted. The differences between fuzzy and non-fuzzy algorithms are not so significant in the unsupervised CL cases, i.e. the UCL and FSCL. However, fuzzy algorithms do perform better particularly in the fuzzier data sets. Furthermore, the fuzzier the training data sets, the more the fuzzy algorithms outperform the crisp one. For supervised CL, the performance gain is much larger, up to approximately 4% misclassification rate. From the results, it can also be observed that using larger *m* generally gives better results for the fuzzier data sets. It seems also that there exists a relationship between the degree of overlapping in data sets and the fuzziness value of the FCL algorithms.



(a)



(b)



(c)

Figure 4.12 Misclassification Rates of Crisp and Fuzzy Competitive Learning : 2-D Gaussian Data Sets with Different Degrees of Cluster Overlapping. See Fig.4.10 for descriptions of the depicted algorithms.

***Experiment #3** : Vowel Recognition Data Set*

The vowel data used in this simulation was collected by Deterding [30] who recorded examples of eleven steady state English vowels spoken by fifteen speakers of which eight were male and the other seven were female. Speech signals were first low-pass filtered at 4.7kHz and then A/D converted with 10kHz sampling rate and 12-bit word length. Twelfth order linear predictive analysis was carried out on six 512-sample Hamming windowed segments from the steady part of the vowel. The reflection coefficients were used to calculate 10 log area parameters, giving a 10 dimensional input space. In order to facilitate generalization performance testing, samples obtained from four male and three female speakers were set aside for testing while those from the other four male and four female speakers were used for training the network. This gave 8x11x6=528 training samples and 7x11x6=462 testing samples. The classification rates of both crisp and fuzzy CL algorithms on these two sets of vowel samples were summarized in Table 4.1 & 4.2 respectively with each reading recorded as the average of 20 runs. Initial learning rate $\alpha(0)=0.1$ and networks of different size were used in this experiment. The results show that such data set is not easy to classify. In fact, it has already been reported by Robinson [99] that the best generalization performance obtained from multilayer perceptron model is only 51%. Obviously, there exists a lot of overlappings between vowel classes and fuzzy algorithms are expected to perform better. According to Table 4.1, the training performances of fuzzy algorithms are indeed superior to those of the crisp ones for m=1.2 & 1.5. However, it is not the case for m=2.0, except the LVQ algorithm where divergencies were found. This indicates the importance of fuzziness value to the performance of proposed fuzzy algorithms. For the vowel data set here, fuzziness value around 1.2-1.5 should be used. However, this is not true for the data sets in experiment #2. It seems that an appropriate specification of fuzziness parameter is related to the characteristics of training data, e.g., the degree of overlapping.

Nevertheless, it is appropriate to choose values around 1.5 for general applications.

By comparing the performances of the three types of CL algorithms, one can conclude that supervised learning is indeed much more superior to unsupervised one and among the two unsupervised type CL algorithms, the win-rate dependent type CL algorithms (i.e. FSCL & FFSCL) are better. In addition, the classification rate of FFSCL is higher than those of the FSCL and UFCL, and this again justifies its development. Regarding the generalization performances of the proposed fuzzy algorithms, Table 4.2 demonstrates that they are generally in line with the training performances, that is, the generalization performances increase or decrease with the training performances. Such generalization performances however are better than those of other neural network models reported in [99]. In particular, the FLVQ algorithm (attaining 58.4%) significantly outperforms the multilayer perceptrons (51%) and the nearest-neighbor method (56%).

Table 4.1 Training Performances of Crisp and Fuzzy Competitive Learning (CL) algorithms: Vowel Data Set.

Neurons/Vowel	Classification Rate of Training Data Set (%)			
	UCL	UFCL-1	UFCL-2	UFCL-3
1	33.9	35.0	36.9	32.1
2	47.5	48.7	49.6	47.1
3	56.1	57.5	55.1	53.8
	LVQ	FLVQ-1	FLVQ-2	FLVQ-3
1	9.1†	70.3	64.4	61.7
2	9.1†	78.9	74.9	73.5
3	9.1†	86.0	81.1	78.9
	FSCL	FFSCL-1	FFSCL-2	FFSCL-3
1	35.4	37.0	36.3	31.0
2	49.6	51.4	50.0	47.0
3	56.8	59.5	56.5	53.4

† Diverged in all the 20 trials

Table 4.2 Generalization Performances of Crisp and Fuzzy CL Algorithms: Vowel Data Set.

Neurons/Vowel	Classification Rate of Training Data Set (%)			
	UCL	UFCL-1	UFCL-2	UFCL-3
1	36.1	36.0	37.9	31.5
2	48.9	49.4	49.2	48.8
3	55.8	55.8	55.9	54.5
	LVQ	FLVQ-1	FLVQ-2	FLVQ-3
1	9.1†	51.7	51.5	50.9
2	9.1†	57.0	56.8	57.2
3	9.1†	58.4	56.6	56.1
	FSCL	FFSCL-1	FFSCL-2	FFSCL-3
1	37.6	38.1	37.9	30.4
2	50.7	49.7	48.9	48.7
3	56.3	56.6	56.5	54.0

† Diverged in all the 20 trials

4.7.2 Performance of Monotonically Decreasing Fuzziness Control Scheme

Experiment #4 *: 2-D Gaussian Distributed Data Set with Distant Clusters*

This data set is the same as that in experiment #1 and was chosen to test on the performance of the fuzziness control scheme depicted in eq.(4.34), i.e.,

$$m(n) = m_f + (m_i - m_f)\exp(-\frac{5n}{n_{\max}}) \tag{4.38}$$

in helping the UFCL algorithm to avoid neuron underutilization. With the final fuzziness value $m_f = 1.2, 1.5, 2.0$, the initial fuzziness value m_i fixed at 2.5, and the maximum number of training iterations allowed equal to 30, the performances of this enhanced fuzzy algorithm, in terms of the neuron underutilization rate in 20 runs, are shown in Fig.4.13. As in Fig.4.10, the three final fuzziness values are symbolized by extensions "-1", "-2", "-3" respectively. Obviously, the UFCL algorithm with decreasing fuzziness are much better than the crisp algorithm. By comparing Fig.4.13 with Fig.4.10(a), the fuzziness control scheme substantially improves the ability of UFCL algorithm in avoiding neuron underutilization.

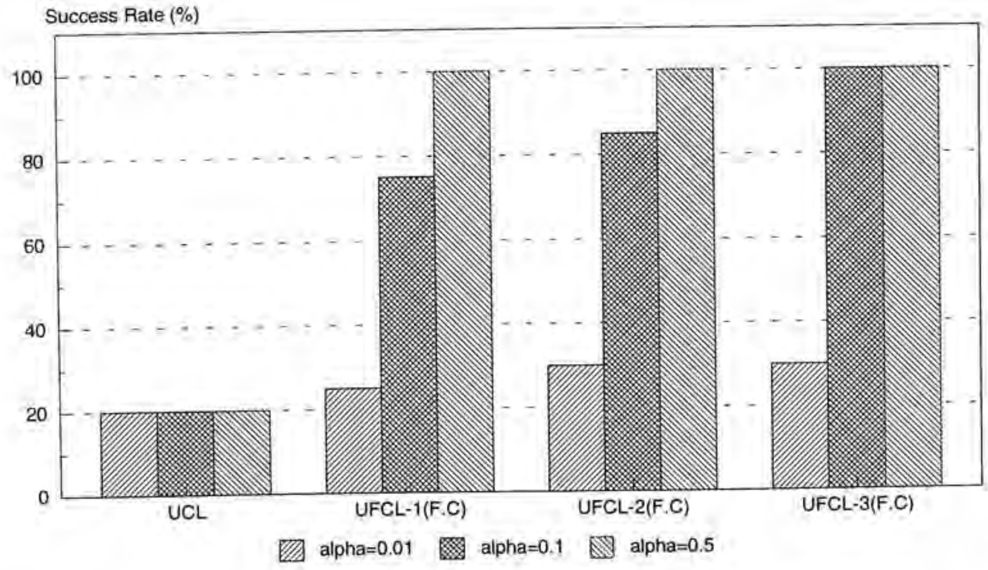


Figure 4.13 Success Rates of UCL and UFCL with fuzziness control in Converging to the Desired Solutions : 2-D Gaussian Data Set with Distant Cluster Locations.

Experiment #5 *: IRIS Data Set*

Anderson's IRIS data set [2] consisting of 50 labelled 4-D patterns for each of the three pattern classes has long been used for bench-marking the performances of clustering algorithms. Properties of the data are well-known [84] and typical error rates for supervised classifier are 0-3.3%; and for unsupervised classifier, around 10% [84]. In Table 4.3, the error rates of the UCL, UFCL, and UFCL with fuzziness control are depicted. Each reading was taken as the average of 20 runs using $\alpha(0) = 0.1$. In addition to using one neuron per pattern class, simulations using three neurons per pattern class were conducted. The results show that the fuzzy algorithms has outperformed the crisp one in all cases except one. By inspecting the effectiveness of the decreasing fuzziness scheme, one may observe that the scheme is effective only for the case of one neuron per pattern class. This is because neuron underutilization is no longer a problem when there are three neurons per pattern class.

Table 4.3 Error Rate of UCL, UFCL and UFCL with fuzziness control : IRIS Data Set.

Neurons/ Class	UCL	UFCL (fixed fuzziness)			UFCL (with fuzziness control)		
		m=1.2	m=1.5	m=2.0	m_f=1.2	m_f=1.5	m_f=2.0
1	16.0%	16.1%	14.7%	12.4%	13.0%	11.3%	10.7%
3	10.4%	8.2%	7.8%	8.3%	8.0%	8.5%	9.4%

4.7.3 Interpretation of Trained Networks

***Experiment #6** : 2-D Model Data Set*

In this data set, there are four clusters of Gaussian-distributed patterns. Each one consists of 100 patterns. The centroids are located at (2,0), (25,0), (15,3), (15,-3) respectively and the variances are 2.5. The first two clusters were labelled class A while the other two were labelled class B. As shown in Fig.4.14, the class A patterns '·' are separated by the class B patterns '+'. Consider that the two dimensions of this data set corresponds to the error and change-of-error states of a fuzzy system. For the class A state patterns, one particular control action takes place. For the class B state patterns, another control action is required. The FLVQ model can be used to estimate the membership functions of both control actions. In Fig.4.15, maximum of the two membership functions were plotted using the constrained fuzzy membership computations in eq.(4.35) with m=1.5 and the trained network's parametric vectors $\vec{m}_{11}(2.1,0.1)$, $\vec{m}_{12}(24.9,0)$, $\vec{m}_{21}(15.1,-3.2)$, and $\vec{m}_{22}(14.7,3.5)$. Note the membership scale is [0.5,1]. It can be seen that the middle "bump" corresponds to the class B patterns while the other two "bumps" correspond to class A's distant pattern clusters in Fig.4.14. As pointed out in Section 4.6, such a membership function is not appropriate for interpretations because it has been formulated as a function of the relative distance rather the absolute distance. Here, it can be observed that the memberships are too high for those far away points, say the four corners whose memberships are about 0.8. For comparisons, the possibilistic membership functions were plotted in Fig.4.16 according to eq.(4.36) where η_i were set to the average fuzzy intra-cluster distances [69]. The membership scale here is [0,1]. Comparatively, the possibilistic membership function is more appropriate for interpretations. By projecting the 2-D membership functions to the error and change-of-error spaces as shown in Fig.4.17 & Fig.4.18 respectively, the linguistic classification rules could be identified. According to the projected membership functions, the error (E) state could be attached with linguistic labels "small", "medium", and "large". On the other hand, the change-of-error (CE) state could be attached with linguistic labels "negative small", "zero", and "positive small". As a result, the trained FLVQ network is interpreted as follows.

Rule$^{(1)}$: If (E is *small* and CE is *zero*) or (E is *large* and CE is *zero*),
then control action A takes place

Rule$^{(2)}$: If (E is *medium* and CE is *negative small*) or (E is *medium* and CE is *positive small*),
then control action B takes place

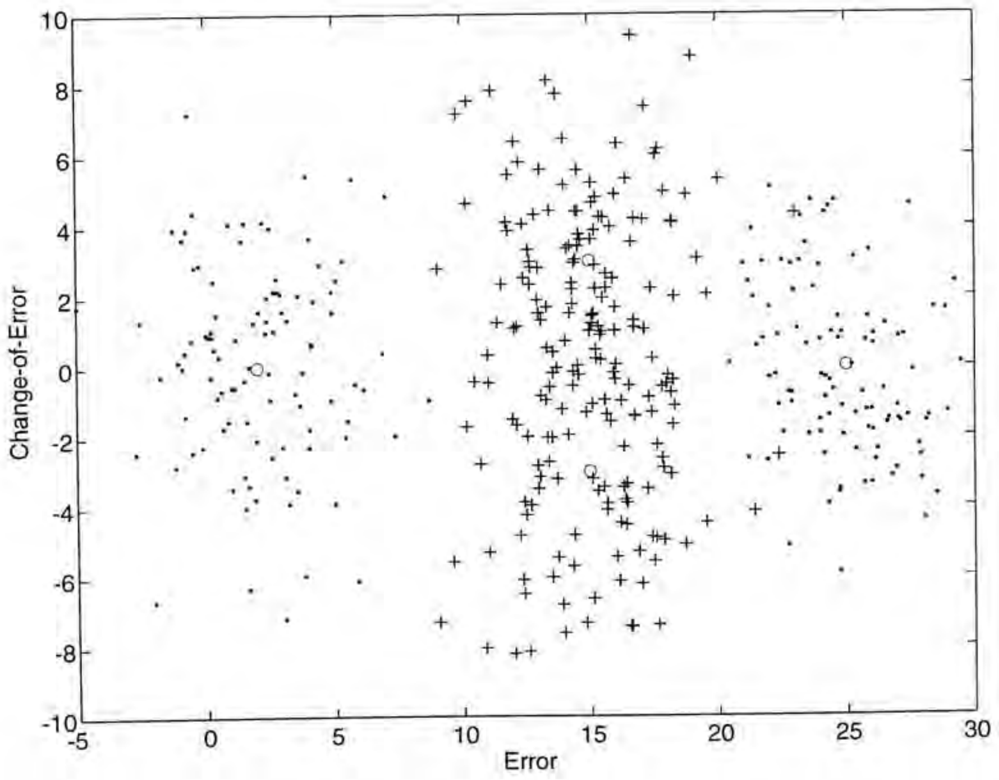


Fig. 4.14 A 2-D Gaussian-distributed data set with class A patterns '·' separated by class B patterns '+'. The Gaussian centroids are denoted by 'o'.

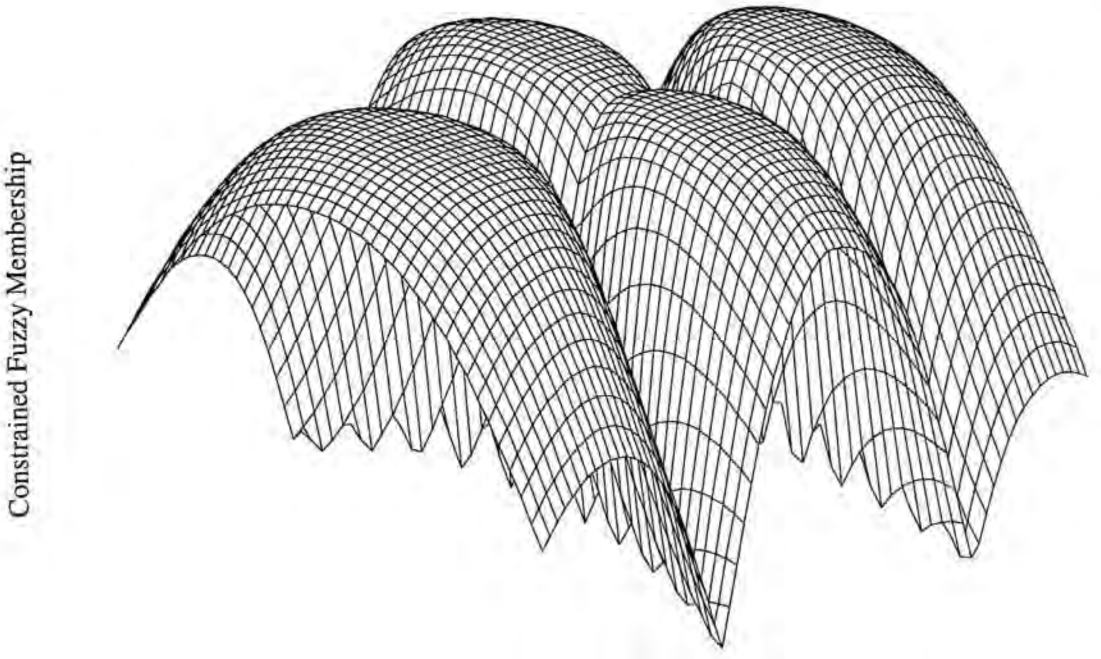


Fig. 4.15 Constrained Fuzzy Membership Functions Formed by FLVQ

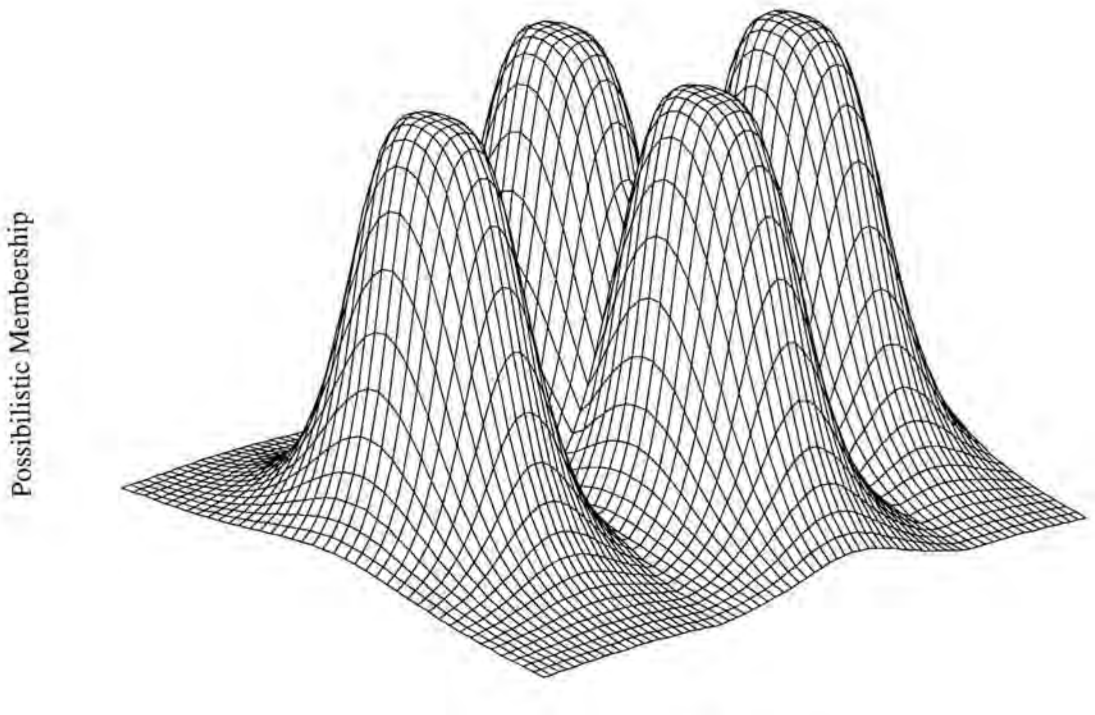


Fig. 4.16 Possibilistic Fuzzy Membership Functions Formed by FLVQ

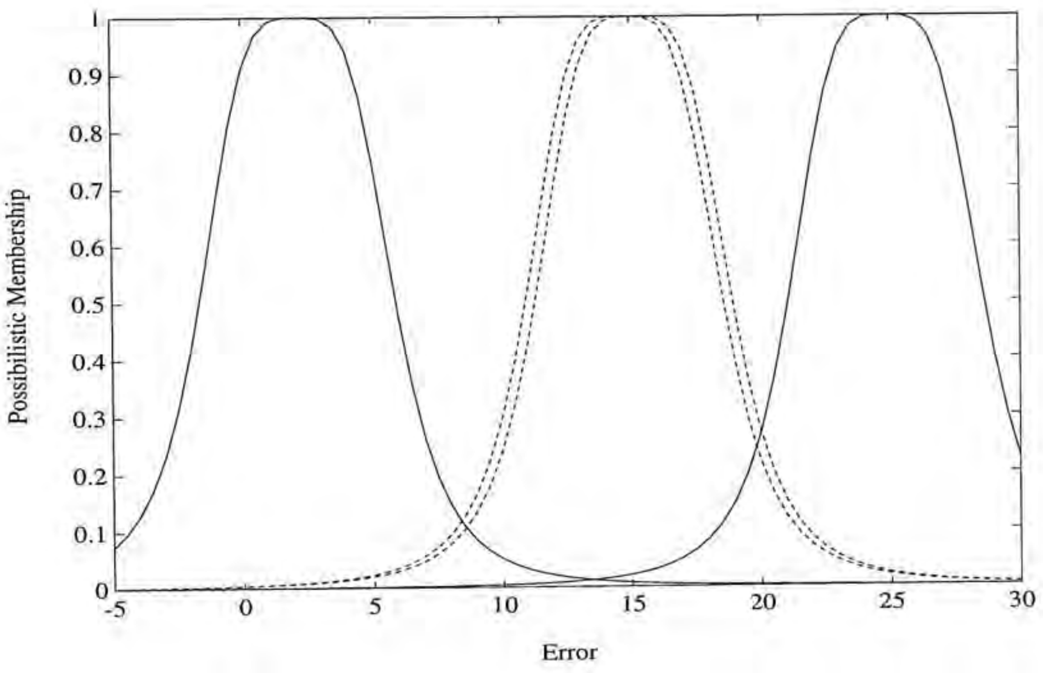


Fig.4.17 Projected Possibilistic Membership Functions : Error State. The class A membership functions are represented by solid lines and the class B membership functions are represented by dashed lines.

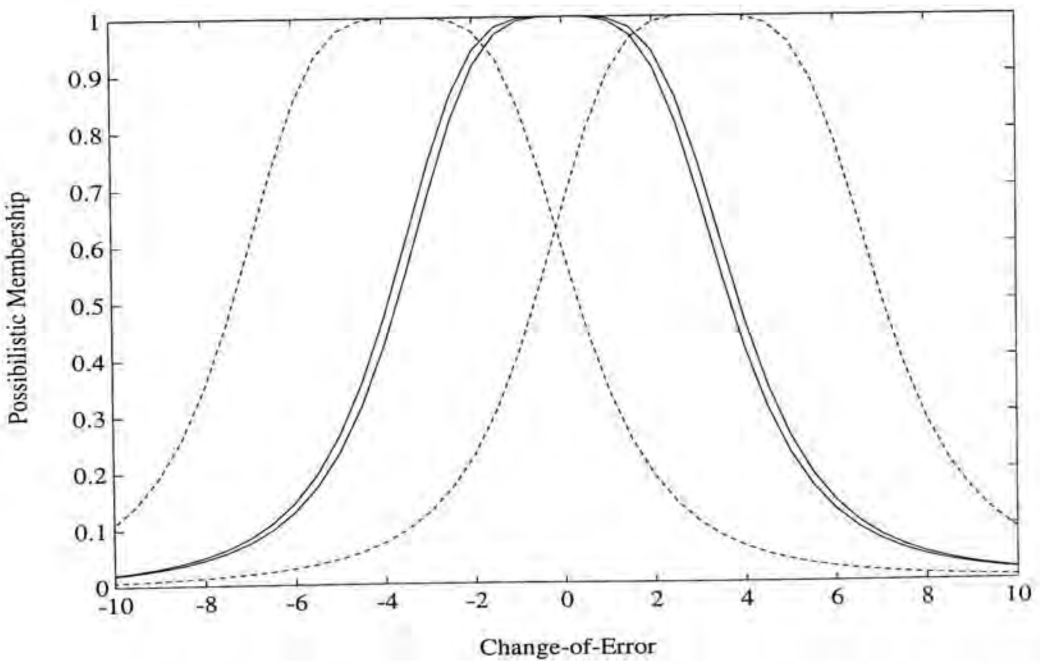


Fig.4.18 Projected Possibilistic Membership Functions : Change-of-Error State. The class A membership functions are represented by solid lines and the class B membership functions are represented by dashed lines.

4.8 Concluding Remarks

In this chapter, the concept of fuzzification of neural network learning has been demonstrated. Through a fuzzification of the crisp concept *win* in conventional competitive learning (CL) paradigm, a fuzzy competitive learning (FCL) paradigm adopting a "learn according to how well it wins" principle has been proposed and based upon which three existing CL algorithms have been fuzzified. Unlike the crisp competitive learning algorithms where only one neuron will win and learn at each competition, every neuron in the proposed FCL networks to a certain degree wins, depending on its distance to the input pattern, and learns the pattern accordingly. As demonstrated by the experimental results, the FCL networks are effective in addressing the two shortcomings of crisp CL models and superior to their crisp counterparts.

Some of the work developed in this chapter have already been reported in various technical publications and have received considerable attention. For example, the work reported in [16] has led Zhu *et al.* [122] to devise a new FCL algorithm called partial-distortion-weighted FCL for vector quantization. Its basic idea is to introduce the partial distortion theorem [42] of the theory of vector quantization into the UFCL algorithm. The theorem states that each partition region represented by one codevector makes an equal contribution to the distortion for an optimal quantizer with asymptotically large codebook size. In the wordings of competitive learning, each competing neuron should be equally responsible to minimize the distortion J_m of eq.(4.12) and this requires the competing neurons to be equally active in learning the input patterns. Such an idea is essentially the same as that of the FSCL and FFSCL algorithms presented in Sections 4.2.3 & 4.3.3 respectively and hence the partial-distortion-weighted FCL algorithm like FSCL and FFSCL is effective in overcoming the problem of neuron underutilization and designing near optimal vector quantizer regardless of the initial condition of the codevectors.

The class of fuzzy competitive learning networks developed in this chapter represents a new input to the first major category of FNS's, i.e., fuzzification of neural networks. It exploits the strength of fuzzy sets in modelling concepts that are fuzzy or vague in nature. One may synthesize another class of FNS's along this way by i) identifying the crisp concepts employed in the neural network model that are deficient in their current forms and are potentially better modelled by fuzzy set theory, e.g., the class identity of certain patterns, and ii) fuzzifying them using the developed theory and methodologies, e.g. [6,7], such that the performance of the models is improved. In the following two chapters, we switch our attentions to the second major category of FNS's and see how neural network and its theory help traditional fuzzy systems. In particular, we present a theoretical analysis of the storage capacity of FAM and FRNS models, both of which have been devised to encode a set of fuzzy rules, and develop high capacity encoding schemes for them via the well-developed perfect recall principle in associative memory.

Chapter 5

Capacity Analysis of Fuzzy Associative Memories

5.1 Introduction

Kosko's work on fuzzy associative memory (FAM) follows immediately from his work on bidirectional associative memory [66]. Its objectives are to capture the fuzzy rule knowledge of human experts in a particular problem domain, to represent it in modular and flexible neural network structure, and to infer from it in an efficient and effective manner. Despite its success in applying to problems such as backing up a truck-and-trailer [65], target tracking [67, Ch.11], and voice cell control in asynchronous transfer mode (ATM) networks [81], the FAM model still suffers from the problem of very low storage capacity — one rule pattern pair per FAM matrix. Subsequently, the implementation requires a large amount of hardware when the fuzzy rulebase is large and hence the FAM model is limited to small rulebase applications. Inspired from the neural network research work on associative memory, the multiple-rule storage property of FAM matrix is identified in this chapter [21,22]. A perfect recall theorem is established and based upon which the implementation of the FAM model can be more efficient. In addition, it is found that by generalizing the

FAM model to max-bounded-product (max-$\otimes$) composition, a single FAM matrix implementation of a set of semi-overlapped fuzzy rules, which is typical in general applications, can be achieved. The resulted model not only leads to higher efficiency in hardware implementation and computation, but also to the capability in bi-directional inference, i.e., forward and backward inference, which is of important value to knowledge-based systems [32].

In the next section, Kosko's FAM model is elaborated. In Section 5.3, the perfect recall theorem is presented and based upon which an efficient implementation of FAM is suggested. The max-$\otimes$ FAM model for single matrix implementation of a set of semi-overlapped fuzzy rules is introduced in Section 5.4. In Section 5.5, the capacity of a FAM matrix is formally discussed. Issues regarding the modification of certain stored fuzzy rules and the inference performance of the established high capacity FAM models are addressed in Section 5.6 and Section 5.7 respectively.

