A FUZZY-GENETIC DECISION SUPPORT SYSTEM FOR MEDICAL DIAGNOSIS

A DISSERTATION SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE AWARD OF THE DEGREE OF

MASTER OF ENGINEERING

IN

CONTROL & INSTRUMENTATION

SUBMITTED BY

BEENA ANTONY
(Roll NO. 8666)

UNDER THE ESTEEMED GUIDANCE
OF

Dr. PARMOD KUMAR
(PROFESSOR & HEAD)

DEPARTMENT OF ELECTRICAL ENGINEERING
DELHI COLLEGE OF ENGINEERING
UNIVERSITY OF DELHI
2004-2006

CERTIFICATE

It is certified that Ms. Beena Antony, Roll No.8666, student of M.E, Control and Instrumentation, Department of Electrical Engineering, Delhi College of Engineering, has submitted the dissertation entitled **“A fuzzy-genetic decision support system for medical diagnosis”**, under my guidance towards partial fulfillment of the requirements for the award of the degree of Master of Engineering (Control & Instrumentation Engineering).

This dissertation is a bonafide record of project work carried out by her under my guidance and supervision. Her work is found to be excellent and her discipline impeccable during the course of the project.

I wish her success in all her endeavors.

Date : **(Dr. Parmod Kumar)**

Professor & Head

Dept. of Electrical Engineering

Delhi College of Engineering

Delhi -110042

CONTENTS

ABSTRACT

The automatic diagnosis of diseases like breast cancer, liver disorder and diabetes are real-world medical problems. A novel approach for diagnosing these diseases is undertaken. This approach based on the evolution of the entire fuzzy inference system for each diagnosis problem, shows that, it is possible to obtain diagnostic systems exhibiting high performance, coupled with interpretability and a confidence measure. The evolved fuzzy systems are able to accurately predict the outcome of a human decision-making process, while providing an understandable explanation of the underlying reasoning.

Fuzzy logic provides a formal framework for constructing systems exhibiting both good classification performance and interpretability. Linguistically, fuzzy system represents knowledge in the form of rules, a natural way for explaining decision processes. Optimization of both knowledge base and rule base is critical to the performance of a fuzzy system. Genetic algorithm, a genetically inspired optimization technique is used to evolve the entire fuzzy system. The concept of evolutionary fuzzy modeling - the design of fuzzy inference systems using evolutionary algorithms is the basis for the decision tool.

The algorithm is successfully applied to model the decision processes involved in the above mentioned three diagnostic problems. The evolved fuzzy system is finally tested for diagnosis. The diagnostic decision tool has been implemented and tested in MATLAB 7.0.

LIST OF FIGURES

LIST OF TABLES

CHAPTER I

INTRODUCTION

1.1 Problem Description

A major class of problems in medical science involves the diagnosis of disease, based upon various tests performed upon the patient. When several tests are involved, the ultimate diagnosis may be difficult to obtain, even for a medical expert. Such data stored in database is growing at a phenomenal rate. Although data collection has become easier, the difficulties required to retrieve relevant knowledge from the database has become significantly increased. Obviously, such kind of raw data is rarely of direct benefit. So, the value of these data is predicated on the ability to extract information useful for decision support or exploration, and understanding the phenomenon governing the data source. Traditionally, data analysis to retrieve the knowledge by the analyst(s) was a manual process instead of the automatic process. However, those manual processes easily break down while the size of the data grows and the number of dimensions increases. For dealing with the scale of data manipulation, exploration going beyond human capacities, the computing technologies for automating the process is desired and need to be developed. This has given rise, over the past few decades, to computerized diagnostic tools, intended to aid the physician in making sense out of the welter of data. A prime target for such computerized tools is in the domain of cancer diagnosis.

Present-day databases contain a huge amount of data concerning human decisions that should be used to model decision-making processes. These data are, however, just a collection of recorded facts that do not contain by themselves any information or knowledge useful to explain or to predict the decisions. There exist many methods that, based on these data, can build systems to predict the outcome of a given decision. Albeit useful and widely used, these methods and systems lack the explanatory power required to transmit knowledge to humans. A good computerized tool for medical decision support should possess two characteristics, which are often in conflict.

- The tool must attain the highest possible performance, i.e. diagnose and classify the presented cases correctly. Moreover, it would be highly desirable for it to have a *degree of confidence*: the system not only provides a binary diagnosis ,but also outputs a numeric value that represents the degree to which the system is confident about its response.

- It would be highly beneficial for such a diagnostic system to be human-friendly, exhibiting good *interpretability*. This means that the physician is not faced with a black box that simply spouts answers with no explanation; rather, we would like for the system to provide some insight as to *how* it derives its outputs.
- Another crucial factor of a hybrid system is the speed of process and the time needed to produce a generalized high-performance decision model.

1.2 Objective of study

The objective of the dissertation is to obtain a fuzzy-genetic diagnostic decision support system in the field of medicine. Breast cancer, Lung cancer, Liver cancer , diabetes diagnosis are the prime focus of this dissertation. The general problem that motivates this thesis is the development of an approach to automatically construct systems which predict, as accurately as possible, the outcome of a human decision-making process while providing an understandable explanation of a possible reasoning leading to it. This report investigates a promising method of control engineering, evolutionary fuzzy modeling. Fuzzy-genetics are soft computing techniques. The strengths of genetic algorithms and fuzzy logic are explained with the express purpose of proposing how, when combined, a useful and workable method of diagnosis may result. This dissertation describes the use of a Hybrid Fuzzy-Genetic Programming system to diagnose respective diagnostic classes in large databases. It does this by evolving an entire fuzzy inference system within a chromosome; generalizing from a training set of labeled classes. Eventually after proper validation, diagnosis can be done using the optimized fuzzy inference system.

1.3 Proposed solution

In this thesis there is combination of two methodologies-fuzzy systems and evolutionary algorithms—so as to automatically produce systems for medical diagnosis. The proposed solution is a diagnostic decision support - a classifier system. A classifier system is basically evolution based learning system .Once the computer or the radiologist extracts relevant features, a classifier must select and optimally merge them into a diagnostic decision aid.

The fuzzy classifier system is a machine learning system which employs linguistic rules and fuzzy sets in its representation and an evolutionary algorithm for rule and membership function parameter discovery. In a fuzzy classification system, a case or an object can be classified by applying a set of fuzzy rules based on the linguistic values of its attributes. It therefore combines an easily understood representation with a general purpose search method. The major advantage of fuzzy systems is that they favor interpretability, however, finding good fuzzy systems can be quite an arduous task. This is where evolutionary algorithms step in, enabling the automatic production of fuzzy systems, based on a database of training cases. The resulting approach, is a fuzzy modeling technique, based on evolution, conceived to provide high numeric precision (accuracy), while incurring as little a loss of linguistic descriptive power (interpretability) as possible. The use of Genetic algorithms in classification is an attempt to effectively exploit the large search space usually associated with classification tasks.

In this solution two coevolving parameters are defined: rules and membership functions. It proves to be very efficient in designing