5.2 Fuzzy Associative Memories (FAMs)

Kosko's FAM model is characterized by a two-layer heteroassociative feedforward network as shown in Fig.5.1 which encodes the involved linguistic term pair (A_k, B_k) of the fuzzy rule

$Rule^{(k)}$: IF the input variable is A_k THEN the output variable is B_k

in a matrix form rather than a look-up table in traditional fuzzy systems. Specifically, it stores the *k*th fuzzy rule as represented by rule pattern pair (A_k, B_k) in its weight matrix M using correlation-minimum encoding

$$M = A_k^T \wedge B_k \tag{5.1}$$

or correlation-product encoding

$$M = A_k^T \cdot B_k \tag{5.2}$$

where the *k*th rule pattern pair is represented by fuzzy sets[1]

$$A_k = \{\mu_{A_k}(a) | a \in \mathbf{U}\} \tag{5.3}$$

and

$$B_k = \{\mu_{B_k}(b) | b \in \mathbf{V}\} \tag{5.4}$$

For the sake of clarity, we drop the membership symbols further, i.e.,

$$A_k = \{A_k(a) | a \in \mathbf{U}\} \tag{5.5}$$

and

$$B_k = \{B_k(b) | b \in \mathbf{V}\} \tag{5.6}$$

In order to avoid crosstalks, Kosko proposed to use separate storage of all the available fuzzy rules. It turns out that L available fuzzy rules are encoded by L FAM matrices using correlation-minimum (or correlation-product) encoding

$$\begin{aligned}
&Rule^{(1)}: M_1 = A_1^T \wedge B_1 \\
&Rule^{(2)}: M_2 = A_2^T \wedge B_2 \\
&\quad\vdots \qquad\quad \vdots \\
&Rule^{(L)}: M_L = A_L^T \wedge B_L
\end{aligned}$$

As shown in Fig.5.2, a FAM system comprises of a bank of L FAM matrices. Inference takes place individually for a fuzzy input A according to the compositional rule of inference (max-min composition) depicted in eq.(2.32), viz.

$$B_k' = A \circ M_k \tag{5.7}$$

The partial fuzzy outputs B_k' are then combined by an appropriate aggregation operator Σ, e.g., weighted sum or maximum, to form the overall fuzzy output. Here, we will focus on the maximum operator only. If crisp output is required, a defuzzifier can be cascaded to the FAM system to form the whole system.

[1] As compared with the formal definition in eq.(2.4), the notation of fuzzy sets here has been simplified to refer to the membership values only. This is applicable to the rest of the thesis.

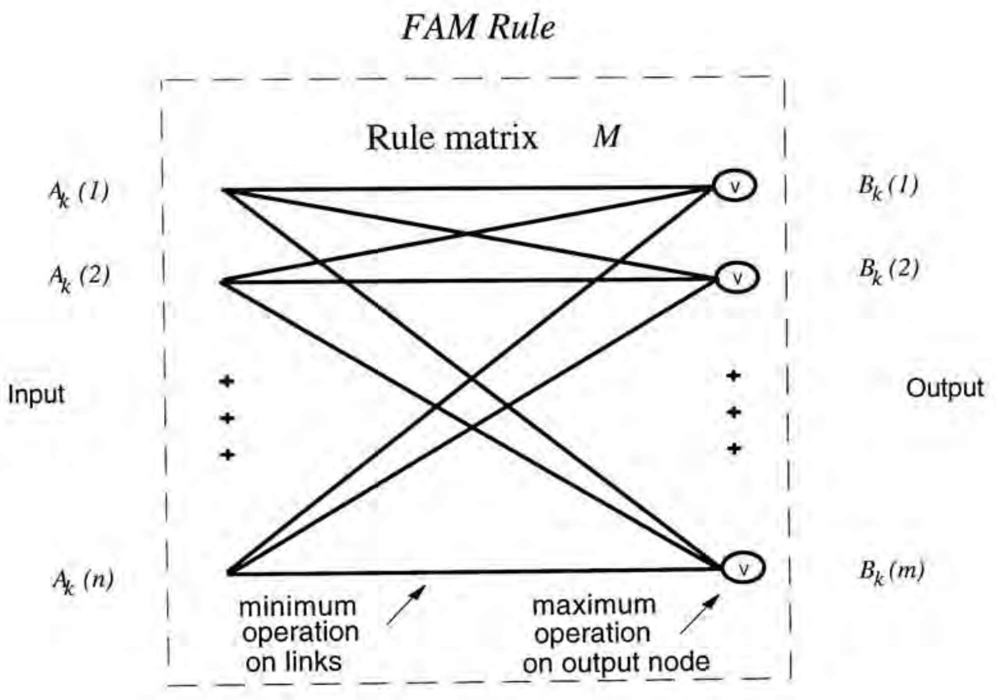


Figure 5.1 FAM Rule Network Architecture

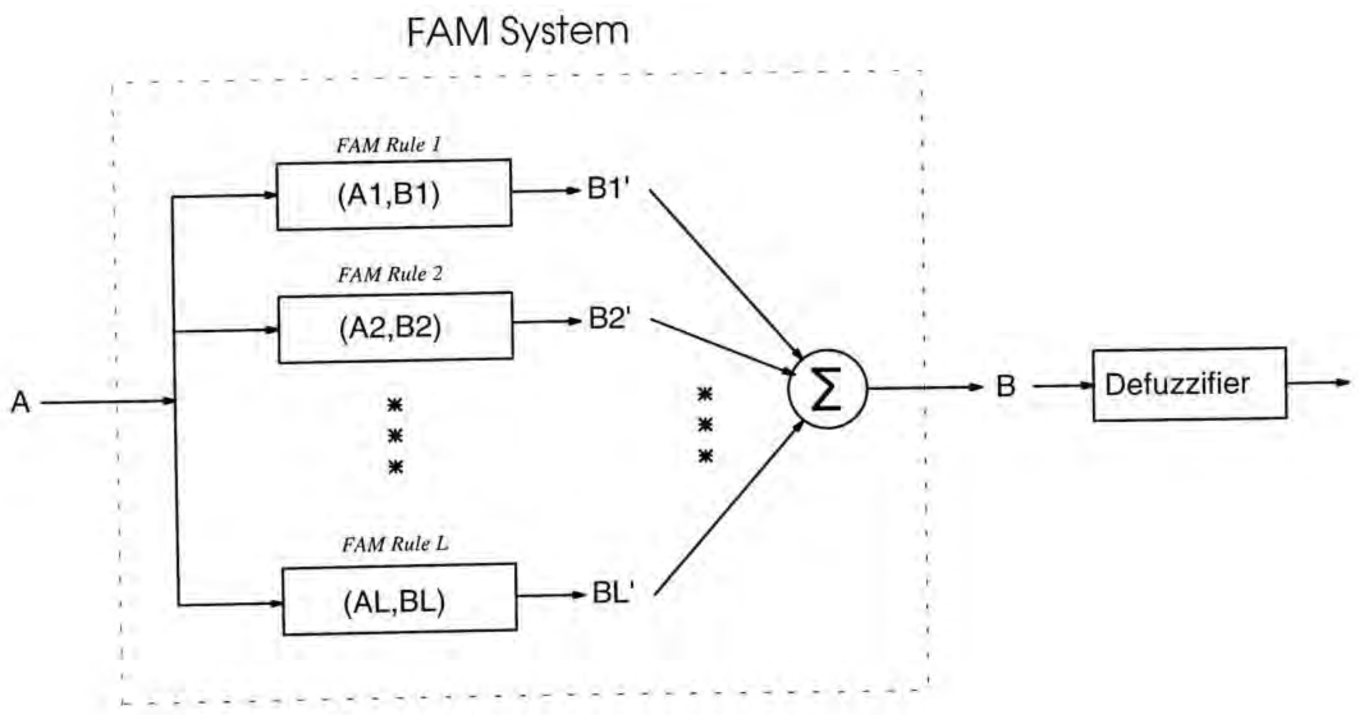


Figure 5.2 FAM System Architecture

For multiantecedent FAM rules, say "IF X is A_k AND Y is B_k THEN Z is C_k", Kosko proposed to split the rule into "IF X is A_k THEN Z is C_k" and "IF Y is B THEN Z is C_k" and hence the 3-D FAM matrix $(A_k, B_k; C_k)$ is decomposed into two 2-D FAM matrix (A_k, C_k^1) and (B_k, C_k^2). The actual fuzzy output is then recomposed by intersecting the partial results $C_k^1 \& C_k^2$. This corresponds to the "AND" connective of antecedent terms in the original rule. If they are related by "OR" connective, recomposition can take place by taking the union of the partial results. Such a *FAM decompositional inference scheme* [67, pp.322-327] requires far less storage than the multidimensional matrix approach at the expense of producing a close approximation of the fuzzy outputs rather than the actual ones. Without loss of generality, we will focus on single-antecedent FAM rules only in the rest of the chapter.

It can be seen that the FAM model implements the desired set-to-set mapping $S: I^n \to I^m$ as specified by the available fuzzy rules. It employs the individual-rule based inference mechanism but stores the fuzzy rule knowledge in fuzzy relation (FAM matrix) format in order to achieve parallel implementation and activation of fuzzy rules. As will be demonstrated in Section 5.7, the involved inference mechanism for crisp inputs is exactly the same as that depicted in Fig.2.6. Therefore, FAM is an efficient but cost-ineffective model as compared with traditional fuzzy systems. The separate storage scheme also has its advantages and disadvantages. On one hand, it leads to very low system capacity but on the other hand it features modularity which enables partial modifications, such as adding and deleting certain fuzzy rules, without disturbing the other storage.

5.3 Storing Multiple Rules in FAMs

In his popular textbook [67, pp.313-314], Kosko claims that information will lose if we superimpose multiple FAM rule matrices. Hence, the attempt to store multiple rules in a single FAM matrix was abandoned. In this section, we show that such a storage scheme is feasible for a certain distribution of input fuzzy sets and hence can lead to a more cost-effective implementation of the model.

In the theory of matrix associative memory, perfect recalls of multiple patterns are possible if the stored (crisp) patterns constitute a set of orthonormal basis vectors [86]. This result indeed can be generalized and applied to the FAM model.

<u>*Definition 5.1*</u> :

Fuzzy rule patterns $A_i = \{A_i(a) | a = 1,2,\dots,n\}$ and $A_j = \{A_j(a) | a = 1,2,\dots,n\}$ are max-min orthogonal to each other if

$$A_i \circ A_j = \max_a (A_i(a) \wedge A_j(a)) = 0 \tag{5.8}$$

i.e., $A_i(a) \wedge A_j(a) = 0 \quad \forall a$. Graphically, they are pairwisely disjoint.

<u>*Theorem 5.1*</u> :

Given a set of fuzzy rules represented by the rule pattern pairs $S=\{(A_k, B_k) | k = 1,2,\dots,L\}$ where $A_k = \{A_k(a) | a = 1,2,\dots,n\}$ and $B_k = \{B_k(b) | b = 1,2,\dots,m\}$ and it is encoded by a FAM weight matrix using max-min encoding as

$$M = \max_k [A_k^T \wedge B_k] \tag{5.9}$$

or in pointwise notation as

$$M(a,b) = \max_k [A_k(a) \wedge B_k(b)] \tag{5.10}$$

then the stored fuzzy rule pattern pairs can be perfectly recalled if the input fuzzy sets A_k are normal ones, i.e., $\max_a A_k(a) = 1$, and max-min orthogonal to each other.

Proof:

By performing max-min composition $A_l \circ M$ for an input fuzzy set $A_l \in S$, we have the element-wise outputs

$$\begin{aligned}\hat{B}_l(b) &= \max_a \left\{ A_l(a) \wedge [\max_k (A_k(a) \wedge B_k(b))] \right\} \\ &= \max_a \left\{ \max_k [A_l(a) \wedge A_k(a) \wedge B_k(b)] \right\} \\ &= \max_a \left\{ \max_{k \neq l} [A_l(a) \wedge A_k(a) \wedge B_k(b)] \vee [A_l(a) \wedge A_l(a) \wedge B_l(b)] \right\} \qquad (5.11)\end{aligned}$$

Since A_l & A_k are max-min orthogonal to each other, we have

$$\begin{aligned}\hat{B}_l(b) &= \max_a \left\{ \max_{k \neq l} [0 \wedge B_k(b)] \vee [A_l(a) \wedge A_l(a) \wedge B_l(b)] \right\} \\ &= \max_a \left\{ A_l(a) \wedge A_l(a) \wedge B_l(b) \right\} \qquad (5.12)\end{aligned}$$

As a result of the normal fuzzy set requirement, eq.(5.12) becomes $\hat{B}_l(b) = B_l(b)$, i.e., the stored output fuzzy set B_l is recalled. **Q.E.D.**

The theorem points out that there is no need to separate the storage of fuzzy rules if the input fuzzy sets are normal and max-min orthogonal to each other. This can reduce the hardware and computation requirements of the FAM model significantly by the proposed max-min encoding method in eq.(5.9). However, the orthogonal requirement is hard to satisfy in practice and semi-overlapped normal input fuzzy sets, as exemplified in Fig.5.3(a), are usually the case. By semi-overlapped, we define here as fuzzy sets satisfying the following two conditions

$$height(A_i \wedge A_j) = 0.5 \quad \forall \text{ adjacent fuzzy sets } A_i, A_j \qquad (5.13)$$

and

$$\sum_{k=1}^{L} A_k(a) = 1 \quad \forall a \qquad (5.14)$$

Regarding such distribution of input fuzzy sets, it can be inferred from the theorem that two FAM matrices are sufficient since the semi-overlapped input fuzzy sets can be decomposed into two sets of orthogonal fuzzy sets as shown in Fig.5.3. The resulted FAM system therefore comprises a bank of two FAM matrices only. This saves a lot of hardware and computations and we term such an implementation as reduced FAM model. The saving is proportional to the resolution of the fuzzy partition of the input universe of discourse. For the case in Fig.5.3, five FAM matrices, i.e., more than 70% of the resources, has been saved. On the basis of the FAM decompositional inference scheme, the result can be applied to multiantecedent fuzzy rules. Furthermore, it is also applicable to max-product FAM model. As will be demonstrated in Section 5.7, such an implementation of the FAM model, i.e., a system consisting of two FAM matrices constructed using max-min (or max-product) encoding, is functionally the same as Kosko's implementation.

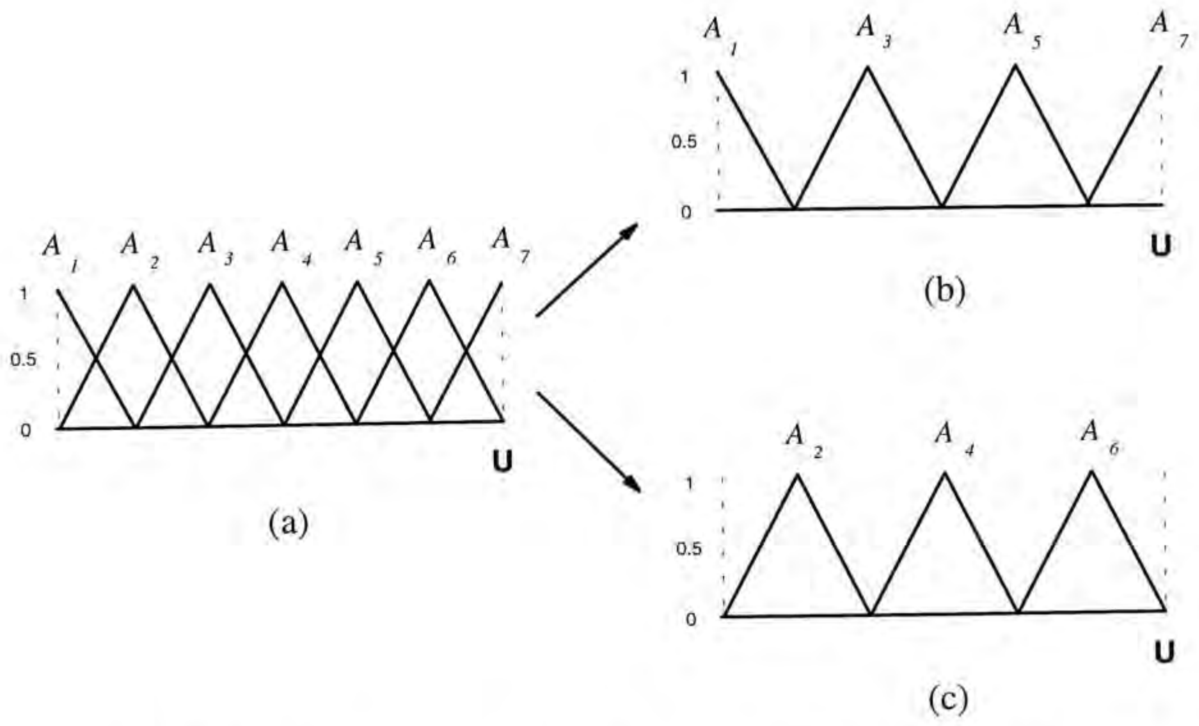


Figure 5.3 The decomposition of semi-overlapped fuzzy sets in (a) into two sets of orthogonal fuzzy sets in (b) & (c).

5.4 A High Capacity Encoding Scheme for FAMs

In the previous section, the derived perfect recall theorem has been applied to optimize the storage of fuzzy rules in Kosko's max-min composition FAM model and significant reductions in hardware and computations have been obtained. In fact, the perfect recall theorem derived can be generalized to the max-*t* composition [94], a more general class of inference operators.

Definition 5.2 :

Fuzzy sets A_i and A_j are max-*t* orthogonal to each other if

$$A_i \circ_t A_j = \max_a (A_i(a)\, t\, A_j(a)) = 0 \tag{5.15}$$

i.e., $A_i(a)\, t\, A_j(a) = 0 \quad \forall a$.

Theorem 5.2 :

Given a set of fuzzy rules $S = \{(A_k, B_k) | k = 1,2,...,L\}$ and it is encoded by a FAM weight matrix using max-*t* encoding

$$M = \max_k [A_k^T\, t\, B_k] \tag{5.16}$$

or in pointwise notation as

$$M(a,b) = \max_k [A_k(a)\, t\, B_k(b)] \tag{5.17}$$

then the stored fuzzy rule pattern pairs (A_k, B_k) can be perfectly recalled if the input fuzzy sets A_k are normal ones and max-*t* orthogonal to each other.

Proof :

Owing to the distributive property of maximum operator with respect to *t*-norm, the proof is the same as that of Theorem 5.1 and therefore is omitted here.

According to the generalized perfect recall theorem, single FAM matrix implementation of semi-overlapped fuzzy rules is possible if there exists a composition operator that makes semi-overlapped fuzzy rules orthogonal to each other. In this regard, the max-bounded-product (max-$\otimes$) composition, due to its computational simplicity and well-behaved inference property (to be discussed in Section 5.7), is proposed. Recall from Chapter 2 that the bounded product *t*-norm is defined as

$$p \otimes q = \max\{0, p+q-1\} \tag{5.18}$$

According to the condition for semi-overlapped fuzzy rules in eqs.(5.13)&(5.14), the max-⊗ composition

$$\begin{aligned} A_i \circ_{\otimes} A_j &= \max_a (A_i(a) \otimes A_j(a)) \\ &= \max_a \{\max[0, A_i(a) + A_j(a) - 1]\} \\ &= 0 \end{aligned} \tag{5.19}$$

i.e., the orthogonal requirement is satisfied. Hence, the semi-overlapped fuzzy rules can be encoded by a FAM matrix using max-⊗ encoding. Due to its matrix structure, such an implementation will be beneficial to accommodate backward inference [32] on top of the usual forward inference in the FAM model. Backward inference does not play a role in fuzzy control, though it is an essential strategy for guiding the querying process in knowledge-based systems [32]. Thus, the max-⊗ FAM model will have more applications.

5.5 Memory Capacity

Based upon the established results, the capacity of a FAM matrix in the two high capacity FAM models, i.e., the reduced model (using max-min composition) and the high capacity model (using max-⊗ composition), is derived in this section. According to Theorem 5.1, one FAM matrix is enough for storing a set of fuzzy rules that are normal and max-min orthogonal to each other. If the max-min orthogonal condition cannot be satisfied, eq.(5.12) of the proof will become

$$\hat{B}_l(b) \geq B_l(b) \tag{5.20}$$

i.e., the perfect recall property is not guaranteed. Hence, the capacity of a FAM under max-min composition is bounded by such conditions. Once they are met, the memory capacity depends on the resolution of the input fuzzy sets involved. For example in Fig.5.4(a), the maximum number of fuzzy rules that can be stored by a FAM is 4. On the other hand in Fig.5.4(b), a FAM matrix can store up to 7 fuzzy rules for the same discrete universe of discourse.

Similarly, the capacity of a FAM matrix under max-⊗ composition is bounded by the semi-overlapped condition, according to Theorem 5.2 and eq.(5.19). Again it depends on the resolution of the input fuzzy sets involved. As one may expect, the capacity under max-⊗ composition is about twice that under max-min composition because the compactness of semi-overlapped condition doubles that of the non-overlapped condition. With the same discrete universe of discourse and resolution in Fig.5.4(a)&(b), the capacities under max-⊗ composition will be 7 and 13 fuzzy rules respectively.

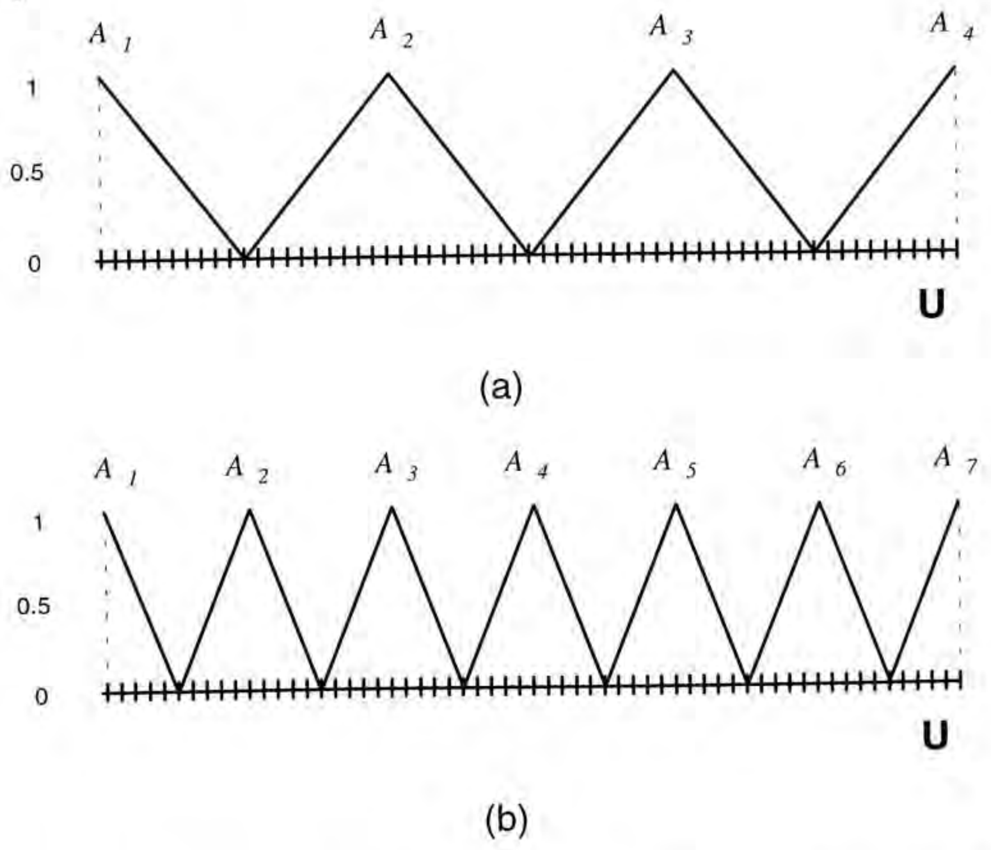


Figure 5.4 (a) Memory Capacity of 4 Rules; (b) Memory Capacity of 7 Rules

5.6 Rule Modification

Recall from Section 5.2 that Kosko's separate storage scheme is advantageous in enabling rule modifications (addition, deletion, and updating) without disturbing the other storage. For example, deleting a rule simply requires disconnecting the corresponding FAM matrix from the system. Conversely, rules can be added by connecting FAM matrices to the system. Rule modifications are essential to the success of FAM because the model as an adaptive fuzzy system is frequently required to follow the environmental changes. Since each FAM matrix now stores more than one rule, rule modifications have to be carried out in another way and some effective methods to do so are developed in this section. Consider the following set of semi-overlapped fuzzy rules.

R1 : IF X is *very small* THEN Y is *medium*

R2 : IF X is *small* THEN Y is *small*

R3 : IF X is *medium* THEN Y is *very small*

R4 : IF X is *large* THEN Y is *large*

R5 : IF X is *very large* THEN Y is *very large*

where the fuzzy terms *very small* (VS), *small* (S), *medium* (M), *large* (L), and *very large* (VL) are defined in Fig.5.5. By using the reduced FAM model, two FAM matrices corresponding to the encoding of the rule sets {R1, R3, R5} and {R2, R4} will be formed and their 3-D surface plots are shown in Figs.5.6(a)&(b) respectively. Each pyramid represents a rule and its projections onto the X and Y variable spaces reveal the associated triangular shape fuzzy terms. The contour plots of these two matrices are depicted in Figs.5.7(a)&(b). As a result of the max-min orthogonal condition, Figs.5.6 & 5.7 show that the pyramids are isolated from each other. Based on such property, rules can be easily deleted by suppressing the corresponding pyramids to zero memberships. This can be achieved by a simple weight updating operation

$$M_{new}(a,b) = \min[M(a,b), \Gamma(a,b)] \tag{5.21}$$

where

$$\Gamma(a,b) = \begin{cases} 0 & (a,b) \in support(M_k = A_k^T \wedge B_k) \\ 1 & otherwise \end{cases} \tag{5.22}$$

via which the *k*th rule can be deleted. On the other hand, rule addition is straight-forward because it is exactly the same as the rule encoding operation. If the *k*th rule has to be added, the corresponding FAM matrix is updated, according to the max-min encoding rule in eq.(5.10), as

$$M_{new}(a,b) = \max[M(a,b), \Gamma(a,b)] \tag{5.23}$$

where

$$\Gamma(a,b) = A_k(a) \wedge B_k(b) \tag{5.24}$$

Thus, the pyramid for the *k*th rule will be formed accordingly. In order to preserve the perfect recall property, the rules being added should conform to the normal fuzzy sets and max-min orthogonal requirements. With the rule deletion and addition schemes described by eqs.(5.21)-(5.24), rule updating can be easily implemented by a delete-and-add process, i.e., the obsolete rule is firstly deleted using eqs.(5.21) & (5.22) and the updated one is then added using eqs.(5.23) & (5.24).

Similar rule modification schemes can be derived for the max-⊗ FAM model. As shown in the 3-D surface plot of Fig.5.8 and the contour plot of Fig.5.9, the single FAM matrix formed consists of five pyramids. Since the pyramids are also isolated from each other, the rule deletion and addition schemes established previously can be applied to the max-⊗ model as

$$M_{new}(a,b) = \min[M(a,b), \Gamma(a,b)] \tag{5.25}$$

where

$$\Gamma(a,b) = \begin{cases} 0 & (a,b) \in support(M_k = A_k^T \otimes B_k) \\ 1 & otherwise \end{cases} \tag{5.26}$$

and

$$M_{new}(a,b) = \max[M(a,b),\Gamma(a,b)] \tag{5.27}$$

where

$$\Gamma(a,b) = A_k(a) \otimes B_k(b) \tag{5.28}$$

respectively. Again, rule updating can be carried out by a delete-and-add process using eqs.(5.25)-(5.28).

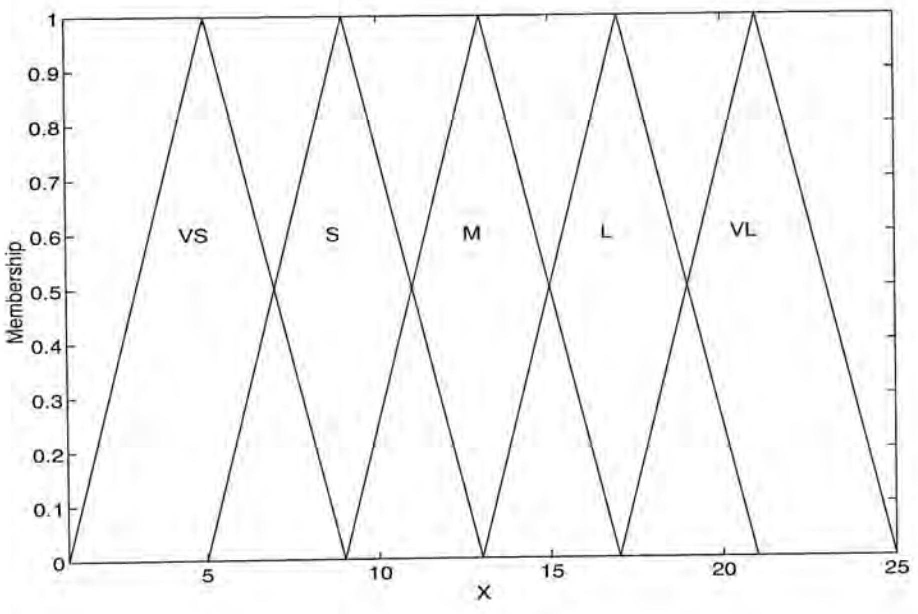
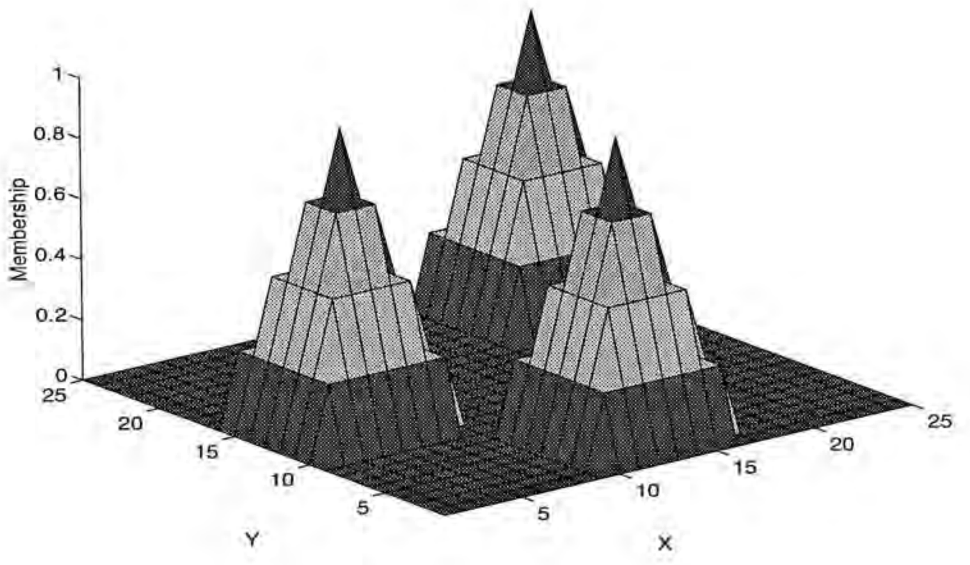
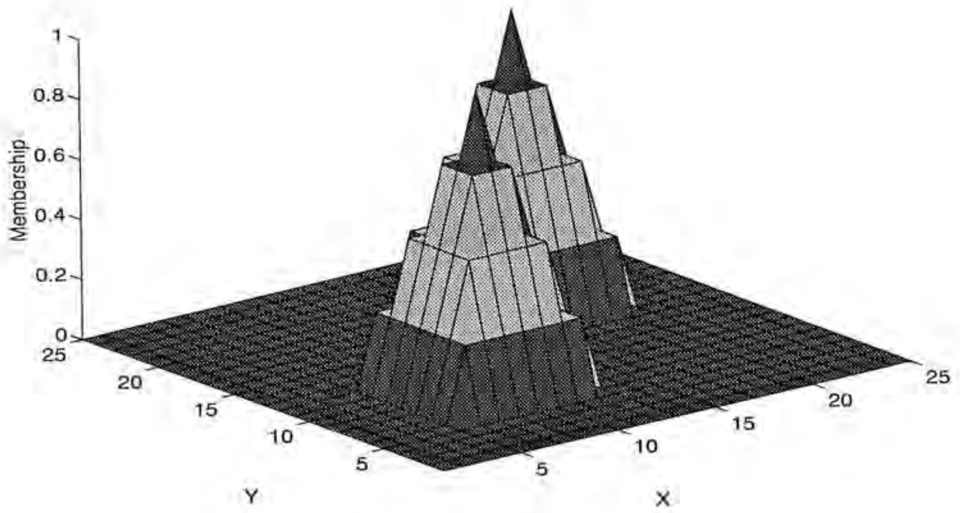


Figure 5.5 The Membership Functions of Input Fuzzy Terms *very small* (VS), *small* (S), *medium* (M), *large* (L), and *very large* (VL)

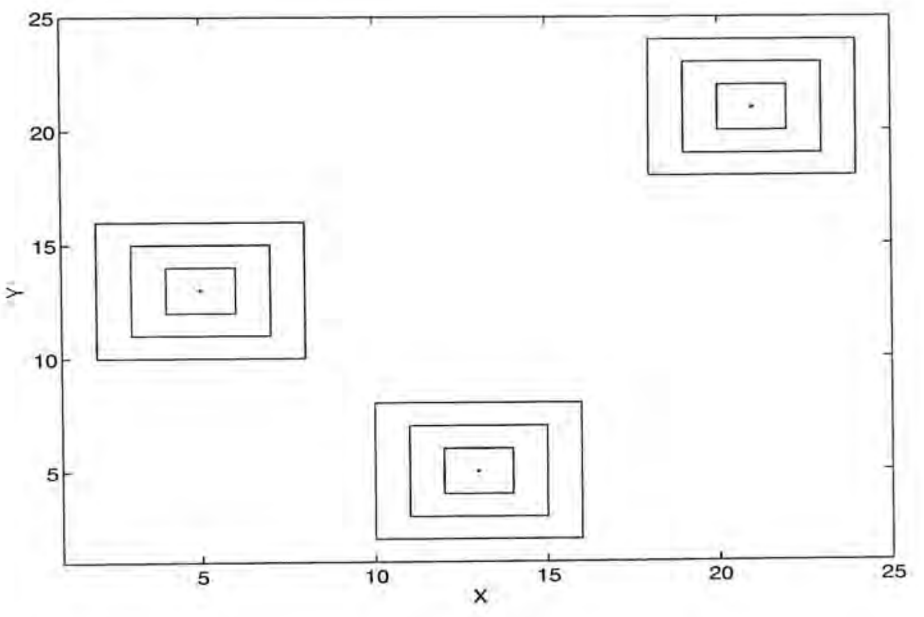


(a)

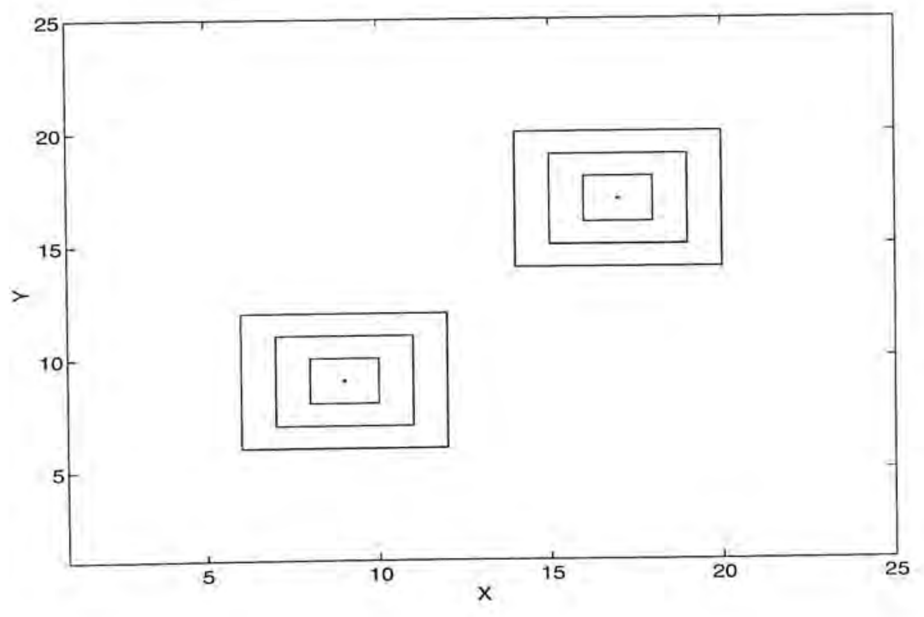


(b)

Figure 5.6 3-D Surface Plots of the Two FAM Matrices in Reduced Model : (a) for the set of rules {R1, R3, R5}, (b) for the set of rules {R2, R4}



(a)



(b)

Figure 5.7 Contour Plots of the Two FAM Matrices in Reduced Model : (a) for the set of rules {R1, R3, R5}, (b) for the set of rules {R2, R4}

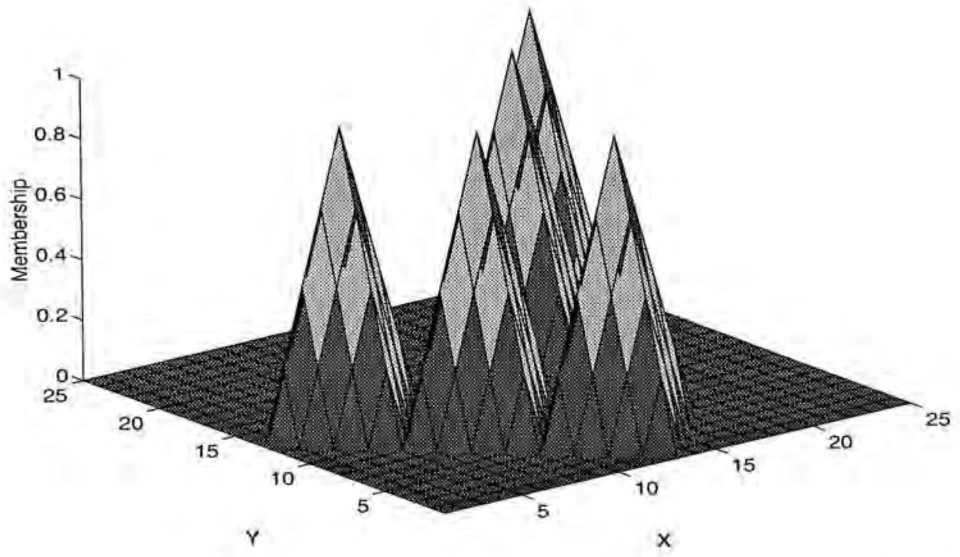


Figure 5.8 3-D Surface Plot of the FAM Matrix in max-⊗ FAM Model

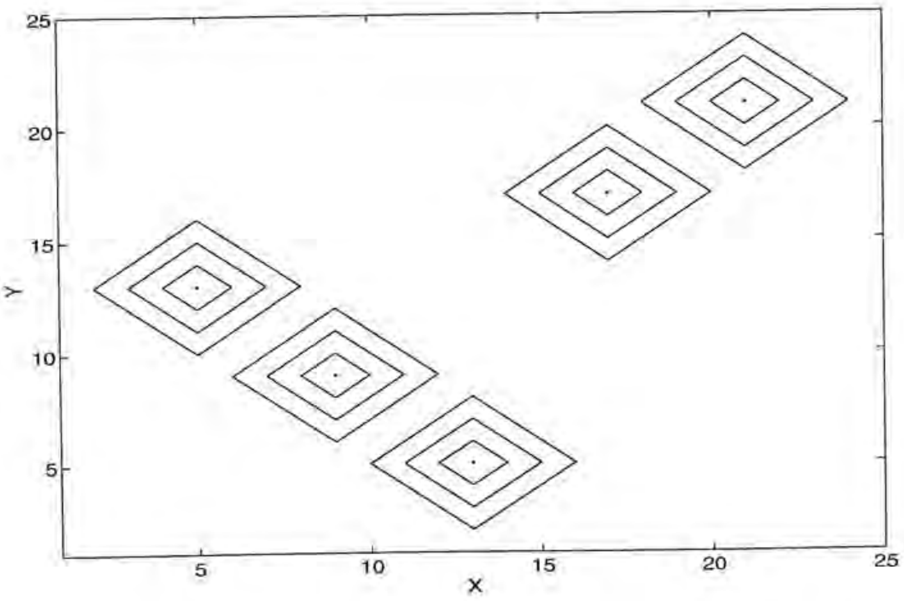


Figure 5.9 Contour Plot of the FAM Matrix in max-⊗ FAM Model

5.7 Inference Performance

In this section, the inference performances of Kosko's model, the reduced model and the max-$\otimes$ models are demonstrated and compared through the semi-overlapped fuzzy rule set described in Section 5.6. It is recalled here as consisting of the fuzzy rule pattern pairs {R1:(*very small, medium*), R2:(*small, small*), R3:(*medium, very small*), R4:(*large, large*), R5:(*very large, very large*)} with the membership functions of the five fuzzy terms defined in Fig.5.5. Three types of fuzzy inputs were used for testing. They includes fuzzy terms from the rule set, modified fuzzy terms, and singleton fuzzy terms.

The first type of fuzzy inputs is used to test on the model's ability in recalling the stored fuzzy output. We present the input "X is *medium*" and see how well the models produce the desired output "Y is *very small*", i.e., realizing R3. In Fig.5.10, the inference output (solid line) of Kosko's model is shown. It is the same as that of the reduced model and their output is the maximum of fully fired *very small* (R3), partially fired *small* (R2), and partially fired *large* (R4). Unfortunately, both of them cannot perfectly recall the desired output fuzzy term *very small.* As shown in Fig.5.11, the max-$\otimes$ FAM model can achieve this. Such a difference is particularly illustrative to this example because the defuzzified output value, using COG defuzzification (see eq.(2.39)), of Kosko's (or reduced model) will be approximately[2] equal to 10 which is around the nucleus of the fuzzy term *small* rather than the expected *very small.*

The second type of fuzzy inputs is modified fuzzy terms [123]. In Fig.5.12, two such terms, i.e., *more-or-less medium* and *very medium*, are shown. They were presented to the three models and the inference performances are reported as follows. The inference outputs (solid lines) of Kosko's model for them are depicted in Figs.5.13 & 5.14 respectively. Again, the outputs of the reduced model are the same

[2] The exact quantity depends on the resolution of the universe of discourse adopted.

as those of Kosko's model. Fig.5.13 shows that in response to "X is *more-or-less medium*", all the rules have been fired with strength 0.2 for R1 & R5, 0.6 for R2 & R4, and 1 for R3. These values correspond to the memberships of the intersecting points of *more-or-less medium* with respect to VS, S, M, L, & VL respectively. For the modified fuzzy input "X is *very medium*", only R2, R3, & R4 have been fired and the strengths are 0.4, 1, & 0.4 respectively. These values can also be observed from Fig.5.14. By comparing the inference outputs for inputs "X is *more-or-less medium*" and "X is *very medium*", the former is fuzzier than the later and its defuzzified output using COG method is ≈11.2 which is more far away from *very small* than the latter one (≈9.8). This matches with the intuition that *more-or-less medium* is a fuzzier input term than *very medium* and therefore should produce more general output.

For the max-⊗ model, the inference output for input "X is *very medium*" is the same as that for input "X is *medium*" already shown in Fig.5.11. This is a common intuitive criterion when the causal relation between input and output of fuzzy rules is not strong [72]. The inference output for input "X is *more-or-less medium*" is depicted in Fig.5.15. Unlike Kosko's or the reduced model, the partially fired outputs are of scaled triangular shape and their peak values, as indicated in Fig.5.15, are the membership value of *more-or-less medium* at the nucleuses of *small* and *large*. The defuzzified output value using COG method in this case is ≈6.4, as compared with ≈ 11.2 of Kosko's or reduced model. Judging from the defuzzified output values, the inference performance of max-⊗ model for modified fuzzy inputs is intuitively better than that of Kosko's or reduced model.

The third type of fuzzy input is singleton fuzzy terms and is used to study the performance for crisp inputs. In Fig.5.16, the inference output of Kosko's or reduced model for a crisp value x=15 is recorded. It shows that rules R3 & R4 have been fired with strength 0.5. This result is indeed exactly the same as that using the individual-rule based inference described in Fig.2.6. The inference output of the max-⊗ model is shown in Fig.5.17. Again, the partially fired fuzzy outputs are of scaled triangular shape. Despite the difference in shape, the defuzzified outputs of Kosko's (or reduced) model and max-⊗ model using COG method are the same in this case. For the other crisp input values, they may not be exactly the same but at least very close to each other. Therefore, it can be concluded that the inference performances of Kosko's (or reduced) model and max-⊗ model for crisp inputs are comparable to each other.

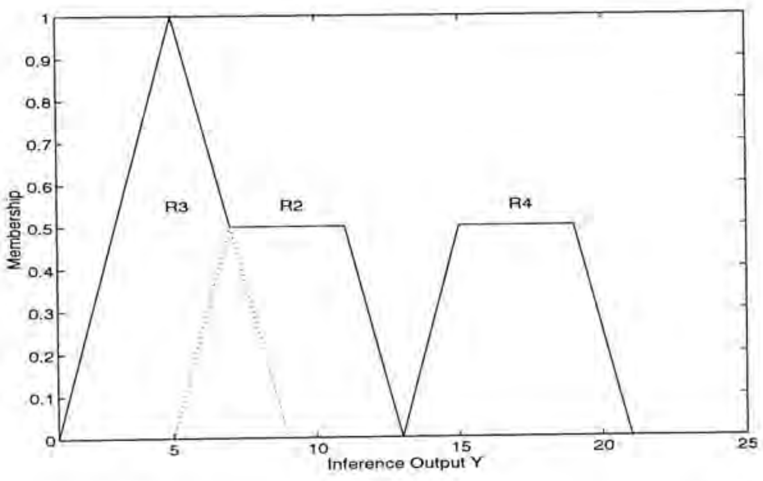


Figure 5.10 The Inference Output of "X is *medium*" : Kosko's and Reduced FAM Models

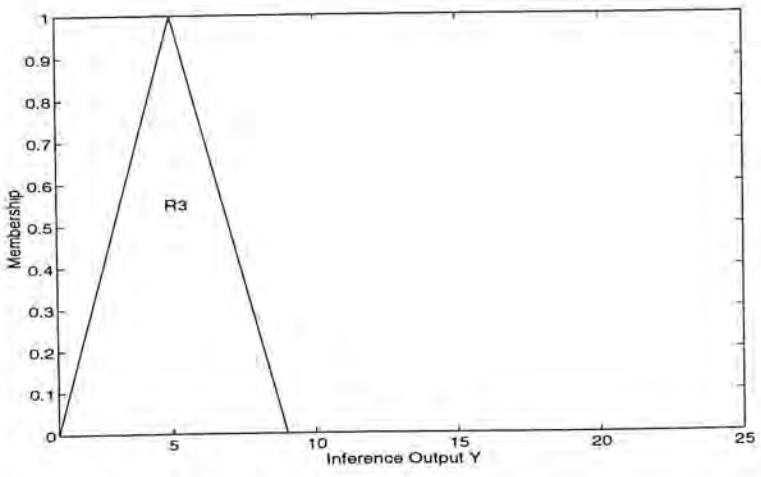


Figure 5.11 The Inference Output of "X is *medium*" : max-⊗ FAM Model

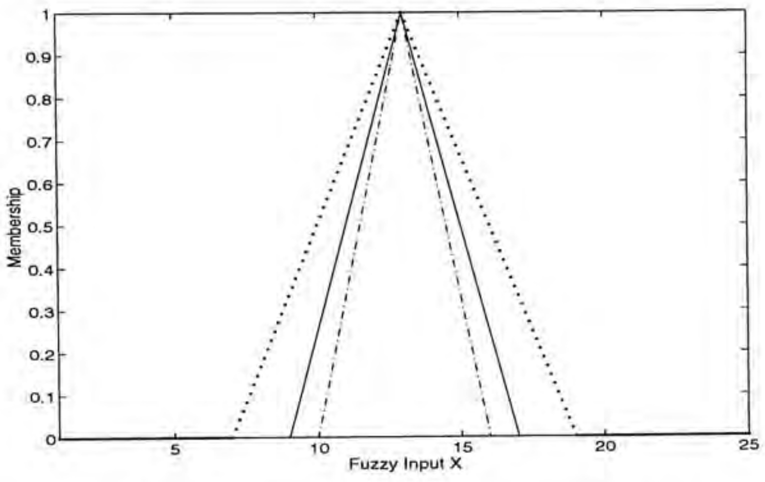


Figure 5.12 Modified Fuzzy Terms : *more-or-less medium* ⋯ ⋯ and *very medium* —·—·—

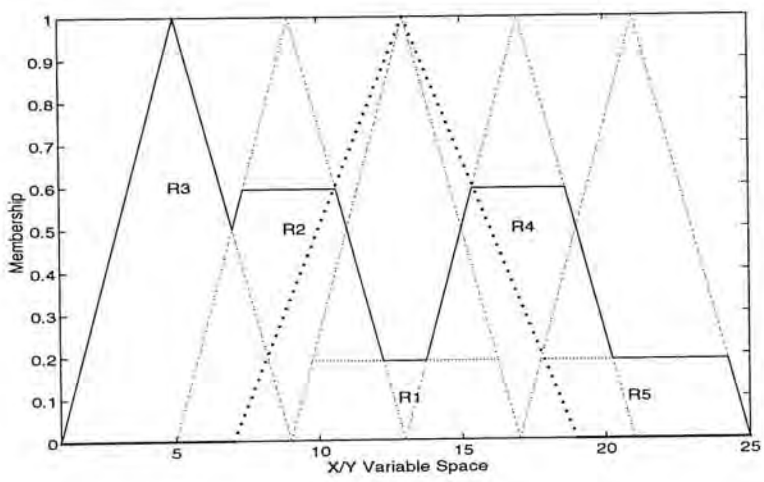


Figure 5.13 The Inference Output of "X is *more-or-less medium*" : Kosko's and Reduced FAM Models

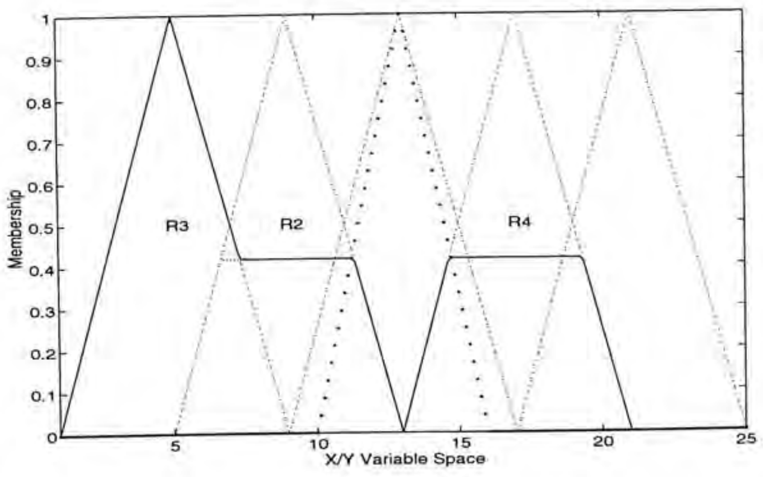


Figure 5.14 The Inference Output of "X is *very medium*" : Kosko's and Reduced FAM Models

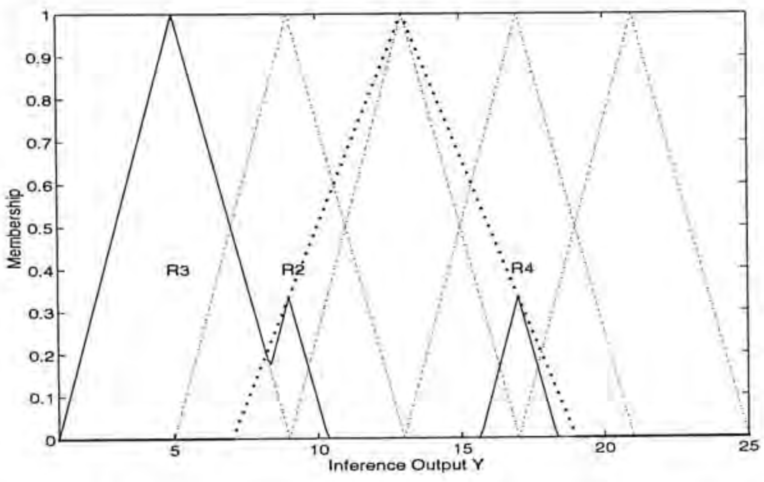


Figure 5.15 The Inference Output of "X is *more-or-less medium*" : max-⊗ FAM Model

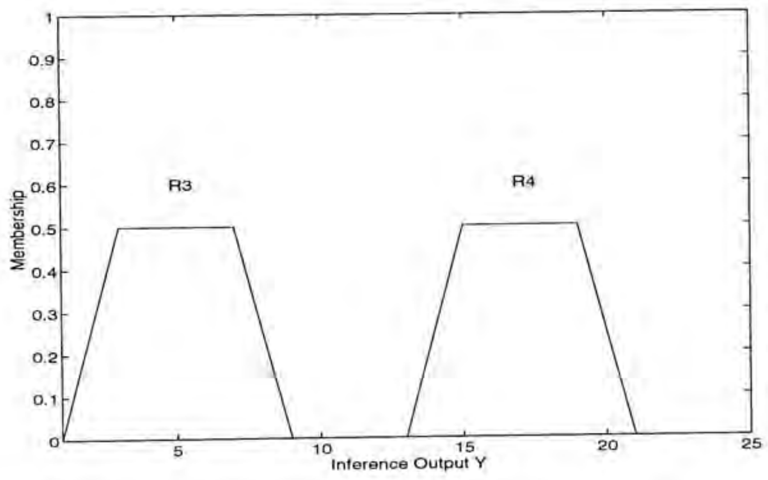


Figure 5.16 The Inference Output of Crisp Input x=15 : Kosko's and Reduced FAM Models

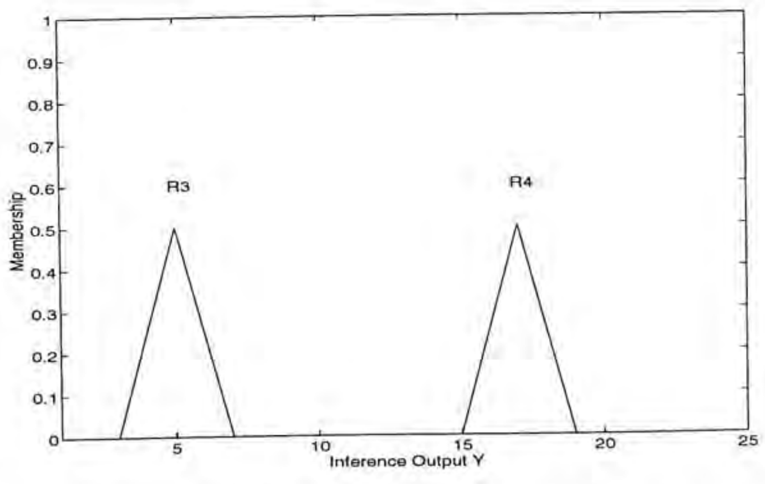


Figure 5.17 The Inference Output of Crisp Input x=15 : max-⊗ FAM Model

5.8 Concluding Remarks

In the light of the research work on neural associative memory, we have proved that the capacity of a FAM matrix is not limited to a single rule. One can store multiple fuzzy rules in a FAM matrix using max-min encoding if the normal and max-min orthogonal conditions on the involved input fuzzy sets are observed. This implies that a FAM system consisting of two FAM matrices only is enough for storing a set of semi-overlapped fuzzy rules, which is typical in general applications. Such an implementation is called reduced FAM model because it reduces the storage requirement of FAM. By generalizing the usual max-min composition to the max-$\otimes$ one, one can even encode such a set of rules by one FAM matrix only. The resulted model not only is cost effective in implementation, it also features bi-directional inference which is essential to applications in knowledge-based systems.

With the introduction of the reduced and max-$\otimes$ models, mechanisms for partial modification of the fuzzy rulebase have been revised. Based upon an observation of the structures of the max-min encoded and max-$\otimes$ encoded FAM matrices, simple weight modification schemes for adding, deleting, updating rules have been devised and they are as effective as the original ones. In addition, the inference performances of the two high capacity models have been illustrated and compared with that of Kosko's max-min model through a typical rulebase. The reduced model has been shown to be functionally the same as Kosko's model while the inference output of the max-$\otimes$ model is generally the most appealing one.

As mentioned in the introductory section, Kosko's FAM is a neural network inspired model for encoding a set of fuzzy rules. In the next chapter, we will look at a mathematical oriented model, i.e., the fuzzy relational equations, and present a rigorous analysis of the storage capacity of the FRNS model.

Chapter 6

Capacity Analysis of Fuzzy Relational Neural Systems

6.1 Introduction

Since its introduction by Sanchez [103] in 1976, fuzzy relational equations have been continuously exploited by researchers in both theories [32,51,90] and applications [32,97]. As mentioned in Chapter 3, fuzzy relational equations in their generic forms are non-linear equations in which the relations determine the mappings between the associated pairs of input and output fuzzy sets. Each equation corresponds to a single rule and a system of equations results when there is a set of rules. Thus, encoding a body of fuzzy rule knowledge by a fuzzy relation requires solving of a system of equations. In the theoretical studies of FRE, focuses have been put on characterizing the families of solutions of equations [31,106], investigating different types of equations [82,89,93], and determining the solutions of a variety of equations and systems of them [32,89,94]. In particular, several analytical resolution methods have been successfully developed to address the last problem. A key assumption was often made in the development that a solution exists, i.e. the system of equations is solvable. The usual understanding on the solvable condition of a system of max-t type FREs is that the input fuzzy sets must be normal and pairwisely disjoint (max-min orthogonal) [33,44,92,97]. As already mentioned, this is often not the case in practical applications and recent developments have been focused on solving the equations approximately [43,44,92].

Recently, it has been shown that FREs are structurally similar to a two-layer feedforward network and hence can be implemented by parallel hardware and solved via a network training process [95,97]. Since it is a direct implementation of the FREs, the storage capacity of the resulted fuzzy relational neural system (FRNS) model remains the same as that of a fuzzy relation. Consequently, it is still governed by the poorly established solvable conditions of the equation systems. In this chapter, a rigorous analysis of the storage capacity of the FRNS model is presented. The backgrounds on max-t type and min-s type FREs and FRNS model are given in the following section. The analytical methods for solving these two types of equations and system of them are introduced in Section 6.3. In Section 6.4, new solvable conditions [24] are presented and based upon which the boundary condition of solvability is derived. In Section 6.5, we show that the results can also be applied to existing analytical methods for approximate resolution of a non-solvable system of equations [25]. The capacity of the FRNS model is then elaborated using the newly established results in Section 6.6. The inference performance is reported in Section 6.7.

6.2 Fuzzy Relational Equations and Fuzzy Relational Neural Systems

In Chapter 2, we have already shown that fuzzy sets play a significant role in coping with the vagueness of concepts and forming the appropriate representations while fuzzy relations are frequently used to encode the imprecise associations between concepts as represented by fuzzy rules. Both are commonly regarded as the basic constructs to model human knowledge which have been found to be vague, imprecise, and uncertain in nature. In his seminal paper [103], Sanchez put them into a mathematical framework called fuzzy relational equation (FRE) and laid down the foundation for its resolution. There exist many different forms of FREs and an overall presentation of the subject can be found in [32,94]. In this section, the max-*t* and min-*s* FREs which are commonly used to encode fuzzy rules are reviewed. Their neural network implementations to form the FRNS model are introduced subsequently.

Let *A* and *B* be fuzzy sets defined in finite universes of discourse **U** and **V** respectively and both the universes be discrete. Then the fuzzy relation *R* is defined in $\mathbf{U}\times\mathbf{V}$, namely, $R:\mathbf{U}\times\mathbf{V}\to[0,1]$. The max-*t* and min-*s* FREs are given by

$$B(b)=\max_{a\in\mathbf{U}}[A(a)\,t\,R(a,b)] \quad \forall b\in\mathbf{V} \tag{6.1}$$

and

$$B(b)=\min_{a\in\mathbf{U}}[A(a)\,s\,R(a,b)] \quad \forall b\in\mathbf{V} \tag{6.2}$$

respectively where *t* & *s* denote the *t*-norm and *s*-norm respectively. Alternatively, they can be represented in compact form as

$$B=A\circ_t R \tag{6.3}$$

and

$$B=A\bullet_s R \tag{6.4}$$

respectively. As for the FAM matrix, they correspond to the fuzzy rule

Rule : IF the input variable is A THEN the output variable is B

It can be seen that eq.(6.3) is simply a generalized version of the compositional rule of inference in eq.(2.32). In fact, fuzzy relational equations has evolved from eq.(2.32) to include eqs.(6.3) & (6.4), the eigen FREs

$$A = A \circ_t R, \tag{6.5}$$

the adjoint FREs

$$B = A \varphi R \tag{6.6}$$

where φ is the inclusion operator [94], and other variants. The theoretical interest of the subject is to solve these equations and systems of them, e.g.,

$$B_k = A_k \circ_t R; \qquad k = 1,2,\ldots,L \tag{6.7}$$

or

$$B_k = A_k \bullet_s R; \qquad k = 1,2,\ldots,L \tag{6.8}$$

Constructive methods to do so have been successfully derived and they will be reviewed in the next section.

In view of the matrix form of FREs, Pedrycz [95,97] proposed to implement the equations by neural architecture. As shown in Fig.6.1, the resulted network model, termed as fuzzy relational neural system (FRNS), is the same as that of the FAM rule network in Fig.5.1 except that the composition operators are generalized ones, i.e., max-t. Unlike Kosko's FAM model which employs a bank of FAM rule networks to form the whole system, such a network by itself is expected to capture the set of fuzzy rules, i.e., to be able to solve a system of FREs, via an iterative descent learning algorithm. The power of FRNS model however is still governed by the capacity of fuzzy relation in FREs. In the subsequent sections, it will be analyzed through a re-examination of the solvability of a system of FREs.

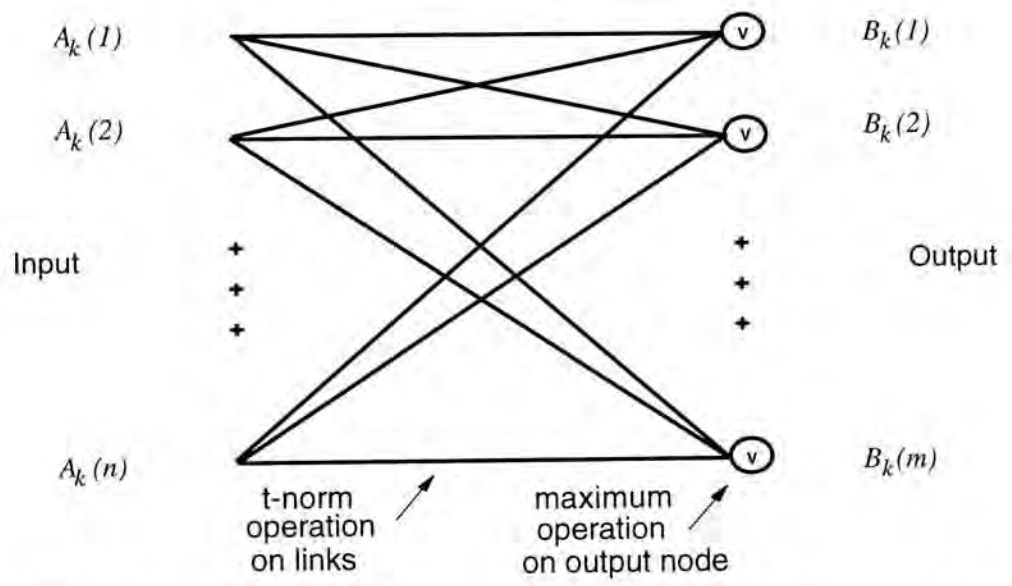


Figure 6.1 Fuzzy Relational Neural System Architecture

6.3 Solving a System of Fuzzy Relational Equations

In this section, the analytical results for solving the max-t and min-s FREs and systems of them are reviewed. Consider the problem of determining the fuzzy relation R for given A and B, i.e., the encoding problem, under the assumption that the family of solutions is non-empty, i.e., $\{R | A \circ_t R = B\} \neq \varnothing$ and $\{R | A \bullet_s R = B\} \neq \varnothing$. The greatest solutions, according to [94], are

$$R(a,b) = A(a)\,\varphi\,B(b) \tag{6.9}$$

and

$$R(a,b) = A(a)\,\beta\,B(b) \tag{6.10}$$

respectively with the φ and β operators defined as

$$p\,\varphi\,q = \sup\{w \in [0,1] | p\,t\,w \leq q\} \tag{6.11}$$

and

$$p\beta q = \inf\{w \in [0,1] | p s w \geq q\} \tag{6.12}$$

For the t-norm of eq.(6.11) and s-norm of eq.(6.12) specified as minimum and maximum operators respectively, the φ-operator will become the α-operator [94]

$$p\alpha q = \sup\{w \in [0,1] | p \wedge w \leq q\} = \begin{cases} 1 & \text{if } p \leq q \\ q & \text{if } p > q \end{cases} \tag{6.13}$$

and the β operator will become the ε operator [94]

$$p\varepsilon q = \inf\{w \in [0,1] | p \vee w \geq q\} = \begin{cases} q & \text{if } p < q \\ 0 & \text{if } p \geq q \end{cases} \tag{6.14}$$

For the max-t FREs, the condition

$$\forall b \in \mathbf{V}, \exists a \in \mathbf{U}: A(a) \geq B(b) \tag{6.15}$$

must be satisfied in order to guarantee the existence of a solution [92]. Such a solvable condition of a single equation is easy to check and always holds true if A is a normal fuzzy set. Similar condition can also be derived for the min-s equations. Suppose now the systems of equations depicted in eqs.(6.7) & (6.8) are to be solved. Based upon the assumption that a solution exists, i.e., $\{R | A_k \circ_t R = B_k \ \ \forall k\} \neq \varnothing$ and $\{R | A_k \bullet_s R = B_k \ \ \forall k\} \neq \varnothing$, the greatest solutions are [94]

$$R = \min_{k=1}^{L}[A_k \varphi B_k] \tag{6.16}$$

and

$$R = \max_{k=1}^{L}[A_k \beta B_k] \tag{6.17}$$

respectively where $A_k \varphi B_k$ and $A_k \beta B_k$ are the compact representations of eqs.(6.9) & (6.10).

As mentioned in the introductory section, the requirement for a system of max-t equations to be solvable, i.e., the existence of a solution, is usually assumed to be that the input fuzzy sets A_k are normal and pairwisely disjoint [33,44,92,97]. The latter condition is equivalent to the max-min orthogonal condition described in definition 5.1. An example with triangular membership functions has been shown in Fig.6.2. They are not common in practice. As the min-s FRE is dual to the max-t

FRE, its solvable conditions are the complement of these two conditions. In the following section, we will re-examine the solvable conditions of a system of max-t FREs and those of the min-s equations as well.

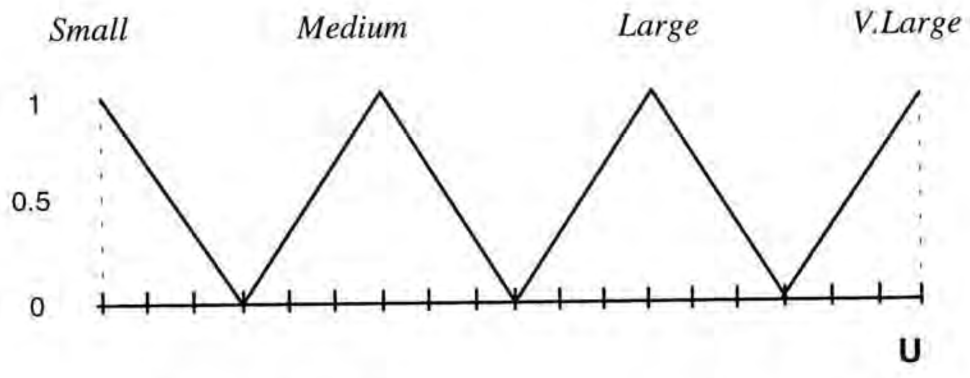
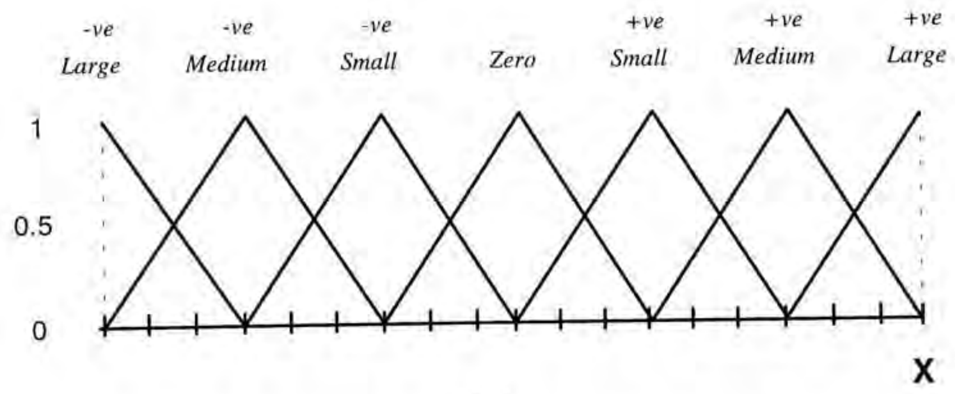
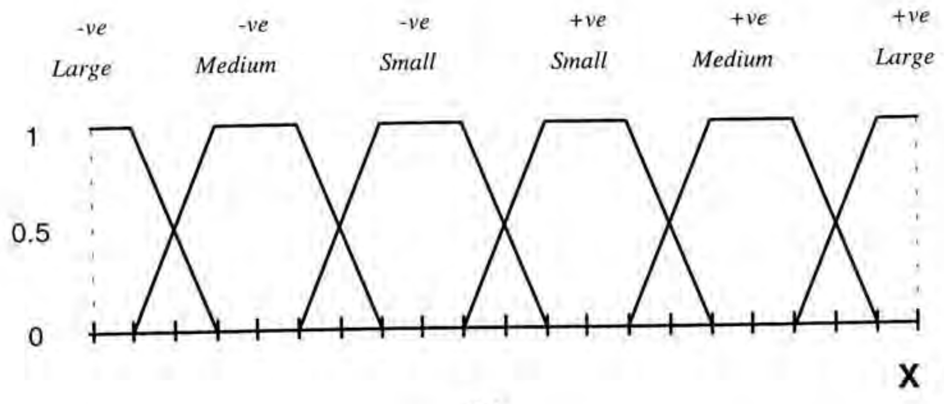


Figure 6.2 Pairwisely Disjoint Fuzzy Sets



(a)



(b)

Figure 6.3 Triangular Type and Trapezoidal Type Semi-Overlapping Fuzzy Sets

6.4 New Solvable Conditions

6.4.1 max-*t* Fuzzy Relational Equations

In contrast with pairwise disjoint condition depicted in Fig.6.2, the fuzzy sets A_k of typical fuzzy rule-based systems are pairwisely overlapped. In particular, they are usually semi-overlapping, as mentioned in Chapter 5, satisfying the conditions described by eqs.(5.13) & (5.14). Examples for triangular and trapezoidal membership functions are shown in Fig.6.3. With such distributions defined, a new solvable condition, though not necessarily a boundary one (to be discussed later), for a system of max-*t* FREs is established below.

<u>*Theorem 6.1*</u> : Given a system of max-*t* FREs

$$B_k = A_k \circ_t R; \quad k = 1,2,\dots L \tag{6.18}$$

then the system of equations can be exactly solved using

$$R = \min_{k=1}^{L}[A_k \varphi B_k] \tag{6.19}$$

i.e., Pedrycz's greatest solution [94], if the input fuzzy sets A_k are normal and semi-overlapping.

<u>*Proof*</u> : By performing the max-*t* composition $A_l \circ_t R$ for any input fuzzy set A_l, the element-wise output is

$$\begin{aligned}\hat{B}_l(b) &= \max_a\left\{A_l(a)t\left[\min_{k=1}^{L}(A_k(a)\varphi B_k(b))\right]\right\} \\ &= \max_a\left\{\min_{k=1}^{L}\left[A_l(a)t(A_k(a)\varphi B_k(b))\right]\right\}\end{aligned} \tag{6.20}$$

Dividing the input universe of discourse $\mathbf{U}$ into

$$S_l = \{a | A_l(a) = 1\} \tag{6.21}$$

and

$$S_l' = \mathbf{U} - S_l \tag{6.22}$$

we have

$$
\begin{aligned}
\hat{B}_l(b) &= \max_{a\in S_l}\left\{\min_{k=1}^{L}\left[1t(A_k(a)\varphi B_k(b))\right]\right\}\vee \\
&\quad \max_{a\notin S_l}\left\{\min_{k=1}^{L}\left[A_l(a)t(A_k(a)\varphi B_k(b))\right]\right\} \\
&= \max_{a\in S_l}\left\{\min_{k=1}^{L}\left[A_k(a)\varphi B_k(b)\right]\right\}\vee \\
&\quad \max_{a\notin S_l}\left\{\min_{k=1}^{L}\left[A_l(a)t(A_k(a)\varphi B_k(b))\right]\right\}
\end{aligned}
\tag{6.23}
$$

According to the semi-overlapping conditions,

$$A_k(a)=0 \quad \forall a\in S_l,\, k\neq l \tag{6.24}$$

Hence,

$$
\begin{aligned}
\hat{B}_l(b) &= \max_{a\in S_l}\left\{\min_{k\neq l}\left[0\varphi B_k(b)\right]\wedge\left[1\varphi B_l(b)\right]\right\}\vee \\
&\quad \max_{a\notin S_l}\left\{\min_{k=1}^{L}\left[A_l(a)t(A_k(a)\varphi B_k(b))\right]\right\} \\
&= B_l(b)\vee\max_{a\notin S_l}\left\{\min_{k\neq l}\left[A_l(a)t(A_k(a)\varphi B_k(b))\right]\wedge\right. \\
&\qquad\qquad \left.\left[A_l(a)t(A_l(a)\varphi B_l(b))\right]\right\}
\end{aligned}
\tag{6.25}
$$

Since $pt(p\varphi q)\leq q$ [89], the maximum term of eq.(6.25) is less than or equal to $B_l(b)$ and therefore

$$\hat{B}_l(b)=B_l(b) \tag{6.26}$$

That is, the output fuzzy set B_l is recalled and the system of max-t FREs is exactly solved. **Q.E.D.**

According to this theorem, the usual pairwisely disjoint condition for solving a system of max-t FREs is too conservative. More importantly, the semi-overlapping condition can be satisfied in most rule-based system applications and the theorem is instructive to the design and implementation of fuzzy rulebases, i.e., if the input fuzzy sets are designed to be semi-overlapping, the rulebase can be merely implemented by a single fuzzy relation, i.e., a FRNS. In other words, the fuzzy knowledge can be captured by the FRNS model via a deterministic encoding process rather than an iterative training process. Furthermore, as illustrated in Fig.6.4, a chained fuzzy rulebase can be implemented by cascaded FRNS. With appropriate parallel hardware realization, the computational speed of such system could be particularly fast.

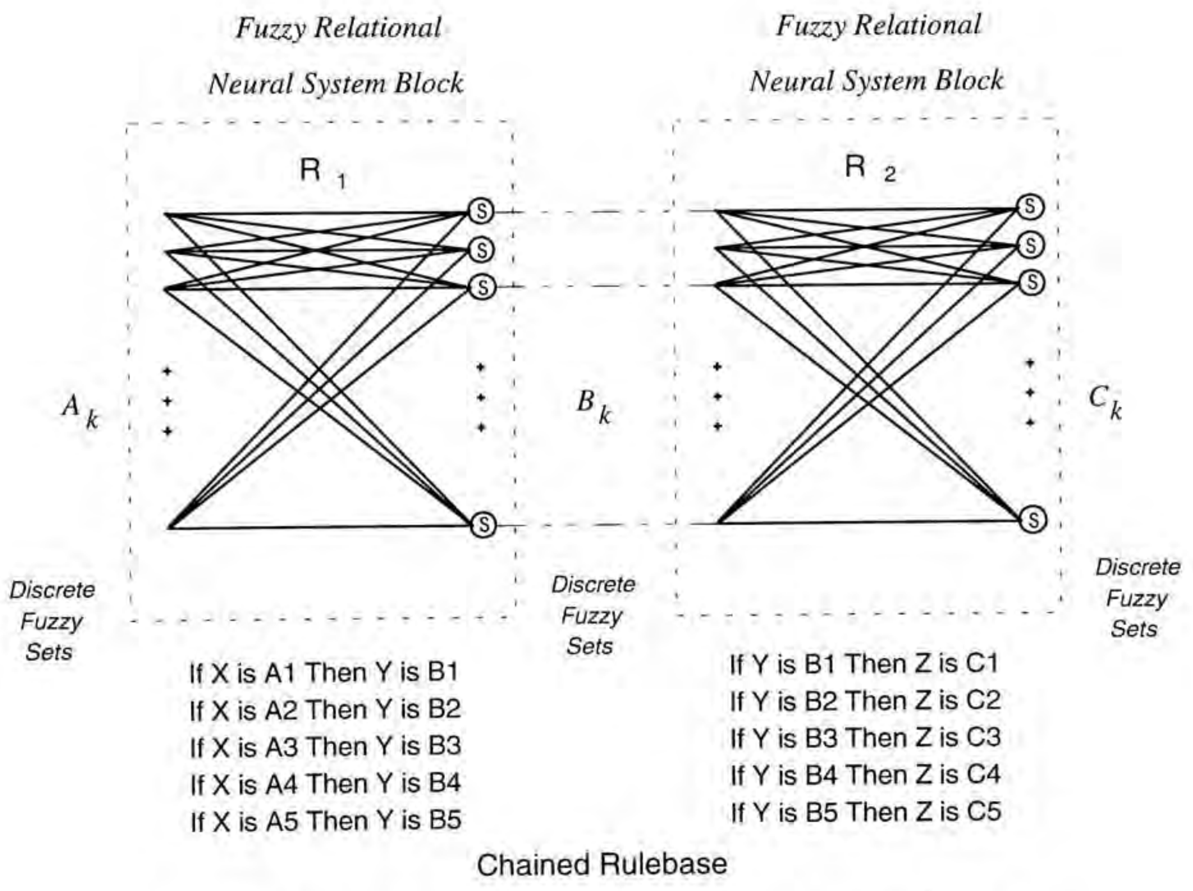


Figure 6.4 Cascaded Fuzzy Relational Neural Systems

It can be deduced from the proof of Theorem 6.1 that the system of equations is solvable as long as the cardinalities of S_l are equal to or greater than one. Otherwise, eq.(6.25) becomes

$$\hat{B}_l(b) = \max_a \left\{ \min_{k \neq l} \left[A_l(a) t (A_k(a) \varphi B_k(b)) \right] \wedge \left[A_l(a) t (A_l(a) \varphi B_l(b)) \right] \right\} \quad (6.27)$$

Hence,

$$\hat{B}_l(b) \leq B_l(b) \quad (6.28)$$

and the solvability is not guaranteed. Recall from eq.(6.21)&(6.24) that S_l consists of the points in the discrete universe of discourse with full membership attached to the input fuzzy set A_l and null membership attached to all the other input fuzzy sets, viz.

$$S_l = \{a \mid A_l(a) = 1, A_k(a) = 0 \ \forall k \neq l\} \tag{6.29}$$

For the triangular membership function example in Fig.6.3(a), the cardinalities of S_l's are one and therefore the semi-overlapping condition is also the boundary condition, beyond which the solvability of the system of equations cannot be guaranteed. This is not the case for the trapezoidal membership functions. As exemplified in Fig.6.3(b), the cardinalities of S_l's are three. Therefore, the distribution of the input fuzzy sets can be made more compact while the system of equations is still solvable. In the limited case, the cardinalities of S_l's are one as shown in Fig.6.5. Thus, the boundary condition of solvability can be formally defined as follows.

Corollary 6.1 : For a system of max-t FREs in eq.(6.18), the boundary condition of solvability is the existence of an *exclusive* point in the universe of discourse for each input fuzzy set, i.e., a point with full membership for the corresponding input fuzzy set and null membership for all the others. Mathematically, it is referred to S_l in eq.(6.29) consisting of one element only for all the input fuzzy sets A_l.

Such condition is easy to check and will be used to study the capacity of FRNS model in Section 6.6.

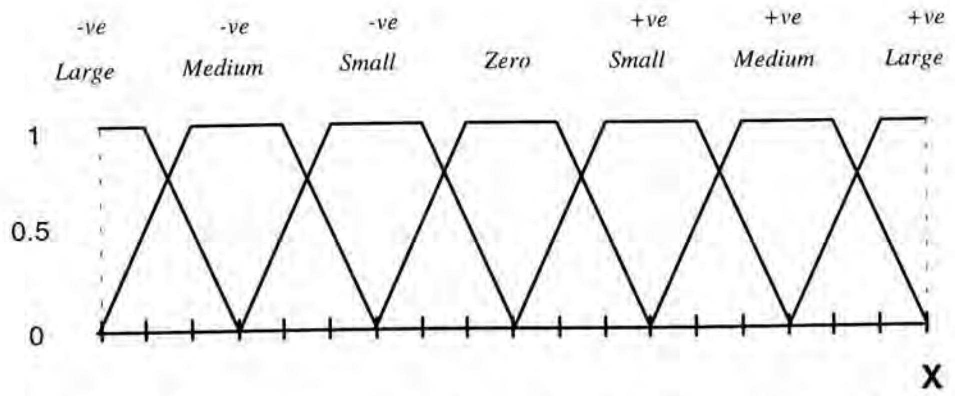


Figure 6.5 Distribution of Trapezoidal Fuzzy Sets Satisfying the Boundary Condition of Solvability

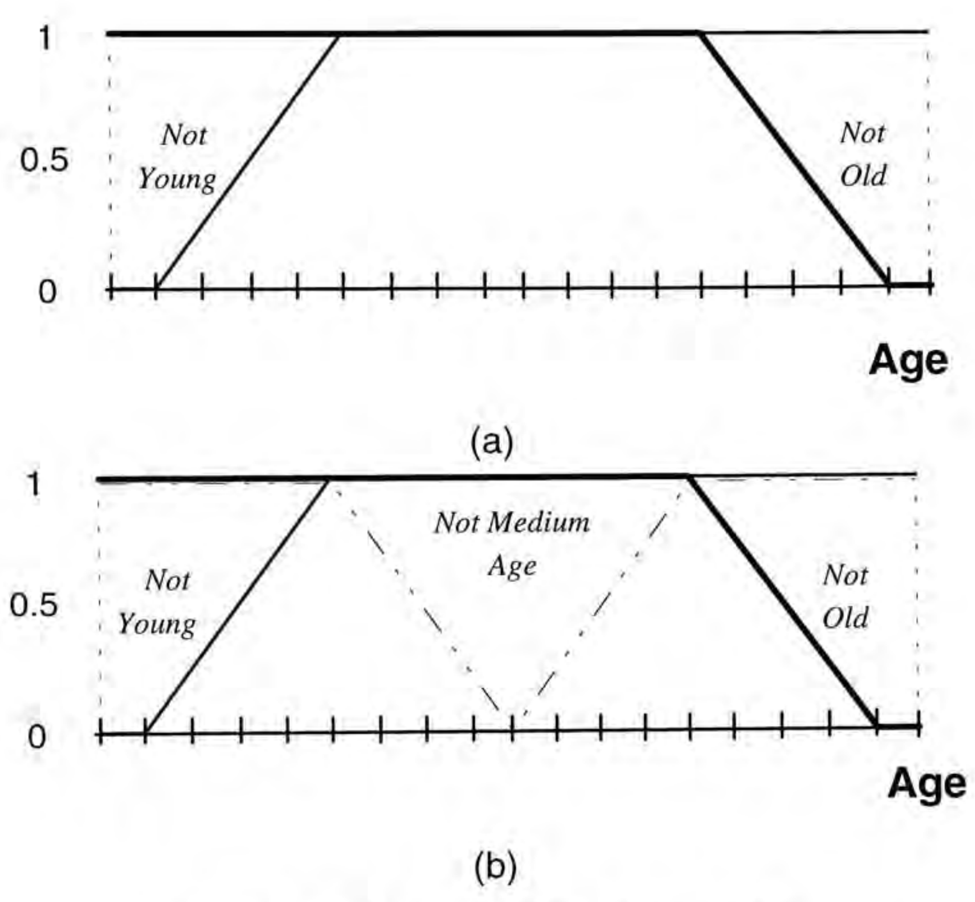


Figure 6.6 Negative Type Fuzzy Sets

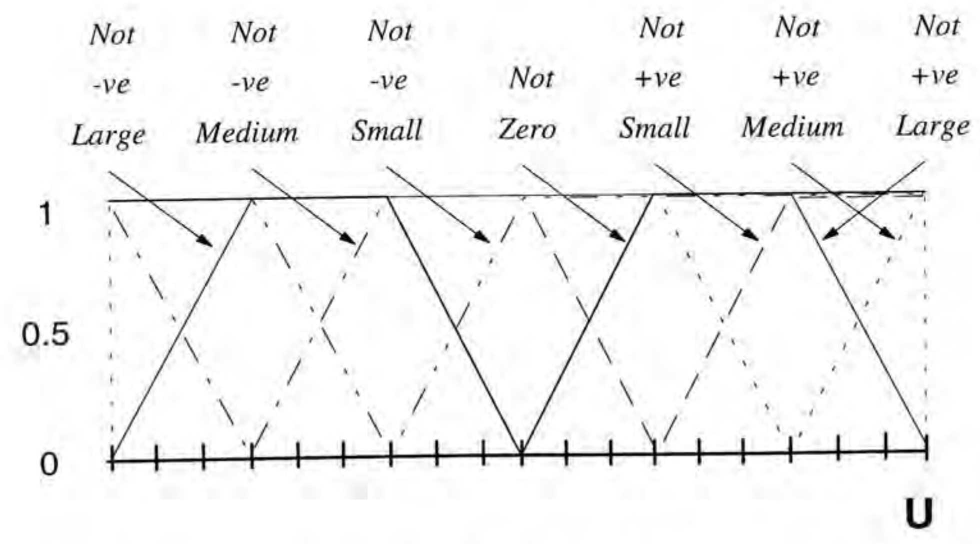


Figure 6.7 Inversely Semi-Overlapped Triangular Fuzzy Sets

6.4.2 min-*s* Fuzzy Relational Equations

According to corollary 6.1, the max-*t* FREs are solvable not only for the semi-overlapping fuzzy sets in Fig.6.3, but also for the negative type input fuzzy sets in Fig.6.6(a). However, they are no longer solvable for the fuzzy sets depicted in Fig.6.6(b). It has been discovered that such types of input fuzzy sets can be encoded easily with the min-*s* FREs, according to Theorem 6.2 below.

<u>*Theorem 6.2*</u> : Given a system of min-*s* FREs

$$B_k = A_k \bullet_s R; \quad k = 1,2,...,L \tag{6.30}$$

then the system of equations can be exactly solved using

$$R = \max_{k=1}^{L}[A_k \,\beta\, B_k] \tag{6.31}$$

i.e., Pedrycz's greatest solution [94], if the input fuzzy sets A_k are inversely semi-overlapping as exemplified in Fig.6.7.

<u>*Proof*</u> : By performing the min-*s* composition $A_l \bullet_s R$ for any input fuzzy set A_l, the element-wise output is

$$\begin{aligned}\hat{B}_l(b) &= \min_a\left\{A_l(a)\, s\left[\max_{k=1}^{L}(A_k(a)\beta B_k(b))\right]\right\}\\ &= \min_a\left\{\max_{k=1}^{L}\left[A_l(a)\, s(A_k(a)\beta B_k(b))\right]\right\}\end{aligned} \tag{6.32}$$

Dividing the input universe of discourse **U** into

$$\Omega_l = \{a \mid A_l(a) = 0\} \tag{6.33}$$

and

$$\Omega_l' = \mathbf{U} - \Omega_l \tag{6.34}$$

we have

$$\begin{aligned}\hat{B}_l(b) = &\min_{a\in\Omega_l}\left\{\max_{k=1}^{L}\left[0\, s(A_k(a)\beta B_k(b))\right]\right\}\wedge\\ &\min_{a\notin\Omega_l}\left\{\max_{k=1}^{L}\left[A_l(a)\, s(A_k(a)\beta B_k(b))\right]\right\}\end{aligned} \tag{6.35}$$

According to the inversely semi-overlapped condition depicted in Fig.6.7, we have

$$A_k(a)=1 \quad \forall a \in \Omega_l,\, k \neq l \tag{6.36}$$

Hence,

$$\begin{aligned}\hat{B}_l(b) &= \min_{a\in\Omega_l}\left\{\max_{k\neq l}\left[1\beta B_k(b)\right]\vee\left[0\beta B_k(b)\right]\right\}\wedge \\ &\quad \min_{a\notin\Omega_l}\left\{\max_{k=1}^{L}\left[A_l(a)s(A_k(a)\beta B_k(b))\right]\right\} \\ &= B_l(b)\wedge\min_{a\notin\Omega_l}\left\{\max_{k\neq l}\left[A_l(a)s(A_k(a)\beta B_k(b))\right]\vee\right. \\ &\qquad \left.\left[A_l(a)s(A_l(a)\beta B_l(b))\right]\right\}\end{aligned} \tag{6.37}$$

Since $p\,s(p\beta q)\geq q$, the minimum term of eq.(6.37) is greater than or equal to $B_l(b)$ and therefore

$$\hat{B}_l(b)=B_l(b) \tag{6.38}$$

That is, the output fuzzy set B_l is recalled and the system of min-*s* FREs is exactly solved. **Q.E.D.**

Similarly, a boundary condition of solvability for the min-*s* equations can be deduced from the proof of Theorem 6.2.

<u>*Corollary 6.2*</u> : For a system of min-*s* FREs in eq.(6.30), the boundary condition of solvability is the existence of an *sacrificial* point in the universe of discourse for each input fuzzy set A_l, i.e., a point with null membership for the corresponding input fuzzy set and full membership for all the others. Again, it is referred to

$$\Omega_l=\{a\,|\,A_l(a)=0, A_k(a)=1 \ \forall k\neq l\} \tag{6.39}$$

whose cardinality is equal to one.

According to this boundary condition, the min-*s* FREs are solvable for the *negative* type fuzzy sets in Fig.6.6 and the inversely semi-overlapped fuzzy sets in Fig.6.7. To the contrary, they are not suitable for the *positive* type fuzzy sets in Fig.6.3. On that basis, it is suggested to use the max-*t* FREs to encode fuzzy rulebases with positive type fuzzy sets and the min-*s* FREs to encode fuzzy rulebases with negative type fuzzy sets.

6.5 Approximate Resolution

We have already shown that a system of max-t FREs (or min-s FREs) can be exactly solved when the distribution of the input fuzzy sets is within the boundary condition. Should it fall beyond the boundary condition, an exact solution cannot be expected. In fact, substantial works [43,44,92] on solving a system of FREs approximately when it is non-solvable have been reported. In this section, we describe how the established results can be applied to two existing approaches to approximate resolution. We concentrate on the max-t FREs only. The corresponding results for min-s FREs can be derived in a similar manner.

With respect to the system of equations being non-solvable, Gottwald and Pedrycz [43] have proposed two approaches to modify the input and output fuzzy sets such that the solvability of the resulted system is increased. One is to replace the fuzzy sets by "fuzzified" versions

$$A_k^{\alpha}(a) = \max(A_k(a), \alpha) = \begin{cases} A_k(a) & \text{if } A_k(a) \geq \alpha \\ \alpha & \text{if } A_k(a) < \alpha \end{cases} \tag{6.40}$$

$$B_k^{\alpha}(b) = \max(B_k(b), \alpha) = \begin{cases} B_k(b) & \text{if } B_k(b) \geq \alpha \\ \alpha & \text{if } B_k(b) < \alpha \end{cases} \tag{6.41}$$

and the other follows the opposite direction to replace them by "sharpened" versions

$$A_i^{\bar{\alpha}}(a) = \begin{cases} A_k(a) & \text{if } A_k(a) \geq \alpha \\ 0 & \text{if } A_k(a) < \alpha \end{cases} \tag{6.42}$$

$$B_k^{\bar{\alpha}}(b) = \begin{cases} B_k(b) & \text{if } B_k(b) \geq \alpha \\ 0 & \text{if } B_k(b) < \alpha \end{cases} \tag{6.43}$$

However, ways to specify an appropriate threshold value α such that the modified system of equations can be exactly solved are lacking. In this regard, the new concepts presented in the previous section can be applied as follows.

Theorem 6.3 : Given a system of max-t FREs

$$B_k = A_k \circ_t R; \quad k = 1,2,\ldots,L \tag{6.44}$$

then the fuzzified system of equations

$$B_k^\alpha = A_k^\alpha \circ_t R; \quad k = 1,2,\ldots,L \tag{6.45}$$

can be exactly solved using

$$R = \min_{k=1}^{L} A_k^\alpha \varphi B_k^\alpha \tag{6.46}$$

if α is set as

$$\alpha = \max_{k=1}^{L} g_k = \max_{k=1}^{L} \max_{\substack{l \neq k, \\ a \in \{a | A_k(a)=1\}}} A_l(a) \tag{6.47}$$

whose computation has been depicted pictorially in Fig.6.8, .

Proof : By performing the max-t composition $A_l^\alpha \circ_t R$ for any fuzzified input fuzzy set A_l^α, the element-wise output is

$$\begin{aligned} \hat{B}_l(b) &= \max_a \left\{ A_l^\alpha(a) t \left[\min_{k=1}^{L} (A_k^\alpha(a) \varphi B_k^\alpha(b)) \right] \right\} \\ &= \max_a \left\{ \min_{k=1}^{L} \left[A_l^\alpha(a) t (A_k^\alpha(a) \varphi B_k^\alpha(b)) \right] \right\} \end{aligned} \tag{6.48}$$

Dividing the input universe of discourse $\mathbf{U}$ into

$$S_l = \{a | A_l^\alpha(a) = 1\} \tag{6.49}$$

and

$$S_l' = \mathbf{U} - S_l \tag{6.50}$$

we have

$$\begin{aligned} \hat{B}_l(b) &= \max_{a \in S_l} \left\{ \min_{k=1}^{L} \left[A_k^\alpha(a) \varphi B_k^\alpha(b) \right] \right\} \vee \\ &\quad \max_{a \notin S_l} \left\{ \min_{k=1}^{L} \left[A_l^\alpha(a) t (A_k^\alpha(a) \varphi B_k^\alpha(b)) \right] \right\} \\ &= \max_{a \in S_l} \left\{ \min_{k \neq l} \left[A_k^\alpha(a) \varphi B_k^\alpha(b) \right] \wedge \left[1 \varphi B_l^\alpha(b) \right] \right\} \vee \\ &\quad \max_{a \notin S_l} \left\{ \min_{k=1}^{L} \left[A_l^\alpha(a) t (A_k^\alpha(a) \varphi B_k^\alpha(b)) \right] \right\} \end{aligned} \tag{6.51}$$

According to eq.(6.47),

$$A_k^\alpha(a) \leq \alpha \quad \forall a \in S_l, k \neq l \tag{6.52}$$

the element-wise output will be

$$\hat{B}_l(b) = B_l^\alpha(b) \vee \max_{a \notin S_l}\Big\{\min_{k \neq l}\big[A_l^\alpha(a)\,t\,(A_k^\alpha(a)\,\varphi\, B_k^\alpha(b))\big] \wedge \big[A_l^\alpha(a)\,t\,(A_l^\alpha(a)\,\varphi\, B_l^\alpha(b))\big]\Big\} \tag{6.53}$$

As $p\,t(p\varphi q) \leq q$, the maximum term of eq.(6.53) is less than or equal to $B_l^\alpha(b)$ and therefore

$$\hat{B}_l(b) = B_l^\alpha(b) \tag{6.54}$$

That is, the desired fuzzified output fuzzy set B_l^α is recalled and the fuzzified system of equations is exactly solved. **Q.E.D.**

Using the computed threshold value, i.e., g_6 or g_7 in Fig.6.8, the fuzzified input fuzzy sets are shown in Fig.6.9(a) and the corresponding system of equations, according to Theorem 6.3, can be solved exactly.

The application of the new concepts to solve the sharpened system of equations exactly is straight forward. As exemplified in Fig.6.9(b), the input fuzzy sets sharpened as shown will satisfy the boundary condition of solvability for max-*t* FREs in Corollary 6.1. Therefore, the sharpened system of equations can be solved exactly. Unlike the fuzzified approach where the outputs are fuzzified (approximate) versions of the original ones, the outputs of the sharpened approach can be exactly the same as the original ones.

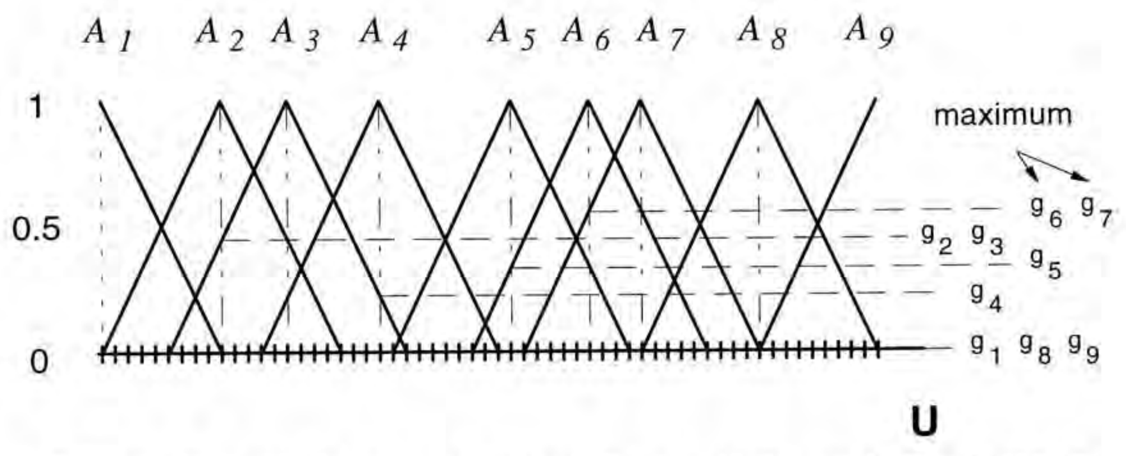
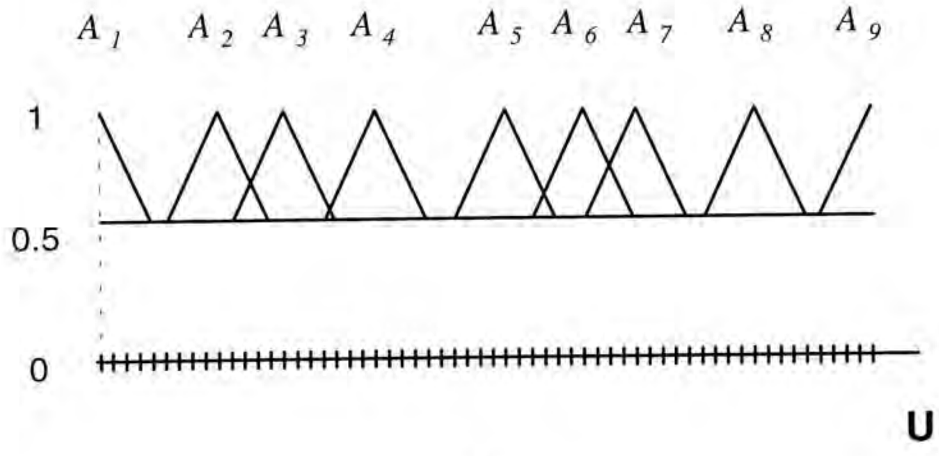
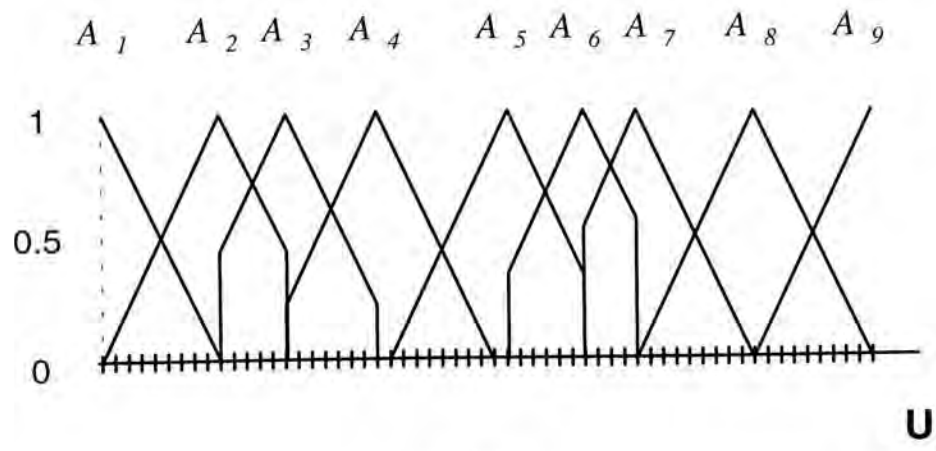


Figure 6.8 Highly Overlapped Fuzzy Sets and the Pictorial Representation of g_k



(a)



(b)

Figure 6.9 (a) Fuzzified and (b) Sharped Highly Overlapped Fuzzy Sets

6.6 System Capacity

The capacity of the FRNS model is referred to the maximum number of fuzzy rules that can be stored. It is equivalent to the maximum number of FREs that can be exactly solved by a fuzzy relation. Therefore, it is closely related to the solvability of a system of equations and indeed can be derived by checking the solvability condition. In this section, it is studied using the results established in the previous sections.

Like the solvability of a system of FREs, the capacity of fuzzy relations under max-*t* compositions is limited by the boundary condition. The exact quantity however depends on the type and resolution of the input fuzzy sets. Here are the results for the symmetric triangular, singleton, and symmetric trapezoidal membership functions. A discrete and uniform universe of discourse $\mathbf{U}=\{u_1,u_2,...,u_n\}$ is assumed.

(C1). A fuzzy set A_k defined by triangular membership function has the form

$$\mu_{A_k}(u)=\begin{cases}(u-u_L)/(u_M-u_L) & \text{if } u_L<u\le u_M\\ (u_R-u)/(u_R-u_M) & \text{if } u_M<u<u_R\\ 0 & \text{otherwise}\end{cases} \tag{6.55}$$

where $u,u_L,u_M,u_R\in\mathbf{U}$, $\mu_{A_k}(u)$ are nonzero only on the interval (u_L,u_R), and full membership occurs only at the midpoint u_M of A_k. If it is a symmetric one, $u_R-u_M=u_M-u_L$ holds true. Thus, the capacity under max-*t* composition for symmetric triangular input fuzzy sets (STriIFS) is

$$Capacity(\text{max-}t,\ \text{STriIFS}) = trunc\left(\frac{u_n-u_1}{u_R-u_M}\right)+1 \tag{6.56}$$

where $trunc(\cdot)$ is the truncation operator. It depends on the resolution of the input fuzzy sets, parameterized by u_R-u_M. For the example in Fig.6.3(a), the capacity is equal to $18(u_i-u_{i-1})/3(u_i-u_{i-1})+1=7$.

(C2). When the STriIFS are degenerated to $u_R-u_M=u_i-u_{i-1}$, i.e. singleton fuzzy sets in the discrete universe of discourse, the capacity under max-*t* composition is

$$Capacity(\text{max-}t,\ \text{singleton}) = \frac{u_n - u_1}{u_i - u_{i-1}} + 1 = \text{card}\{\mathbf{U}\} \tag{6.57}$$

i.e., the cardinality of the input universe of discourse.

(C3). A fuzzy set A_k defined by trapezoidal membership function has the form

$$\mu_{A_k}(u) = \begin{cases} (u-u_a)/(u_b-u_a) & \textit{if } u_a < u < u_b \\ 1 & \textit{if } u_b \le u \le u_c \\ (u_d-u)/(u_d-u_c) & \textit{if } u_c < u < u_d \\ 0 & \textit{otherwise} \end{cases} \tag{6.58}$$

where $u, u_a, u_b, u_c, u_d \in \mathbf{U}$, $\mu_{A_k}(u)$ are nonzero only on the interval (u_a, u_d), and full membership occurs on the interval (u_b, u_c). If it is a symmetric one, $u_b - u_a = u_d - u_c$ holds true. Thus, the capacity of fuzzy relations under max-*t* composition for symmetric trapezoidal input fuzzy sets (STraIFS) is

$$Capacity(\text{max-}t,\ \text{STraIFS}) =$$
$$trunc\left(\frac{u_n - u_1}{(u_b - u_a) + trunc[0.5(u_c - u_b)]}\right) + 1 \tag{6.59}$$

Again, it depends on the resolution of the input fuzzy sets. For the example in Fig.6.3(b), the capacity is equal to $18(u_i - u_{i-1})/(2+1)(u_i - u_{i-1}) + 1 = 7$.

Similar results can be obtained for the other types of fuzzy sets and the min-*s* FREs. In all cases, the fuzzier the input fuzzy sets, the lower the capacity is. Furthermore, the maximum capacity among them is equal to the cardinality of the input universe of discourse.

6.7 Inference Performance

As a comparison to the FAM model, this section reports the inference performance of the FRNS model for the three types of fuzzy inputs described in Section 5.7 using the same rulebase. For input "X is *medium*", the inference output depicted in Fig.6.10 confirms that the FRNS model like the max-$\otimes$ FAM can perfectly recall the desired output "Y is *very small*". The inference output for input "X is *more-or-less medium*" is shown in Fig.6.11 and its defuzzified value using COG method is $\approx$9.3, which is larger than that of the max-$\otimes$ FAM ($\approx$6.4) and smaller than that of Kosko's model ($\approx$ 11.2). It is hard to say whether this output is better or worse than that of the max-$\otimes$ or Kosko's FAM model. As for the max-$\otimes$ FAM, the inference output of FRNS for input "X is *very medium*" is *very small.* For singleton fuzzy input representing crisp input value x=15, it was found that a null output fuzzy set (zero memberships) was produced. As shown by the contour plot of the fuzzy relation of FRNS in Fig.6.12, there is a gap between the range [13,17] of X which in turn will produce null output for all crisp values in the range. This is a very undesirable performance and hence FRNS is not suitable for inference with crisp inputs.

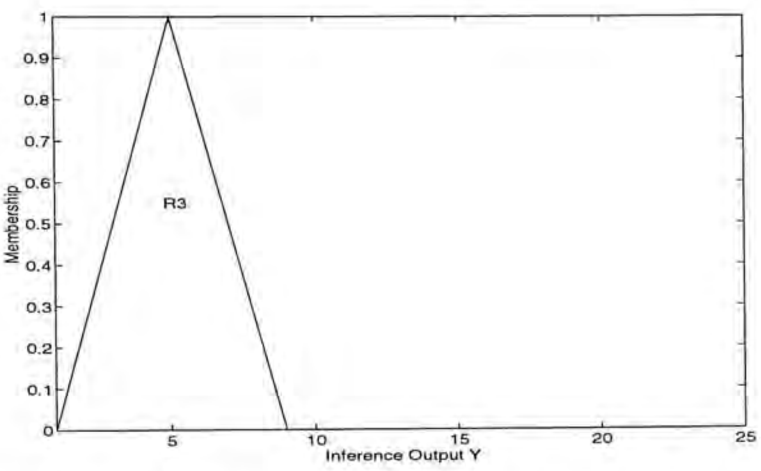


Figure 6.10 The Inference Output of "X is *medium*" and "X is *very medium*" : FRNS Model

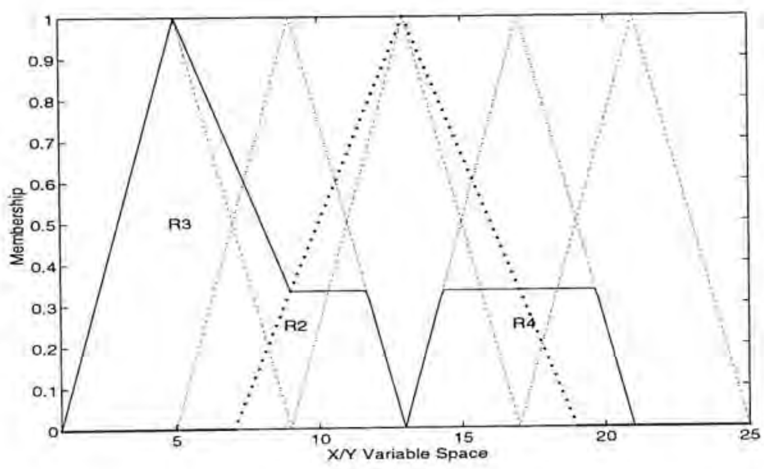


Figure 6.11 The Inference Output of "X is *more-or-less medium*" : FRNS Model

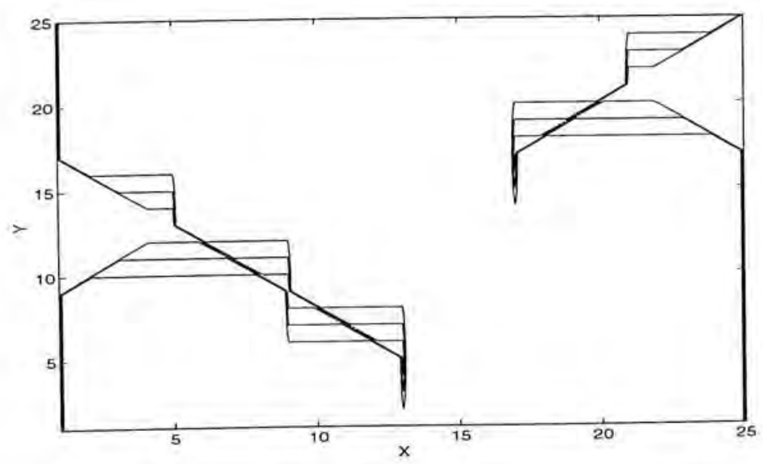


Figure 6.12 Contour Plot of the Fuzzy Relation in FRNS Model

6.8 Concluding Remarks

In this chapter, a rigorous analysis of the storage capacity of the so-called fuzzy relational neural system (FRNS) model has been conducted. New solvable conditions on the input fuzzy sets of a system of max-t FREs and a system of min-s FREs have been identified and they point out that the existing ones have been too conservative. The new conditions are found to conform to the distribution of the input fuzzy sets that are common in rule-based system applications. This implies that a FRNS is sufficient to encode the given fuzzy rulebase in most applications and moreover, such system can be constructed deterministically, i.e., without involving any training process. The established theorems were found to be instructive to the operations of two existing methods for approximate resolution of a non-solvable system of equations and to formally determine the capacity of FRNS under different types and resolutions of input fuzzy sets. The inference performance of the FRNS model is comparable to the max-$\otimes$ FAM model for ordinary and modified fuzzy inputs. However, it indicates that FRNS is not suitable for inference with crisp inputs.

Throughout the chapter, we have been working on the basic type of FREs that involves set-relation compositions. If there exists more than one, say N, input variables in the available fuzzy rules, i.e., multiandecedent fuzzy rules, the corresponding FREs will involve relation-relation compositions and the input relation will cover the product space of the input variables, i.e., X : $\mathbf{U}_1 \times \mathbf{U}_2 \times \cdots \mathbf{U}_N \rightarrow [0,1]$. In that case, the derived results have to be extended, fortunately in a straight forward manner by taking the advantages of the independency between different input variable space. For example in Theorem 6.1, the semi-overlapping condition has to be satisfied by all the input fuzzy variables in order to have the system of equations solvable. On the other hand, the capacity of FRNS will be equal to the product of the capacity in the individual input space, i.e., eqs.(6.56), (6.57), or (6.59).

In these two chapters, we have dealt with the encoding problem of the FAM and FRNS models, i.e., how to form an appropriate neural network representation for a given set of fuzzy rules as represented by discrete fuzzy set pairs. Thus, the established results can be applied to the construction of rule-based systems. In the next chapter, we will focus on the identification problem of the FRNS model where the available information is a set of input-output crisp data, the fuzzy inputs/outputs are in reference fuzzy set format [91] rather the discrete one, and the objective is to construct a FRNS capable to capture the dynamic of the system being modelled.

Chapter 7

Structure and Parameter Identifications of Fuzzy Relational Neural Systems

7.1 Introduction

Applications of fuzzy relational equations (FREs) are commonly found in fuzzy control [97] and most of which have been devoted to nonlinear dynamic system modelling [91,73,113] with FREs to describe the static and dynamic behaviour of the systems. The identification process is similar to the conventional ones by observing the input-output crisp data pairs and should involve both the determination of structure and the estimation of parameters. However, little has been done in the area of structure identification, as compared with parameter identification which is simply the resolution of a system of equations, viz. estimation of the matrix elements in the fuzzy relations. Since the number of input variables in such applications is quite large, the discrete representation for the input-output fuzzy sets of the FREs will consume a lot of memory and make the system not feasible for implementation. In order to reduce the memory requirement and provide an unified treatment of crisp and fuzzy forms of information, Pedrycz [91] proposed to modify the discrete fuzzy set representation in FREs to a reference fuzzy set one. Under the concept of reference fuzzy sets, each FRE corresponds to an input-output crisp data pair and consequently the equation system being solved may consist of tens or thousands equations. That makes the system of equations far from being solvable and hence numerical techniques [73,88] rather than the analytical methods are used to obtain a good approximate solution. Both off-line and on-line identification algorithms have been successfully developed. For examples, Pedrycz [88] has proposed to use the Newton's iteration scheme to minimise the system's squared error in an off-line manner while Lee et al. [73] take use of the linguistic approach to form an initial fuzzy relation and update it by the prediction-error method upon the arrival of new data. Similar on-line algorithms, adopting some sorts of learn-and-forget strategies, can also be found in [12,26,113]. Besides, with the introduction of the FRNS model, neurocomputational approaches have also been suggested to achieve both off-line and on-line identifications [14,83,95].

In view of the absence of an effective structure identification algorithm for FRNS and the success of evolutionary computation approach to determine the topology of neural networks [115], an evolutionary identification (EVIDENT) algorithm [23,25] based on Yip and Pao's guided evolutionary simulated annealing (GESA) method [116,117] is developed in this chapter. Through experiments with various benchmarking data sets, the new algorithm is demonstrated to be effective in both structure and parameter identifications. In the following section, an overview of applying FRNS to dynamic system modelling is provided. A general FRNS identification algorithm is described in Section 7.3. In Section 7.4, the backgrounds on evolutionary computation and GESA algorithm are reviewed and the EVIDENT algorithm is derived subsequently. The simulation results are reported in Section 7.5 and the final section offers the conclusion.

7.2 Modelling Nonlinear Dynamic Systems by Fuzzy Relational Equations

It is commonly accepted that fuzzy modelling is useful when confronted with a dynamic system whose mathematical structure are unknown, ill-defined or too complex for analysis by conventional techniques. Without loss of generalization, the systems being modelled are described by a discrete time MISO (multi-input/single-output) system of FREs [113]

$$\begin{aligned} Y(t) = {} & Y(t-\tau)\circ Y(t-\tau-1)\circ\cdots\circ Y(t-\tau-n_0) \\ & \circ U_1(t-\tau_1)\circ U_1(t-\tau_1-1)\circ\cdots\circ U_1(t-\tau_1-n_1) \\ & \vdots \\ & \circ U_m(t-\tau_m)\circ U_m(t-\tau_m-1)\circ\cdots\circ U_m(t-\tau_m-n_m)\circ R \end{aligned} \tag{7.1}$$

where $U_i(\cdot)$ and $Y(\cdot)$ for i=1,...,m are the fuzzy inputs and output of the equations; R is the fuzzy relation between them; t is the sampling instant; $\tau,\tau_1,\cdots,\tau_m$ are the time delays; and $n_0,n_1,\cdots,n_m$ are the system orders. The symbol "$\circ$" denotes one of the possible composition operators, e.g., max-min, max-t, or s-t [98]. Thus, structure identification is the determination of the time delays, the system orders and the composition operator while parameter identification is the resolution of a system of FREs. For simplicity, let

$$\begin{aligned} & X_1(t)=Y(t-\tau), \quad X_2(t)=Y(t-\tau-1), \quad \cdots, \; X_{n_0+1}(t)=Y(t-\tau-n_0) \\ & X_{n_0+2}(t)=U_1(t-\tau_1), \; X_{n_0+3}(t)=U_1(t-\tau_1-1), \quad \cdots, \; X_{n_0+n_1+2}(t)=U_1(t-\tau_1-n_1) \\ & \vdots \\ & X_{N-n_m}(t)=U_m(t-\tau_m), \; X_{N-n_m+1}(t)=U_m(t-\tau_m-1), \; \cdots, \; X_N(t)=U_m(t-\tau_m-n_m) \end{aligned} \tag{7.2}$$

where

$$N=\sum_{i=0}^{m}(n_i+1) \tag{7.3}$$

Hence, the system of FREs in eq.(7.1) can be rewritten as

$$Y(t)=X_1(t)\circ X_2(t)\circ\cdots\circ X_N(t)\circ R \tag{7.4}$$

and the fuzzy relation can be represented by $R = X_1 \times \cdots \times X_N \times Y$ where $\times$ denotes the Cartesian product. Note here that the input and output fuzzy sets have been denoted by $X_i(\cdot)$ and $Y(\cdot)$ respectively instead of the usual $A_i(\cdot)$ and $B(\cdot)$ in Chapter 6 to highlight their reference fuzzy set representations as discussed below. Let $\mathbf{U_i}$ and $\mathbf{V}$ be the universes of discourse of fuzzy inputs $X_i(\cdot)$ and output $Y(\cdot)$ respectively. Then the fuzzy relation R has $|\mathbf{V}|\cdot|\mathbf{U_1}|\cdot|\mathbf{U_2}|\cdots|\mathbf{U_N}|$ elements where $|\cdot|$ is the cardinality operator. This is a huge number when there are lots of inputs and the cardinality of the universes of discourse is large. The latter is common for discrete representation of fuzzy sets, see, e.g., Fig.7.1, which may involve tens or even hundreds of quantization levels. Therefore, Pedrycz proposed to use the reference fuzzy set representation [91]. Its underlying idea is essentially a fuzzification operation when the measured data are crisp, and is essentially a possibility measure [60] when they are in fuzzy form. Let $A_{ij}:\mathbf{U_i} \to [0,1]$ and $B_j:\mathbf{V} \to [0,1]$ be the jth reference fuzzy sets of the ith input and output variables respectively and assume that they satisfy the completeness conditions [118], i.e.

$$\forall u_i(t) \in \mathbf{U_i}, \exists j : A_{ij}(u_i(t)) > 0 \tag{7.5}$$

For the crisp data pair $(u_i(t), y(t))$, the reference fuzzy set representation is given by

$$X_i(t) = \{A_{ij}(u_i(t)) | \forall j\} \tag{7.6}$$

$$Y(t) = \{B_j(y(t)) | \forall j\} \tag{7.7}$$

of which eq.(7.6) has already depicted in Fig.7.2. On the other hand, the reference fuzzy set representations for fuzzy set pair (A_i, B) are

$$X_i(t) = \{Poss(A_i | A_{ij}) | \forall j\} = \left\{ \max_{u_i(t) \in \mathbf{U_i}} [A_i(u_i(t))\, t\, A_{ij}(u_i(t))] | \forall j \right\} \tag{7.8}$$

$$Y(t) = \{Poss(B | B_j) | \forall j\} = \left\{ \max_{y(t) \in \mathbf{V}} [B(y(t))\, t\, B_j(y(t))] | \forall j \right\} \tag{7.9}$$

whose cardinality, as in the crisp case, will be equal to the number of reference fuzzy sets employed in the corresponding variable space. Consequently, the memory requirement of the FRE system can be substantially reduced. However, as exemplified in Fig.7.2, the referenced fuzzy set (0,0,0.33,0.67,0,0,0) fails to be a normal fuzzy set. Furthermore, it will be highly-overlapped with the referenced fuzzy set of the left adjacent crisp input which is (0,0,0.67,0.33,0,0,0). As such, the existing analytical methods or even the approximate resolution methods together with those developed in Chapter 6 for discrete fuzzy set represented FRE system cannot be applied to dynamic system modelling applications and numerical techniques have been frequently adopted in order to obtain good approximate solutions.

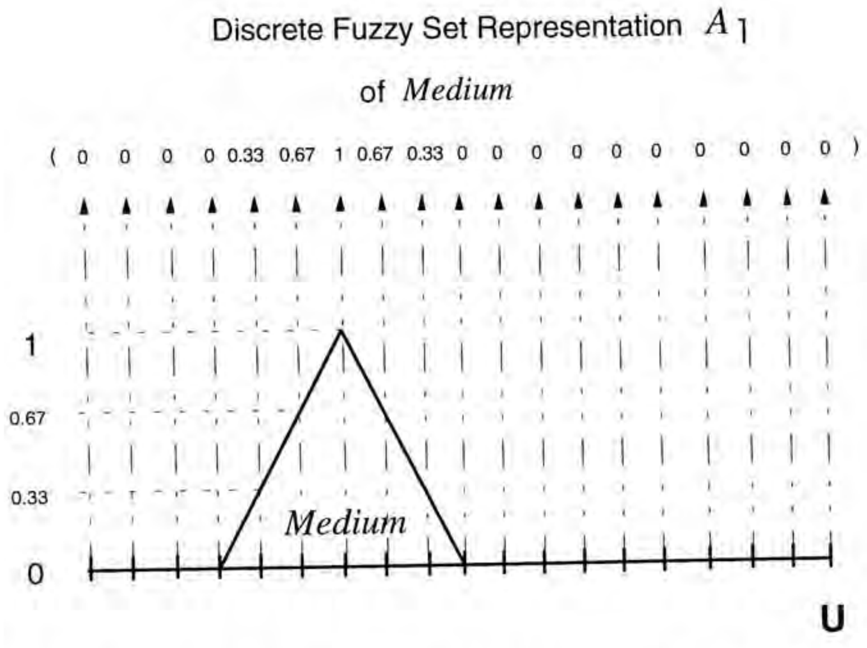


Figure 7.1 Discrete Fuzzy Set Representation of Concept *Medium*

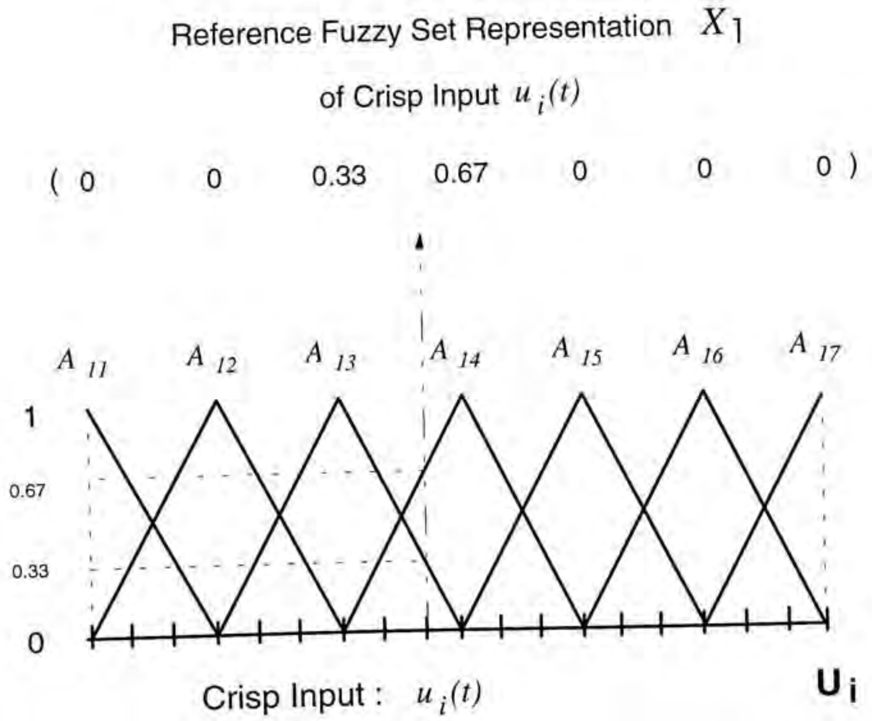


Figure 7.2 Reference Fuzzy Set Representation of Crisp input $u_i\ (t)$

Since crisp data can also be considered as singleton fuzzy sets, the reference fuzzy set representation in eqs.(7.8)&(7.9) provides a unified treatment of fuzzy and crisp data. A neural network representation of eq.(7.8) can be found in the referencing block of Fig.7.3 where the solid lines denote the *t*-norm operations on the input fuzzy set and the reference fuzzy sets A_{ij} and the open dots indicate the maximum operation on the incoming signals. Corresponding to Fig.7.1 & 7.2, the number of inputs to this referencing block is 19 while the number of outputs is 7. Fig.7.3 shows also the operation of a SISO (single-input/single-output) system. Upon receiving the referenced input fuzzy sets, the fuzzy relational neural system block will carry out the *s-t* inference operation, as denoted in the figure by solid lines and open dots with a 's' inside. After that, the referenced output fuzzy sets will be defuzzified and the crisp outputs are resulted. As mentioned in Chapter 2, there are lots of defuzzification strategies and the commonly used height defuzzification method, see eq.(2.40), is depicted in the rightmost block of Fig.7.3 where y_j^c is the nucleus of the *j*th output reference fuzzy set. Mathematically, the crisp output $y^*(t)$ is determined by

$$y^*(t)=\frac{\sum_j Y_j(t)y_j^c}{\sum_j Y_j(t)} \tag{7.10}$$

where $Y_j(t)$ is the *j*-th element of the reference fuzzy set representation $Y(t)$ in eq.(7.9). On the basis of Fig.7.3, Fig.7.4 depicts a MISO system which includes dynamic feedbacks from the outputs. Thus, the objective of identification is to minimize

$$\underset{\substack{R \\ \text{model structure} \\ \text{reference fuzzy sets} \\ \text{defuzzification scheme}}}{J} = \frac{1}{N_D-\tau_{\max}} \sum_{t=\tau_{\max}+1}^{N_D} \left(y(t)-y^*(t)\right)^2 \tag{7.11}$$

where $\tau_{\max}=\max(\tau,\tau_1,\cdots,\tau_m)$, $y(t)$ is the desired output and N_D is the total number of available data pairs.

In this part of the research, we have concentrated on the identifications of the fuzzy relation *R* and the model structure. The reference fuzzy sets and the defuzzification scheme are assumed to be fixed in advance. A new FRNS identification algorithm is proposed. Unlike the existing algorithms which fix the system structure beforehand and identify the fuzzy relations subsequently, the proposed one carries out both structure and parameter identifications in a concurrent manner. This can be achieved via the harness of evolutionary computation (EC) [40]. Before proceeding to describe the new algorithm, a general FRNS identification algorithm is described in the next section.

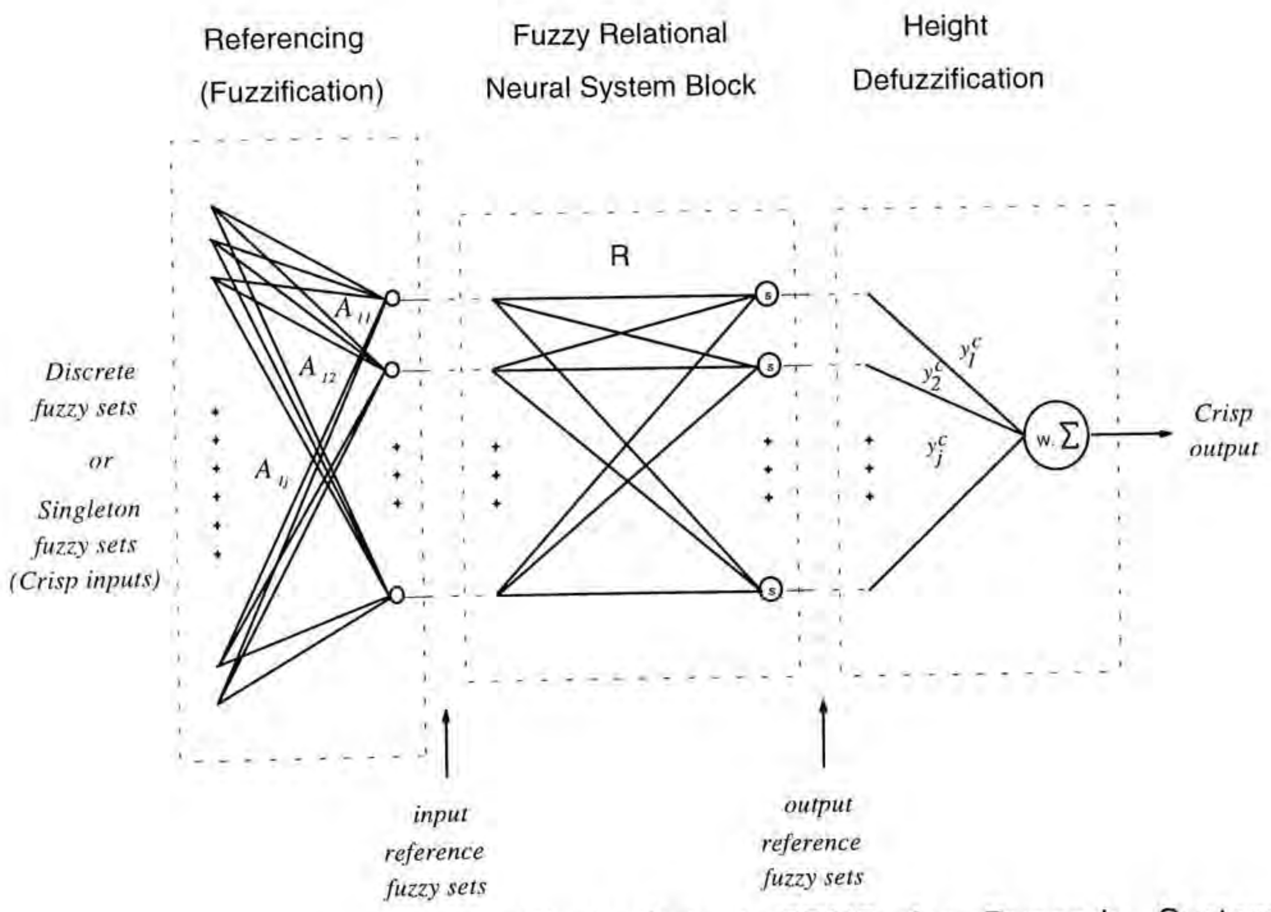


Figure 7.3 A Single-Input-Single-Output FRNS for Dynamic System Modelling

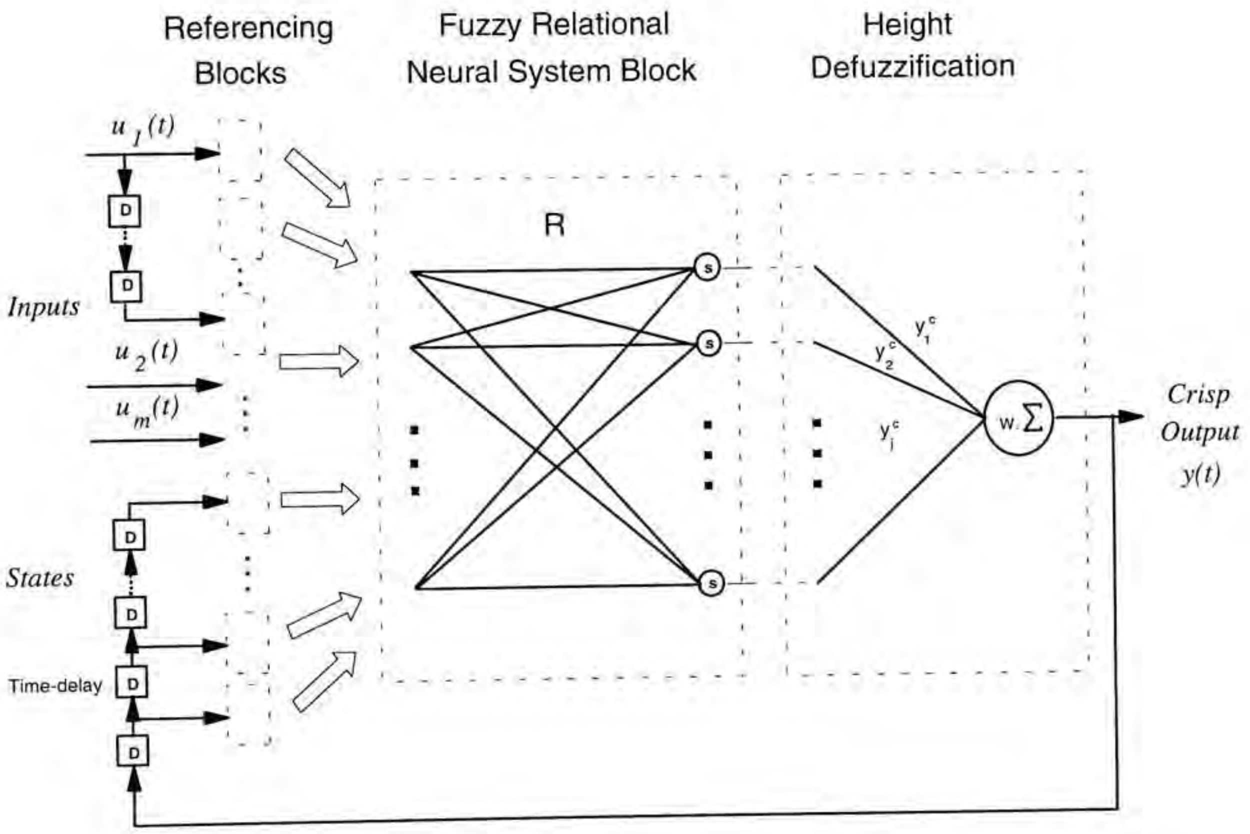


Figure 7.4 A Multiple-Input-Single-Output FRNS for Dynamic System Modelling

7.3 A General FRNS Identification Algorithm

According to [91], the FRNS identification can be described by the following six general steps.

A General FRNS Identification Algorithm

Step 1. *Data Acquisition*

- Collect the crisp and/or fuzzy input-output data.
- Define the universes of discourse involved.

Step 2. *Data Pre-processing*

- Perform clustering of data.
- Construct the reference fuzzy sets.
- Transform the raw data into reference fuzzy set representations.

Step 3. *Structure Identification*

- Identify/Fix the system structure, i.e.,
  - time delays,
  - composition operators,
  - system orders.

Step 4. *Parameter Identification*

- Compute the fuzzy relation R of the system such that the system error, defined by a given performance index, e.g., eq.(7.11), is minimized.

Step 5. *Testing*

- Test the system obtained against the collected data and check if it meets the requirement.

Step 6. *On-Line Identification (optional)*

- Update the system upon receiving new data.

Existing FRNS identification algorithm mainly worked on Steps 4 & 6. In general, gradient descent type optimization methods were employed in Step 4

[14,83,88,91,95] while a learn-and-forget principle was frequently adopted in Step 6 [12,113]. In [91], the fuzzy *c*-means clustering algorithm was proposed for Step 2. Neural clustering algorithms such as competitive learning and fuzzy competitive learning proposed in Chapter 4 can also be used and are particularly appropriate for on-line identification. Due to the lacking of feasible optimization tools, none has been done for identifying the system orders and time delays. However, identifying the composition operators by gradient descent has been proposed recently in [14] where the max-*t* composition is considered and the *t*-norm is defined by the Yager's class [114].

7.4 An Evolutionary Computation Approach to Structure and Parameter Identifications

In recent years, the interest in evolutionary computation (EC) has increased dramatically. It is usually considered as an effective optimization methodology that can be simulated on a computer, particularly, a parallel one, and applied to various real-world problems. Currently, there are three main avenues of research, namely, genetic algorithms, evolutionary strategies, and evolutionary programming. Each method has its root from different facet of natural evolution [41]. Basically, EC begins with a population of trial solutions brought to the task at hand. An objective function is used to assess the "fitness" of each trial solution, and a selection mechanism determines which solutions for subsequent generation. New solutions are created by altering the existing solutions with biologically inspired operations like cross-over and mutation. Therefore, EC algorithms can be generally described by a repeated sequence of evaluation, selection, and reproduction steps. Also, the principle of survival-of-the-fittest is exploited throughout the evolutionary process and as such EC can be considered as an intelligent random search method. It is superior to gradient descent techniques as the search is not biased toward the locally optimal solution.

7.4.1 Guided Evolutionary Simulated Annealing

Research works in EC are abundant for the past few years, see, e.g. [5,39,40]. One particular area of interest is the integration of EC with other local or global optimization techniques such that a more powerful search algorithm is devised. The guided evolutionary simulated annealing (GESA) method [116,117] is one of the most impressive examples. It combines the ideas of simulated evolution [38] and simulated annealing [59], and introduces a "regional guidance" mechanism by which the algorithm measures the qualities of multiple regions and focuses the search onto high quality regions automatically. Furthermore, it can be easily implemented on a massively parallel machine. In Fig.7.5 & 7.6, the GESA method is depicted. It can be seen that GESA mainly works on the selection step of EC. The search strategy, as a result of Steps 3.1-3.3 of Fig.7.5, depends not on the fitness of an individual, but on the fitness of a family of individuals which characterizes the quality of the region involved. Steps 3.2 & 3.3 have been designed to operate in a simulated annealing manner. It is due to Step 3.3, i.e., Fig.7.6, that the search is focused on high quality regions by allocating more individuals (family members) to those regions (family). For the quadratic assignment problems and the travelling salesman problems being solved in [116] and [117] respectively, simple random mutation was used for reproduction, though the performance is very encouraging. The use of recombination operators for reproduction has not been mentioned in [116,117]. In this regard, GESA can be classified as an evolutionary programming (EP) algorithm since EP stresses on behavioral change of species and there is no sexual communication between them [41].

Step 1 *Initialization*

- Set the initial temperatures t and form the initial population G(0) by randomly generate n parents and m children for each of them.

Step 2 *Evaluation*

- Evaluate the objective value (inverse fitness value) of the current population G(t).

Step 3.1 *Local Competition (Fittest Child Determination)*

- Find the fittest child for each family.

Step 3.2 *Parent Selection*

- For each family, the best child found in Step 3.1 is accepted as the new parent for next generation if

$$J_{C_{Best}} < J_P \qquad \text{or} \qquad \exp\left[-(J_{C_{Best}} - J_P)/t\right] > \rho$$

where $J_{C_{Best}}$ is the objective value of the best child,

J_P is the objective value of the parent,

t is the current temperature coefficient,

ρ is a random number uniformly distributed between 0 and 1.

Step 3.3 *Global Competition (Children Allocation)*

- Determine the number of children to be generated next in each family. The details of this step are given in Fig.7.6.

Step 4 *Reproduction*

- Form the next generation G(t+1) by generating children from the selected parents via mutation operation according to the number of children allocated to each family.

Step 5 *Annealing*

- Decrease the temperature parameters for Step 3 according to the cooling schedule.

Step 6 *Termination*

- Repeat Step 2 to Step 5 until an acceptable solution has been found or the maximum number of generation has been reached.

Figure 7.5 The Guided Evolutionary Simulated Annealing (GESA) Method

```
Step 1   Repeat Step 2 to Step 3 for each family, goto Step 4.

Step 2   Set the acceptance number of the ith family $A_i = 0$.

Step 3   For each child, increase $A_i$ by 1 if

             $J_C < J_{Lowest}$     or     $\exp[-(J_C - J_{Lowest})/t] > \rho$

           where $J_C$          is the objective value of the child,
                 $J_{Lowest}$   is the lowest objective value ever found,
                 $t$            is the current temperature coefficient,
                 $\rho$         is a random number uniformly distributed between 0 and 1.

Step 4   Sum up the acceptance numbers of all the families, i.e., $S = \sum_{i=1}^{n} A_i$

Step 5   For each family, the number of children to be generated in the next
         generation is determined by the following formula

             $m_i = T \cdot \frac{A_i}{S}$

         where $m_i$ is the number of children that will be generated for the ith
           family
                 $T = \sum_{i=1}^{n} m_i$ is the total number of children, i.e., $n \times m$.
```

Figure 7.6 Algorithm for the Children Allocation Step of Fig.7.5

7.4.2 An Evolutionary Identification (EVIDENT) Algorithm

Based upon the GESA algorithm, an evolutionary identification algorithm called EVIDENT is proposed. The GESA algorithm was adopted because i) evolutionary programming is preferable to genetic algorithms when there is no sufficient calculus to guide recombinations [3], as in our case where the relationship between fuzzy relations for different time delays and inference operators is still unclear, ii) the representation of the time delays, composition operators, and fuzzy relations are easier and more straight-forward, and iii) it is a powerful method that can be implemented in parallel. Since the system orders are related to the generalization property of FRNS and their identification is too involved to be included in current study, they are assumed to be fixed in advance.

The EVIDENT algorithm addresses both the structure identification and parameter identification steps of the general identification algorithm described in Section 7.3 using the GESA method. The fitness function is simply the mean squared error J defined in eq.(7.11). The search space is the set of all valid representations of the time delays, composition operators, and fuzzy relations. For the time delays, a $(m+1)$-dimensional vector, corresponding to the FRE system in eq. (7.1),

$$\bar{\tau} = (\tau, \tau_1, \cdots, \tau_m) \tag{7.12}$$

where $\tau \in (1, \tau_{\max})$ and $\tau_i \in (0, \tau_{\max})$ are used. $\tau_{\max}$, as in eq.(7.11), is the maximum time delay allowed. For the composition operators, the representation depends on the class of operators adopted. In this work, the *s*-*t* composition, due to its generality, is used and the *t*-norm and *s*-norm are defined as the Dubois type [35]

$$p t_D q = \frac{pq}{p \vee q \vee g_t} \tag{7.13}$$

and

$$p s_D q = 1 - \frac{(1-p)(1-q)}{(1-p) \vee (1-q) \vee g_s} \tag{7.14}$$

where $0 \le g_t, g_s \le 1$ respectively. When g_t, g_s=0, t_D will be the minimum operator and s_D will be the maximum operator. On the other hand, when g_t, g_s=1, they will become the algebraic product ' $\cdot$ ' in eq.(2.13) and algebraic sum '$\dot{+}$' in eq.(2.18) respectively. It has been found that the relationships

$$\cdot \le t_D \le \wedge \tag{7.15}$$

and

$$\vee \le s_D \le \dot{+} \tag{7.16}$$

hold when $0 \le g_t, g_s \le 1$ [80]. Although the Dubois type of *t*-norm and *s*-norm is not the most general one [80], it is simple and computationally fast. The representation for the composition operators is simply a 2-D vector, i.e.,

$$\vec{g} = \{g_t, g_s\} \tag{7.17}$$

where $g_t, g_s \in [0,1]$. For the fuzzy relation, its (*N*+1)-dimensional matrix form, recalled from eq.(7.4), can be represented by the vector

$$\vec{r} = (r_1, r_2, \cdots, r_M) \tag{7.18}$$

where $r_i \in [0,1]$ and $M = |\mathrm{Y}| \cdot |\mathrm{X}_1| \cdot |\mathrm{X}_2| \cdots |\mathrm{X}_k|$, according to Section 7.2. Therefore, each valid assignment of the time delays, composition operators, and fuzzy relation is represented by a (*m*+3+*M*)-dimensional vector :

$$\vec{c} = (\tau, \tau_1, \cdots, \tau_m, g_t, g_s, r_1, r_2, \cdots, r_M) \tag{7.19}$$

such that $\tau \in (1, \tau_{\max})$, $\tau_i \in (0, \tau_{\max})$, $g_t, g_s \in [0,1]$, $r_i \in [0,1]$.

In the wordings of genetic algorithms, mutation is the occasional alteration of some gene values in a chromosome. Therefore, each element of $\vec{c}$ is chosen for mutation with a mutation probability. Such probability should be kept at a low value to maintain the regional guidance characteristic. The mutation can take place either by replacing the selected element with a random number in the range or by applying a reasonable perturbation to it. The latter is adopted here. In fact, this scheme has also been adopted by [87] for a similar application.

As an additional mechanism to the GESA, the elitist selection [45] has been incorporated by the EVIDENT algorithm although it has not been included in the original implementation by Yip & Pao [116,117]. It simply always retains the solution with the best fitness in every generation. Since the best solution may not necessary become the parent due to the simulated annealing effect in the parent selection process, we may just give away the true solution and the simulated annealing process would just make the system unstable. To avoid this, when the best solution has not been selected as the parent, it is retained to be a child in the family. Such mechanism is frequently used by genetic algorithms and is found to be essential to guarantee the asymptotic convergence of the algorithms [100] and indeed gives better results to our experiments.

7.5 Simulation Results

In this section, the performances of the proposed algorithm in three data sets, namely, the gas-furnace data from Box-Jenkins [8], a simulated dynamic model data set [73,113], and the chaotic time series [76], are reported. Unless otherwise stated, the parameter settings of the EVIDENT algorithm in Table 7.1 were adopted. The basic principle for setting the initial temperatures is that the simulated annealing process should have a conditional probability of acceptance around 98% initially. This will encourage more random walks at the beginning of the training process. The cooling factor is the same as that adopted by Yip and Pao [116,117]. The mutation ranges were selected to cover about half of the full range. The initializations for time delays and composition operators are totally random. For the fuzzy relations, the linguistic initialization scheme commonly adopted by existing parameter identification algorithms [73,113] was adopted. For example, the element of the initial fuzzy relation for max-min composition is calculated as

$$R = \max_{t=\tau_{\max}+1}^{N_D} R(t) = \max_{t=\tau_{\max}+1}^{N_D} \left\{ \min[X_1(t), X_2(t), \cdots, X_N(t), Y(t)] \right\} \tag{7.20}$$

The aggregation operations change with respect to the composition adopted. As remarked in Table I, two random initialization schemes for the fuzzy relation were also used for testing in the gas-furnace data set. One is totally random in [0,1] and the other is biased random around 0.5. The latter allows the search process starting at the central area of the search space.

Table 7.1 Parameter Settings of the Evolutionary Identification Algorithm

	Gas-Furnace Data Set	Simu. Dynamic Model	Chaotic Time Series
Population Size	50 (10×1 parents, 10×4 children)	80 (10×1 parents, 10×7 children)	100 (10×1 parents, 10×9 children)
Maximum No. of Generations	1000	1000	1000
Cooling Schedule :			
- Initial temp. (parent selection)	50	0.1	10
- Init. temp. (global competition)	2000	4	40
- Cooling factor	0.99	0.99	0.99
Mutation Probability	0.05†	0.05	0.05
Mutation Range :			
- Time delays	±2	±2	±5
- Composition operators	±0.2	±0.2	±0.2
- Fuzzy relation	±0.2	±0.2	±0.2
$T_{\max}$	10	10	20
Initialization :			
- Time delays	random	random	random
- Composition operators	random	random	random
- Fuzzy relation	linguistic‡	linguistic	linguistic

† Mutation probability=0.01, 0.1, 0.2 have also been used for sensitivity analysis

‡ Random in [0,1] and random in [0.4,0.6] have also been tested in the gas-furnace data set

Experiment #1 ***: Box-Jenkins' Gas-Furnace Data Set*** [8]

The gas furnace data set is a well-known problem for benchmarking FRNS/FRE system identification algorithms. It consists of 296 input-output data pairs where the input $u_1(t)$ is the input gas flow rate and the output $y(t)$ is the concentration of CO_2 in outlet gas. The sampling interval is 9 seconds. In order to benchmark against the best performance to date, the reference fuzzy sets adopted are the same as those in [73] and have been depicted in Fig.7.7. Furthermore, the model is represented by

$$Y(t) = X_1(t) \circ X_2(t) \circ R \tag{7.21}$$

where $X_1(t)=Y(t-\tau)$ and $X_2(t)=U_1(t-\tau_1)$, and its objective function is defined as

$$J = \frac{1}{286} \sum_{t=11}^{296} \left(y(t) - y^*(t)\right)^2 \tag{7.22}$$

In Table 7.2, a comparison of the EVIDENT algorithm using linguistic initialization and three existing algorithms is made. The performance of the proposed algorithm has been recorded as an average of 10 trials using different initializations[1] while those of three existing algorithms are quoted from the cited references. Among the four sets of simulations for the EVIDENT algorithm, the first two correspond to the inhibition of the structure identification capability so that a direct comparison of the parameter identification performance is allowed. Obviously, the performance of EVIDENT is much better than that of the existing algorithms. In the third case, the result shows that the proposed algorithm can identify the optimal time delays (1,4) and the algebraic-sum-product composition in 9 out of the 10 trials. As indicated by the superior performance of the fourth set of simulations which has the composition operator and time delays fixed at algebraic-sum-product and (1,4) respectively, the composition operator identified is also an optimal one. A typical identified system

[1] Since the linguistic initialization is deterministic for the simulations with fixed time delays, small perturbations have been applied to the computed fuzzy relations to obtain ten different initializations.

outputs of the EVIDENT algorithm have been plotted in Fig.7.8. They are almost indistinguishable from the original outputs.

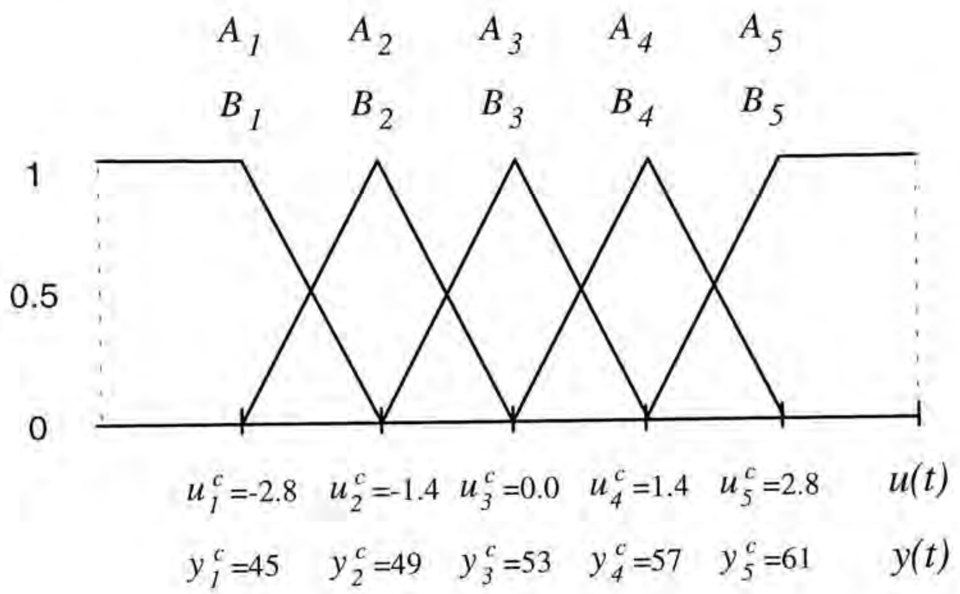

A_1 A_2 A_3 A_4 A_5
B_1 B_2 B_3 B_4 B_5
1
0.5
0
u_1^c=-2.8 u_2^c=-1.4 u_3^c=0.0 u_4^c=1.4 u_5^c=2.8 $u(t)$
y_1^c=45 y_2^c=49 y_3^c=53 y_4^c=57 y_5^c=61 $y(t)$

Figure 7.7 The Reference Fuzzy Sets Used : Box-Jenkins' Gas-Furnace Data Set

Table 7.2 Comparison of Performance : Gas-Furnace Data Set

Algorithm	Composition	Time Delays (τ, τ_1)	System Error J
Pedrycz [91]	max-product	(1,3)	0.776
Xu & Lu [113]	max-product	(1,4)	0.328
Lee *et al.* [73]	max-product	(1,3)	0.211
EVIDENT	fixed at max-product	fixed at (1,3)	0.167
	fixed at max-product	fixed at (1,4)	0.156
	s-t†	(τ, τ_1)‡	0.129
	fixed at alg.sum-product	fixed at (1,4)	0.124

† Algebraic Sum-Product was identified in 9 (out of 10) trials.

‡ (1,4) was identified in 9 (out of 10) trials.

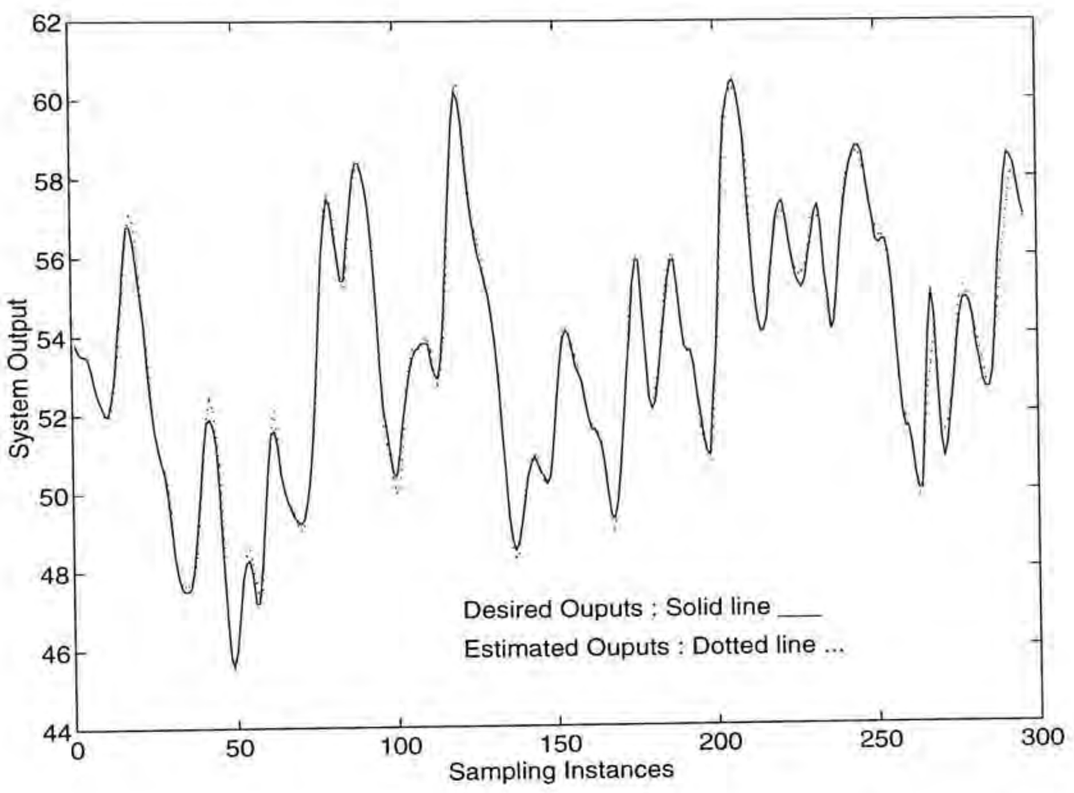


Figure 7.8 Actual and Estimated Outputs Obtained by EVIDENT : Box-Jenkins' Gas-Furnace Data Set

In Fig.7.9, the fitness history (learning curve) of the best individual in an EVIDENT algorithm simulation carrying out both structure and parameter identifications is depicted. It can be seen that the system error is lower than the best result todate which is 0.211, after approximately 200 generations of training and the algorithm converged in about 400 generations. Such a computational requirement is quite common in EC algorithms. Sensitivity analysis of the proposed algorithm has been conducted. In Table 7.3, the sensitivity to the initialization scheme of the fuzzy relations in various sets of simulations is depicted. As expected, the linguistic scheme is the best because it can provide a good initial solution for the EVIDENT algorithm to optimize. Furthermore, it is more robust as compared with the other two schemes, in particular, the unbiased random initialization (in [0,1]) where the system error variance of the solutions obtained is as high as 0.312. This matches with the results reported in [87] which shows that genetic-based algorithms despite their goal for obtaining globally optimal solutions are also sensitive to the initial conditions. Regarding the sensitivity to the mutation rate, Table 7.4 shows that the proposed algorithm is quite robust to a range of values, i.e., 0.01 to 0.1. The performance was degraded significantly when larger mutation rate was adopted. In fact, using small enough mutation rate is essential to the preservation of the regional guidance property of the GESA method; otherwise, it will become a pure random search algorithm. As a result of this analysis, mutation rate in [0.01,0.1] is suggested for further applications of EVIDENT.

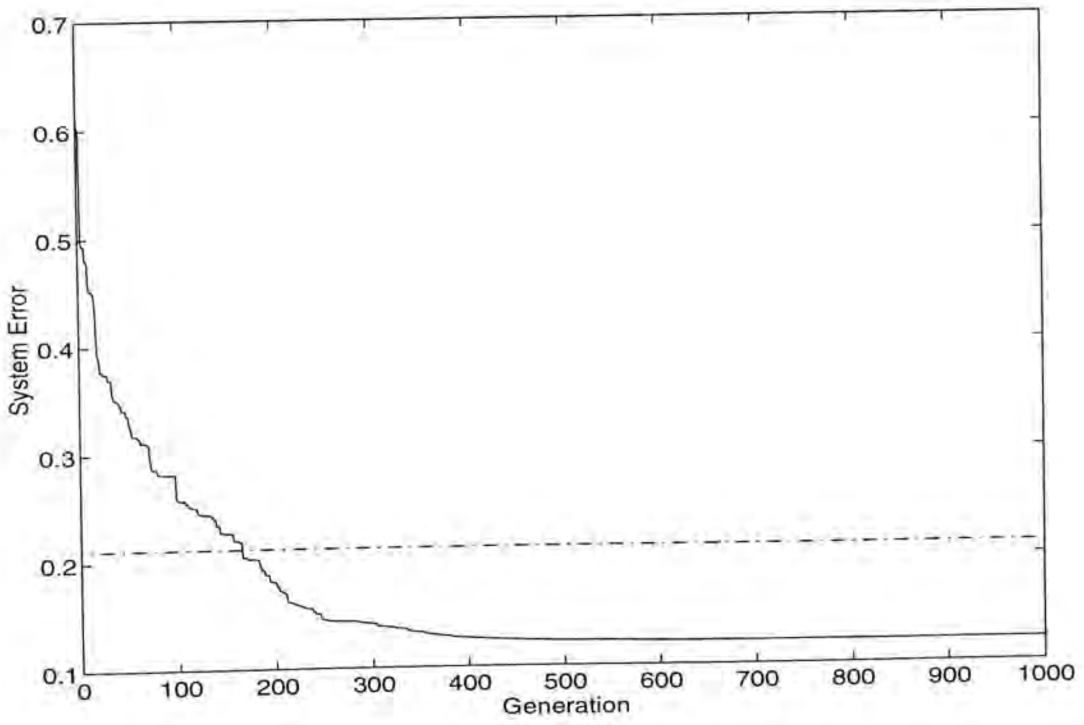


Figure 7.9 Fitness History of the Best Individual in an EVIDENT algorithm simulation : Box-Jenkins' Gas-Furnace Data Set

Table 7.3 Sensitivity Analysis of Initialization Schemes : Gas-Furnace Data Set

Initialization	System Error J		
	Sim#1	Sim#2	Sim#3
	max-product, (1,4)	alg.sum-product, (1,4)	*s-t*, (τ, τ_1)
Linguistic	0.156	0.124	0.129 (0.007)†
Random in [0,1]	0.245	0.129	0.437 (0.312)
Biased Random	0.242	0.127	0.219 (0.118)

† The bracketed values show the variances

Table 7.4 Sensitivity Analysis of Mutation Probability : Gas-Furnace Data Set

Mutation	Sim#1	Sim#2	Sim#3
Probability	max-product, (1,4)	alg.sum-product, (1,4)	*s-t*, (τ, τ_1)
0.01	0.161	0.125	0.122
0.05	0.156	0.124	0.129
0.1	0.176	0.127	0.126
0.2	0.246	0.146	0.151

Experiment #2 *: A Simulated Dynamic Model*

The two-input/single-output dynamic model [73,113]

$$y(t) = 0.8y(t-1)u_1(t) + 0.5u_1(t-1)y(t-2) + u_2(t-4) \tag{7.23}$$

is another popular benchmarking data set for FRNS identification. A set of 400 output data is generated from eq.(7.23) by letting inputs $u_1(t)$ and $u_2(t)$ be uncorrelated random sequences uniformly distributed on (0.1,0.9). The model is described by

$$Y(t) = X_1(t) \circ X_2(t) \circ X_3(t) \circ R \tag{7.24}$$

where $X_1(t)=Y(t-\tau)$, $X_2(t)=U_1(t-\tau_1)$ and $X_3(t) = U_2(t-\tau_2)$, and its objective function is defined as

$$J = \frac{1}{390}\sum_{t=11}^{400}\left(y(t) - y^*(t)\right)^2 \tag{7.25}$$

Again, we have adopted Lee *et al.*'s reference fuzzy sets. Their distributions are the same as those in Fig.7.7 and the nucleuses for inputs $u_1(t)$ and $u_2(t)$ are $u_{11}^c, u_{21}^c = 0.1$, $u_{12}^c, u_{22}^c = 0.3$, $u_{13}^c, u_{23}^c = 0.5$, $u_{14}^c, u_{24}^c = 0.7$, and $u_{15}^c, u_{25}^c = 0.9$, and those for the output $y(t)$ are $y_1^c = 0.3$, $y_2^c = 1.1$, $y_3^c = 1.9$, $y_4^c = 2.7$, and $y_5^c = 3.5$. In Table 7.5, the performances of EVIDENT and the other two algorithms are compared. As in Table 7.2, each reading of EVIDENT corresponds to the average of 10 runs and those of the other two algorithms are quoted from the cited references. Once again in all cases, EVIDENT has outperformed the Xu & Lu's algorithm and the Lee *et al.*'s algorithm, which is the best FRNS parameter identification algorithm reported todate. It has identified the optimal structure in all the 10 trials. It can be seen from the third set of simulations for EVIDENT that the best performance is obtained when the structure is fixed at the optimal one which has been determined by the earlier attempts. This is reasonable because EVIDENT, in that case, can fully utilize its resources to search the parameter space and hence can explore more regions and attain better parameter identification.

Table 7.5 Comparison of Performance : Simulated Dynamic Model

Algorithm	Composition	Time Delays (τ,τ_1,τ_2)	System Error J
Xu & Lu [113]	max-product	(1,0,4)	0.0230
Lee *et al.* [73]	max-product	(1,0,4)	0.0186
EVIDENT	fixed at max-product	fixed at (1,0,4)	0.0130
	s-t†	(τ,τ_1,τ_2)‡	0.0118
	fixed at alg.sum-product	fixed at (1,0,4)	0.0106

† algebraic-sum-product composition has been identified in all trials

‡ time-delays (1,0,4) has been identified in all trials

Experiment #3 *: Chaotic Time Series Data Set*

The chaotic time series has long been used to benchmark neural network learning algorithms. In this experiment, the time series described by the Mackey-Glass differential equations [76]

$$\frac{dy(t)}{dt} = \frac{0.2y(t-17)}{1+y^{10}(t-17)} - 0.1y(t) \tag{7.26}$$

was adopted. The equation was integrated using the fourth-order Runge-Kutta method to provide values of *y(t)* at discrete time steps. From the computed values, we extracted 1000 points of which the first 500 points were used for training and the rest 500 points were used for prediction. For neural networks, the task is usually formulated as predicting $y(t+6)$ from four past data points $y(t)$, $y(t\text{-}6)$, $y(t\text{-}12)$, $y(t\text{-}18)$. Taking the advantages of the structure identification capability of EVIDENT, we have reformulated the problem as predicting $y(t+6)$ from four past data points $y(t-\tau)$, $y(t-\tau_1), y(t-\tau_2)$, $y(t-\tau_3)$ whose time delays are to be identified. As such, the following model was employed

$$Y(t+6) = X_1(t) \circ X_2(t) \circ X_3(t) \circ X_4(t) \circ R \tag{7.27}$$

where $X_1(t)=Y(t-\tau)$, $X_2(t)=Y(t-\tau_1)$, $X_3(t)=Y(t-\tau_2)$, & $X_4(t)=Y(t-\tau_3)$, and its objective function is defined as

$$J = \frac{1}{474}\sum_{t=27}^{500}\left(y(t)-y^*(t)\right)^2 \tag{7.28}$$

According to eq.(7.27), the fuzzy relation is a 5-dimensional one and will consist of $5^5 = 3125$ elements if five reference fuzzy sets are adopted. In order to reduce that number, only three reference fuzzy sets, corresponding to only $3^5 = 243$ matrix elements, were used and they are depicted in Fig.7.10. The prediction results have been recorded in Table 7.6 where EVIDENT is benchmarked against four other methods that are not based on FRNS. The first three rows of results are from [27] and they correspond to the auto-regressive (AR) model, the cascade-correlation learning neural network [37], and the back-propagation neural network respectively. The fourth row is the result of Jang's ANFIS model [54] introduced in Chapter 3. The prediction performance was measured by the non-dimensional error index (NDEI) which is defined as the root mean squared error divided by the standard deviation of the target series [27]. It can be seen that EVIDENT is superior to the AR, cascade-correlation learning, and BP models among which the latest one is well-known for its generalization performance and has long been used to predict (chaotic) time series and identify nonlinear dynamic systems. By examining the structure obtained, EVIDENT has identified the algebraic-sum-product composition operator in all the ten trials. However, it has identified a variety of time delays, including 0, 1, 3, 12, 13, 15, 16, 17, 18, & 20, and the most popular ones are 0, 13, 15, & 20. Hence, we have tried to simulate EVIDENT with the four time delays fixed at these values and the composition operator fixed at algebraic-sum-product. As shown in Table 7.6, the result is better. We further tried to check the optimality of the set of time delays (0, 6, 12, 18) commonly adopted by neural network models and the result demonstrates that it is not comparable to that of (0, 13, 15, 20) as $(\tau_1, \tau_2, \tau_3, \tau_4)$. On the other hand, it can be seen that EVIDENT is not so good as Jang's ANFIS model because the ANFIS model optimizes the shape and location of the membership functions (reference fuzzy sets) while EVIDENT in its current form predefines the input reference fuzzy sets to be evenly distributed, semi-overlapped triangular as shown in Fig.7.7 and Fig.7.10. Such difference has indeed been demonstrated to be influential to the performance of fuzzy systems [75]. Since Jang's model is based on Takagi & Sugeno's fuzzy system and is therefore different from FRNS in both structure and inference mechanism. One has to carry out more experiments on these two models in order to determine which one is better and this has been left for further research.

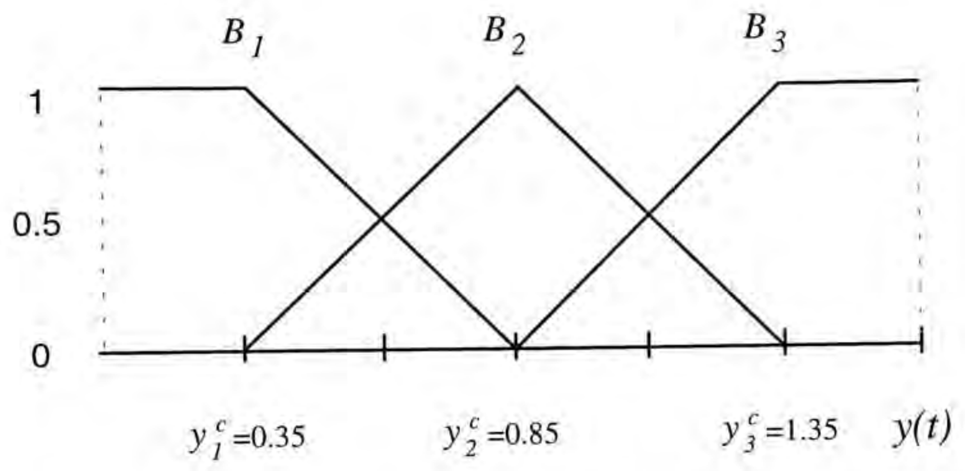


Figure 7.10 The Reference Fuzzy Sets Used : Chaotic Time Series

Table 7.6 Comparison of Prediction Performance : Chaotic Time Series

Algorithm	Non-Dimensional Error Index
AR Model	0.19
Cascaded-Correlation NN	0.06
Back-Propagation NN	0.02
ANFIS [54]	0.007
EVIDENT :	
- *s-t* and $(\tau_1, \tau_2, \tau_3, \tau_4)$	0.0105
- fixed at alg.sum-prod., (0,13,15,20)	0.0089
- fixed at alg.sum-prod., (0,6,12,18)	0.0115

7.6 Concluding Remarks

In this chapter, we have addressed an important but not-yet-explored learning problem in the fuzzy relational neural system (FRNS) model, i.e., the identification of both model structure and system parameters. An evolutionary identification algorithm called EVIDENT based on the GESA method has been developed. As demonstrated by the simulation results of various benchmarking data sets, EVIDENT not only is effective in determining a near optimal structure of the systems, but is also capable for identifying a better parametric solution than the best FRNS identification algorithms reported todate. With such algorithm, the uncertain trail-and-error process normally required by existing FRNS identification algorithms [73,91,113] for structure determination can be eliminated. Thus, the new algorithm should be a very attractive method for applications whose underlying system dynamic is unknown. For example in financial forecast, EVIDENT would be distinctive in determining which type of past data is critical to the prediction tasks.

Chapter 8

Conclusions

In attempt to identify methodologies to integrate fuzzy systems with neural networks technology, three areas have been chosen to study in more details after careful survey of the development of fuzzy neural systems todate. Through a common strategy to synthesize fuzzy neural systems, viz. fuzzifying a learning algorithm for a neural system, a new class of FNS model has been developed. Since another strategy to synthesize fuzzy neural system is to implement fuzzy system in neural network architecture, attention is then given to capacity analysis and numerical identification of such systems. While the results have not been applied to solve practical problems, they have all been demonstrated to be effective for various benchmarking experiments. Some of these works have already been published and received considerable attention. They and some related publications are listed in Appendix A for reference. In the following, the contributions made by this work towards the synthesis of fuzzy neural systems is summarized and directions for further research is suggested.

8.1 Summary of Contribution

8.1.1 Fuzzy Competitive Learning

A new class of FNS model has been proposed. Methodology to fuzzify any CL algorithm that conforms the general description given in Section 4.2 has been developed and it has been presented in Section 4.3. According to this methodology, three existing CL algorithms covering unsupervised and supervised types have been fuzzified. Compared with their crisp counterparts, the newly developed fuzzified models were found to be distinctive in i) converging more often to the desired solutions, or equivalently, reducing the likelihood of neuron underutilization; ii) obtaining superior classification and generalization performance, especially in cases of overlapping data sets; and iii) allowing interpretation of trained networks. Thus, the class of FCL networks is a very attractive model for applications such as image coding, pattern classification, and membership function estimation.

In addition, the work of fuzzy competitive learning has laid down an example of how to fuzzify neural network models. It takes use of the strength of fuzzy sets in modelling a concept employed by CL that is fuzzy or vague in nature. One may synthesize a new FNS model along this way by first identifying the crisp concepts in the neural networks that are inefficient and potentially be better modelled by fuzzy sets, e.g., the class of training patterns, and then fuzzifying them using fuzzy theories and methodologies.

8.1.2 Capacity Analysis of FAM and FRNS

Storage capacity of FAM has been theoretical studied. A perfect recall theorem for multiple storage of FAM has been developed and the condition to allow multiple storage by a FAM matrix has been identified. Moreover, the capacity of a FAM matrix has been enhanced by generalizing the usual max-min composition to the max-

$\otimes$ one to achieve single FAM matrix implementation of a set of semi-overlapped fuzzy rules, which is typical in general applications.

By re-examining the solvability condition of the FREs, the new one has been identified. Based on the new result, boundary conditions for several typical types of FREs have been derived. Also, based upon the new results, capacities of some systems have been derived in Section 6.6. Since FRNS is functionally equivalent to FREs, the results are applicable to FRNS and instructive to the design and implementation of fuzzy rulebases, i.e., if the input fuzzy sets are designed to be normal and semi-overlapped, the rulebase can be merely implemented by the FRNS model.

This study has dealt with efficient network implementations of relation based fuzzy systems. It has not only consolidated the theoretical understanding on the storage property of the FAM and FRNS models but has also exemplified the benefit of borrowing neural network encoding principle to fuzzy systems.

8.1.3 Numerical Identification of FRNS

Adopting evolutionary computing approach, a numerical identification algorithm of FRNS called EVIDENT that identifies both its structure and parameters concurrently has been developed. The algorithm can be applied to identify nonlinear dynamic system without going through a trail-and-error process to determine the system structure. If the optimal structure is known in advance, the newly developed algorithm is still capable of obtaining better parameter estimation. Comparative study has been made to evaluate the performance of the developed algorithm in nonlinear dynamic system identification against some well-known algorithms using various benchmarking data sets.

8.2 Further Investigations

In the derivation of the three FCL algorithms, we have assumed the network size be fixed in advance. An appropriate choice however requires a good understanding of the data set being learnt. In practice, this knowledge may not be available and the network has to be designed by an intelligent guess approach. Such a network design problem has long been an active research problem in neural networks [15,20,116] and many effective methodologies such as network growth [20] and evolutionary computation [116] have been developed. Application of these methods to FCL networks, however is non-trivial.

Confidence measure is a special feature of fuzzy pattern recognition algorithms [56]. Since the output neurons of the FCL networks have been formulated as fuzzy class memberships of input patterns, one may exploit such valuable and ready-to-use information to measure the confidence of the classification results. For example, if the highest and the second highest class memberships are very close to each other, we may say that the trained network is not sure about its classification. Such kind of knowledge is very useful in applications like signature verification where the users may require to sign again if the network is not confident in its classification.

The experimental results of FCL networks indicate that their performances are quite sensitive to the fuzziness parameter used. We have then conjectured that the fuzziness value is related to the characteristics of the training data set, in particular, the degree of overlapping. As we mentioned in Section 4.5, all the membership values will become $1/c$ when the fuzziness tends to infinity which is the fuzziest condition. According to the learning law in eq.(4.16), this will cause all the competing neurons converged to the centroid of the training data and such a phenomenon indeed reflects that the fuzzy model considers the data clusters being totally overlapped with each other. As the fuzziness decreases, the competing neurons will deviate from the centroid accordingly, indicating the assumption of lesser degree of overlapping. In the extreme case of m=1, the model will perform hard clustering, i.e., non-overlapping data clusters are being considered. Thus, it looks promising to carry out further research in this direction to derive the relationship between the fuzziness parameter and the degree of overlapping in the training data set, and to obtain a better understanding on data clustering.

The design of the EVIDENT algorithm has been focused on the identification of system time delays, the inference operator, and the relation parameters. As the final goal of using FRNS to model nonlinear dynamic systems is to identify all the system structure and parameters, including the reference fuzzy sets, defuzzification scheme, and system orders, in an autonomous manner, incorporating them into EVIDENT will be pursued in our further studies. This requires to search for a generalized representation of each of them, to formulate it under the framework of EVIDENT, and to devise the corresponding reproduction schemes. Since the system orders determine the accuracy of FRNS in modelling and consequently are influential to its generalization performance, the objective function (see eq.7.11)) for fitness evaluation may have to be modified to take that into consideration. The enhanced algorithm will then be very useful to analyze the importance of the reference fuzzy sets, defuzzification scheme, and system orders to the performance of fuzzy system modelling.

Appendix A

Publication List of the Candidate

I. Publications Reporting the Work in this Thesis

1. F.L. Chung and T. Lee, "A fuzzy competitive learning algorithm with decreasing fuzziness," *Electronics Letters*, vol.29, no.13, pp.1206-1208, 1993.
2. F.L. Chung and T. Lee, "Fuzzy learning vector quantization," *Proc. Int'l Joint Conference on Neural Networks (IJCNN '93)*, Nagoya, Japan, pp.2739-2742, 1993.
3. F.L. Chung and T. Lee, "A fuzzy learning model for membership function estimation and pattern classification," *Proc. IEEE International Conference on Fuzzy Systems (FUZZ-IEEE'94)*, Orlando, Florida, pp.426-431, 1994.
4. F.L. Chung and T. Lee, "Fuzzy competitive learning," *Neural Networks*, vol.7, no.3, pp.539-552, 1994.
5. F.L. Chung and T. Lee, "Two methods for perfect encoding of a set of semi-overlapped fuzzy rules," to appear in *Proc. 6th Int. Fuzzy Systems Association World Congress (IFSA '95)*, Sao Paulo, Brazil, 1995.
6. F.L. Chung and T. Lee, "EVIDENT : An evolutionary identification algorithm for fuzzy relational systems," to appear in *Proc. World Congress on Neural Network (WCNN '95)*, Washington, DC, 1995.
7. F.L. Chung and T. Lee, "On fuzzy associative memory with multiple-rule storage capacity," submitted to *IEEE Trans. on Fuzzy Systems*
8. F.L. Chung and T. Lee, "A new look at solving a system of fuzzy relational equations," submitted to *Fuzzy Sets and Systems.*

9. F.L. Chung and T. Lee, "Analytical resolution and numerical identification of fuzzy relational systems," submitted to *IEEE Trans. on System, Man & Cybernetics*.

II. Other Related Publications

10. F.L. Chung and T. Lee, "A node pruning algorithm for backpropagation networks," *Neural Systems*, vol.3, no.3, pp.301-314, 1992.
11. F.L. Chung and T. Lee, "A network growth approach to the design of feedforward neural networks," to appear in *IEE Proceedings - Control Theory and Applications*.

III. Manuscript Under Preparation

12. F.L. Chung and T. Lee, "Signature verification using fuzzy learning vector quantization," under preparation.
13. F.L. Chung and T. Lee, "Incremental learning in fuzzy competitive learning," under preparation.

BIBLIOGRAPHY

[1] S.C. Ahalt, A.K. Krishnamurthy, P. Chen, & D.E. Melton, "Competitive learning algorithms for vector quantization," *Neural Networks*, vol.3, pp.277-290, 1990.

[2] E. Anderson, "The IRISes of the Gaspe Peninsula," *Bulletin of the American IRIS Society*, vol.59, pp.2-5, 1939.

[3] P.J. Angeline, G.M. Saunders, and J.B. Pollack, "An evolutionary algorithm that constructs recurrent neural networks," *IEEE Trans. on Neural Networks*, vol.5, no.1, pp.54-65, 1994.

[4] J.S. Baras and A. LaVigna, "Convergence of Kohonen's Learning Vector Quantization," *Proc. Int. Joint Conf. on Neural Networks (IJCNN-90-WASH-DC)*, Washington, DC, Vol.III, pp.17-20, 1990.

[5] R.K. Belew and L.B. Booker (Eds.), *Proceedings of the fourth international conference on genetic algorithms.* San Mateo, CA:Morgan Kaufmann, 1991.

[6] J.C. Bezdek, *Fuzzy mathematics in pattern classification.* Ph.D Dissertation, Cornell University, Ithaca, NY, 1973.

[7] J.C. Bezdek and S.K. Pal, *Fuzzy models for pattern recognition : Methods that search for structures in data.* New York:IEEE Press, 1992.

[8] G.E.P. Box and G.M. Jenkins, *Time series analysis, forcasting and control.* San Francisco, CA:Holden Day, 1970.

[9] J.J. Buckley and Y. Hayashi, "Fuzzy neural networks : A survey," *Fuzzy Sets and Systems*, vol.66, pp.1-13, 1994.

[10] P. Burrascano, "Learning vector quantization for the probabilistic neural network," *IEEE Trans. on Neural Networks*, vol.2, no.4, pp.458-460, 1991.

[11] G.A. Carpenter, S. Grossberg, & D.B. Rosen, "Fuzzy ART : Fast stable learning and categorization of analog patterns by an adaptive resonance system," *Neural Networks*, vol.4, pp.759-771, 1991.

[12] J.Q. Chen, H.H. Lu, and L.J. Chen, "An on-line identification algorithm for fuzzy systems," *Fuzzy Sets and Systems*, vol.64, pp.63-72, 1994.

[13] Y.S. Cheung, & K.P. Chan, "Modified fuzzy ISODATA for the classification of handwritten chinese characters," *Proceedings 1986 International Conference on Chinese Computing*, Singapore, pp.361-364, 1986.

[14] M.F. Chun and Z Bien, "Neurocomputational approach to solve a convexly combined fuzzy relational equation with generalized connectives," *Fuzzy Sets and Systems*, vol.57, pp.321-333, 1993.

[15] F.L. Chung and T. Lee, "A node pruning algorithm for backpropagation networks," *Neural Systems*, vol.3, no.3, pp.301-314, 1992.

[16] F.L. Chung and T. Lee, "A fuzzy competitive learning algorithm with decreasing fuzziness," *Electronics Letters*, vol.29, no.13, pp.1206-1208, 1993.

[17] F.L. Chung and T. Lee, "Fuzzy learning vector quantization," *Proc. Int'l Joint Conference on Neural Networks (IJCNN '93)*, Nagoya, Japan, pp.2739-2742, 1993.

[18] F.L. Chung and T. Lee, "A fuzzy learning model for membership function estimation and pattern classification," *Proc. IEEE International Conference on Fuzzy Systems (FUZZ-IEEE'94)*, Orlando, Florida, pp.426-431, 1994.

[19] F.L. Chung and T. Lee, "Fuzzy competitive learning," *Neural Networks*, vol.7, no.3, pp.539-552, 1994.

[20] F.L. Chung and T. Lee, "A network growth approach to the design of feedforward neural networks," to appear in *IEE Proceedings - Control Theory and Applications*.

[21] F.L. Chung and T. Lee, "Two methods for perfect encoding of a set of semi-overlapped fuzzy rules," to appear in *Proc. 6th Int. Fuzzy Systems Association World Congress (IFSA '95)*, Sao Paulo, Brazil, 1995.

[22] F.L. Chung and T. Lee, "On fuzzy associative memory with multiple-rule storage capacity," submitted to *IEEE Trans. on Fuzzy Systems*

[23] F.L. Chung and T. Lee, "EVIDENT : An evolutionary identification algorithm for fuzzy relational systems," to appear in *Proc. World Congress on Neural Network (WCNN '95)*, Washington, DC, 1995.

[24] F.L. Chung and T. Lee, "A new look at solving a system of fuzzy relational equations," submitted to *Fuzzy Sets and Systems.*

[25] F.L. Chung and T. Lee, "Analytical resolution and numerical identification of fuzzy relational systems," submitted to *IEEE Trans. on System, Man & Cybernetics.*

[26] P.J. Costa Branco and J.A. Dente, "A new algorithm for on-line relational identification of nonlinear dynamic systems," *Proc. IEEE Int. Conf. Fuzzy Systems*, San Francisco, vol.2, pp.1173-1178, 1993.

[27] R.S. Crowder, "Predicting the Mackey-Glass timeseries with cascade-correlation learning," *Proc. 1990 Connectionist Models Summer School*, D. Touretzky, G. Hinton, and T. Sejnowski, Eds., Carnegie Mellon University, pp.117-123, 1990.

[28] C. Dacaestecker, "Competitive clustering," *Proceedings International Neural Networks Conference*, Paris, vol.2, pp.833, 1990.

[29] D. DeSieno, "Adding a conscience to competitive learning," *Proceedings IEEE International Conference on Neural Networks*, pp.1117-1124, 1988.

[30] D.H. Deterding, *Speaker normalization for automatic speech recognition.* Ph.D. Dissertation, Cambridge University Engineering Department, Cambridge, UK, 1989.

[31] A. Di Nola, W. Pedrycz, S. Sessa, and P.Z. Wang, "Fuzzy relation equations under a class of triangular norms : A survey and new results," *Stochastica*, vol.2, pp.99-145, 1984

[32] A. Di Nola, S. Sessa, W. Pedrycz, and E. Sanchez, *Fuzzy relation equations and their applications to knowledge engineering.* Dordrecht:Kluwer Academic Publishers, 1989.

[33] A. Di Nola, S. Sessa, and W. Pedrycz, "A study on approximate reasoning mechanisms via fuzzy relation equations," *Approximate Reasoning* , vol.6, pp.33-44, 1992.

[34] D. Driankov, H. Hellendoorn, and M Reinfrank, *An introduction to fuzzy control.* Heidelberg:Springer-Verlag, 1993.

[35] D. Dubois and H. Prade, "A review of fuzzy sets aggregation connectives," *Inform. Sci.*, vol.36, pp.85-121, 1985.

[36] J.C. Dunn, "A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters," *Cybernetics*, vol.3, pp.32-57, 1974.

[37] S.E. Fahlman and C. Lebiere, "The cascade-correlation learning architecture," in *Advances in Neural Information Processing Systems II*, D.S. Touretzky, G. Hinton, and T. Sejnowski, Eds., San Mateo, CA:Morgan Kaufmann, pp.524-532, 1990.

[38] D.B. Fogel, "An evolutionary approach to the traveling salesman problem," *Biological Cybernetics*, vol.60, pp.139-144, 1988.

[39] D.B. Fogel and W. Atmar (Eds.), *Proceedings of the Second Annual Conference on Evolutionary Programming.* La Jolla, CA:Evolutionary Programming Society, 1993.

[40] D.B. Fogel and L.J. Fogel (Guest Eds.), *IEEE Trans. on Neural Networks (Special Issue on evolutionary computation)*, vol.5, no.1, 1994.

[41] D.B. Fogel, "An introduction to simulated evolutionary optimization," *IEEE Trans. on Neural Networks,* vol.5, no.1, pp.3-14, 1994.

[42] A. Gersho, "Asymptotically optimal block quantization," *IEEE Trans. on Information Theory*, vol.25, no.4, pp.373-380, 1979.

[43] S. Gottwald and W. Pedrycz, "Solvability of fuzzy relational equations and manipulation of fuzzy data," *Fuzzy Sets and Systems* , vol.18, pp.45-65, 1986.

[44] S. Gottwald, "Approximately solving fuzzy relation equations : Some mathematical results and some heuristic proposals," *Fuzzy Sets and Systems,* vol.66, pp.175-193, 1994.

[45] J.J. Grefenstette, "Optimization of control parameters for genetic algorithms," *IEEE Trans. on Syst. Man, Cybernetics*, vol.16, no.1, pp.122-128, 1986.

[46] S. Grossberg, "Adaptive pattern classification and universal recoding : I. Parallel development and coding of neural feature detectors," *Biological Cybernetics*, vol.23, pp.121-134, 1976.

[47] S. Grossberg, "Adaptive pattern classification and universal recoding, II: Feedback, expectation, olfaction, and illusions," *Biological Cybernetics,* vol.23, pp.187-202, 1976.

[48] M.M. Gupta and J. Qi, "Design of fuzzy logic controllers based on generalized T-operators," *Fuzzy Sets and Systems*, vol.40, pp.473-489, 1991.

[49] S. Haykin, *Neural networks : A comprehensive foundation.* New York:IEEE Press, 1994.

[50] J. Hertz, A. Krogh, & R.G. Palmer, *Introduction to the theory of neural computation.* California:Addison-Wesley, 1991.

[51] M. Higashi and G.J. Klir, "Identification of fuzzy relation systems," *IEEE Trans. on Syst. Man, Cybernetics*, vol.14, pp.349-355, 1984.

[52] J.J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," *Proc. Nat'l. Acad. Sci. of USA*, vol.79, pp.2554-2558, April 1982.

[53] T. Huntsberger & P. Ajjimarangsee, "Parallel self-organizing feature maps for unsupervised pattern recognition," *General Systems*, vol.16, pp.357-372, 1990.

[54] J.S.R. Jang, "ANFIS : Adaptive-network-based fuzzy inference system," *IEEE Trans. on Sys., Man, & Cybern.*, vol.23, no.3, pp.665-684, 1993.

[55] A. Kaufmann and M.M. Gupta, *Introduction to fuzzy arithmetic.* New York:Van Nostrand, 1985.

[56] J.M. Keller, M.R. Gray, & J.A. Givens Jr., "A fuzzy k-nearest neighbor algorithm," *IEEE Transactions on Systems, Man, and Cybernetics*, vol.15, pp.580-585, 1985.

[57] J.M. Keller & D.J. Hunt, "Incorporating fuzzy membership functions into the perceptron algorithm," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol.7, pp.693-699, 1985.

[58] S.J. Kia & G.G. Coghill, "Unsupervised clustering and centroid estimation using dynamic competitive learning," *Biological Cybernetics*, vol.67, pp.433-443, 1992.

[59] S. Kirkpatrick, C.D. Gelatt, Jr. and M.P. Vecchi, "Optimization by simulated annealing," *Science*, vol.220, pp.671-680, 1983.

[60] G.J. Klir and T.A. Folger, *Fuzzy sets, uncertainty, and information.* Englewood Cliffs, NJ:Prentice-Hall, 1988.

[61] T. Kohonen, "Learning vector quantization for pattern recognition," Technical Report No. TKK-F-A601, Helsinki University of Technology, Finland, 1986.

[62] T. Kohonen, G. Barna, and R. Chrisley, "Statistical pattern recognition with neural networks : Benchmarking Studies," *Proc. IEEE Int. Conf. on Neural Networks (ICNN-88)*, San Diego, CA, Vol.I, pp.61-68, 1988.

[63] T. Kohonen, "The self-organizing map," *Proceedings of the IEEE*, vol.78, no.9, pp.1464-1480, 1990.

[64] S.G. Kong and B. Kosko, "Differential competitive learning for centroid estimation and phoneme recognition," *IEEE Transactions on Neural Networks*, vol.2, pp.118-124, 1991.

[65] S.G. Kong and B. Kosko, "Adaptive fuzzy systems for backing up a truck-and-trailer," *IEEE Trans. on Neural Networks*, vol.3, no.2, pp..211-223, 1992.

[66] B. Kosko, "Bidirectional associative memories," *IEEE Trans. Sys., Man & Cybern.*, vol.18, no.1, pp.49-60, 1988.

[67] B. Kosko, *Neural networks and fuzzy systems : A dynamical system approach to machine intelligence.* New Jersey:Prentice-Hall, 1992.

[68] A.K. Krishnamurthy, S.C. Ahalt, D.E. Melton, & P. Chen, "Neural networks for vector quantization of speech and images," *IEEE Journal on Selected Areas in Communications*, vol.8, pp.1449-1457, 1990.

[69] R. Krishnapuram and J.M. Keller, "A possibilistic approach to clustering," *IEEE Trans. Fuzzy Systems*, vol.1, no.2, pp.98-110, 1993.

[70] P.M. Larsen, "Industrial applications of fuzzy logic control," Man-Machince Studies, vol.12, no.1, pp.3-10, 1980.

[71] C.C. Lee, "Fuzzy logic in control systems : Fuzzy logic controller, Part I," *IEEE Trans. on Sys., Man, & Cybern.*, vol.20, no.2, pp.404-418, 1990.

[72] C.C. Lee, "Fuzzy logic in control systems : Fuzzy logic controller, Parts II," *IEEE Trans. Syst. Man, Cybern.*, vol.20, no.2, pp.419-435, 1990.

[73] Y.C. Lee, C. Hwang, and Y.P. Shih, "A combined approach to fuzzy model identification," *IEEE Trans. on Syst. Man, Cybernetics*, vol.24, pp.736-743, 1994.

[74] C.T. Lin and C.S.G. Lee, "Neural-network-based fuzzy logic control and decision system," *IEEE Trans. on Computers*, vol.40, no.12, pp.1320-1336, Dec. 1991.

[75] C.T. Lin, *Neural fuzzy control systems with structure and parameter learning*. Singapore:World Scientific, 1994.

[76] M.C. Mackey and L. Glass, "Oscillation and chaos in physiological control systems," Science, vol.197, pp.287-289, July 1977.

[77] E.H. Mamdani, "Application of fuzzy logic to approximate reasoning using linguistic synthesis," *IEEE Trans. Computers*, vol.26, pp.1182-1191, 1977.

[78] S. Mitra and S.K. Pal, "Self-organizing neural network as a fuzzy classifier," *IEEE Trans. Sys., Man & Cybern.*, vol.24, no.3, pp.385-399, 1994.

[79] M. Mizumoto, "Comparison of fuzzy reasoning methods," *Fuzzy Sets and Systems*, vol.8, pp.253-283, 1982.

[80] M. Mizumoto, "Pictorial representations of fuzzy connectives Part I : Cases of t-norms, t-conorms and averaging operators," *Fuzzy Sets and Systems*, vol.31, pp.217-242, 1989.

[81] T.D. Ndousse, "Fuzzy neural control of voice cells in ATM networks," IEEE J. on Selected Areas in Communications, vol.12, no.9, pp.1488-1494, 1994.

[82] A. Ohsato and T. Sekiguchi, "Comvexly combined form of fuzzy relational equations and its application to knowledge representation," *Proc. Int. Conf. Syst. Man & Cybernetics*, Bombay, pp. 294-299, 1984.

[83] J. V. de Oliveira, "Neuron inspired learning rules for fuzzy relational structures," *Fuzzy Sets and Systems*, vol.57, pp.41-53, 1993.

[84] N.R. Pal, J.C. Bezdek, and Eric C.K. Tsao, "Generalized clustering networks and Kohonen's self-organizing scheme," *IEEE Trans. Neural Networks*, vol.4, no.4, pp.549-557, 1993.

[85] S.K. Pal and S. Mitra, "Multilayer perceptron, fuzzy sets, and classification," *IEEE Trans. Neural Networks*, vol.3, no.5, pp.683-697, 1992.

[86] Y.H. Pao, *Adaptive pattern recognition and neural networks.* New York:Addison-Wesley, 1989.

[87] D. Park, A. Kandel, and Gideon Langholz, "Genetic-based new fuzzy reasoning models with application to fuzzy control," *IEEE Trans. on Syst. Man, Cybernetics*, vol.24, pp.39-47, 1994.

[88] W. Pedrycz, "Numerical and applicational aspects of fuzzy relational equations," *Fuzzy Sets and Systems*, vol.11, pp.1-18, 1983.

[89] W. Pedrycz, "Fuzzy relational equations with generalized connectives and their applications," *Fuzzy Sets and Systems*, vol.10, pp.185-201, 1983.

[90] W. Pedrycz, "Identification in fuzzy systems," *IEEE Trans. on Syst. Man, Cybernetics*, vol.14, pp.361-366, 1984.

[91] W. Pedrycz, "An identification algorithm in fuzzy relational systems," *Fuzzy Sets and Systems*, vol.13, pp.153-167, 1984.

[92] W. Pedrycz, "Approximate solutions of fuzzy relational equations," *Fuzzy Sets and Systems,* vol.28, pp.183-202, 1988.

[93] W. Pedrycz, "Fuzzy relational equations with equality and difference composition operators," *Fuzzy Sets and Systems*, vol.25, pp.205-215, 1988.

[94] W. Pedrycz, "Processing in relational structures : Fuzzy relational equations," *Fuzzy Sets and Systems*, vol.40, pp.77-106, 1991.

[95] W. Pedrycz, "Neurocomputations in relational systems," *IEEE Trans. on Pattern Analysis and Machince Intelligence*, vol.12, pp.289-297, 1991.

[96] W. Pedrycz, "Fuzzy neural network with reference neurons as pattern classifiers," *IEEE Trans. Neural Networks*, vol.3, no.5, pp.770-775, 1992.

[97] W. Pedrycz, *Fuzzy control and fuzzy systems*, 2nd extended edition. Taunton/New York:Research Studies Press/John Wiley & Sons, 1993.

[98] W. Pedrycz, "s-t Fuzzy relational equations," *Fuzzy Sets and Systems*, vol.59, pp.189-195,1993.

[99] A.J. Robinson, *Dynamic error propagation networks.* Ph.D. Dissertation, Cambridge University Engineering Department, Cambrigde, UK, 1989.

[100] G. Rudolph, "Convergence properties of canonical genetic algorithms," *IEEE Trans. on Neural Networks (Special Issue on evolutionary computation)*, vol.5, no.1, pp.96-101, 1994.

[101] D.E. Rumelhart, G.E. Hinton, & R.J. Williams, "Learning internal representations by error propagation," In D.E. Rumelhart & J.L. McClelland (Eds.), *Parallel distributed processing : Explorations in the microstructure of cognition* (Vol.I, Ch.7), Cambridge, MA:MIT Press, 1986.

[102] D.E. Rumelhart & D. Zipser, "Feature discovery by competitive learning," In D.E. Rumelhart & J.L. McClelland (Eds.), *Parallel distributed processing : Explorations in the microstructure of cognition* (Vol.I, Ch.5), Cambridge, MA:MIT Press, 1986.

[103] E. Sanchez, "Resolution of composite fuzzy relation equations," *Inform. and Control* , vol. 30, pp. 38-48, 1976.

[104] D.G. Schwartz, G.J. Klir, H.W. Lewis III, Y. Ezawa, "Applications of fuzzy sets and approximate reasoning," *Proceedings of the IEEE*, vol.82, no.4, pp.482-498, 1994.

[105] S.Z. Selim & M.S. Kamel, "On the mathematical and numerical properties of the fuzzy c-means algorithm," *Fuzzy Sets and Systems*, vol.49, pp.181-191, 1992.

[106] S. Sessa, "Some results in the setting of fuzzy relation equations theory," *Fuzzy Sets and Systems*, vol.14, pp.281-297, 1984.

[107] H. Takagi and M. Sugeno, "Derivation of fuzzy control rules from human operator's control actions," *Proc. IFAC Symp. Fuzzy Information, Knowledge Representation and Decision Analysis*, pp.55-60, 1983.

[108] T. Tanaka, M. Naka, and K. Yoshida, "Improved back-propagation combined with LVQ," *Proc. Int. Joint Conf. on Neural Networks (IJCNN-90-WASH-DC)*, Washington, DC, Vol.I, pp.731-734, 1990.

[109] E. Trauwaert, L. Kaufman, & P. Rousseeuw, "Fuzzy clustering algorithms based on the maximum likelihood principle," *Fuzzy Sets and Systems*, vol.42, pp.213-227, 1991.

[110] E.C.K. Tsao, J.C. Bezdek, & N.R. Pal, "Fuzzy Kohonen clustering networks," *Pattern Recognition*, vol.27, no.5, pp.757-764, 1994.

[111] T. Uchiyama and M.A. Arbib, "Color image segmentation using competitive learning," *IEEE Trans. on Pattern Analy. & Mach. Intell.*, vol.16, no.12, pp.1197-1206, 1994.

[112] L.X. Wang, *Adaptive fuzzy systems and control : Design and stability analysis.* Englewood Cliffs, NJ:PTR Prentice Hall, 1994.

[113] C.W. Xu and Y.Z. Lu, "Fuzzy model identification and self-learning for dynamic systems," *IEEE Trans. on Systems, Man, and Cybernetics*, vol.17, pp.683-689, 1987.

[114] R.R. Yager, "On a general class of fuzzy connectives," *Fuzzy Sets and Systems*, vol.4, pp.235-242, 1980.

[115] X. Yao, "A review of evolutionary artificial neural networks," *Intelligent Systems*, vol.8, pp.539-567, 1993.

[116] P.P.C. Yip and Y.H. Pao, "A guided evolutionary simulated annealing approah to the quadratic assignment problem," *IEEE Trans. on Syst. Man, Cybernetics*, vol.24, pp.1383-1387, 1994.

[117] P.P.C. Yip and Y.H. Pao, "Combinatorial optimization with use of guided evolutionary simulated annealing," *IEEE Trans. on Neural Networks*, vol.6, no.2, pp.290-295, 1995.

[118] L.A. Zadeh, "Fuzzy sets," *Information and Control*, vol.8, pp.338-353, 1965.

[119] L.A. Zadeh, "Outline of a new approach to the analysis of complex systems and decision processes," *IEEE Trans. Sys., Man & Cybern.*, vol.3, no.1, pp.28-44, 1973.

[120] L.A. Zadeh, "The concept of a linguistic variable and its application to approximate reasoning," *Information Science*, Part I : vol.8, pp.199-249, 1975, Part II : vol.8, pp.301-357, 1975, Part II : vol.9, pp.43-80, 1975.

[121] L.A. Zadeh, "Calculus of fuzzy restriction," in L.A. Zadeh et al., Eds., *Fuzzy sets and their applications to cognitive and decision processes.* New York:Academic Press, pp.1-39, 1975.

[122] C. Zhu, L.H. Li, T.J. Wang, and Z.Y. He, "Parital-distortion-weighted fuzzy competitive learning algorithm for vector quantization," *Electronic Letters*, vol.30, no.6, pp.505-506, 1994

[123] H.J. Zimmermann, *Fuzzy set theory and its applications.* Boston:Kluwer Academic Publishers, 1985.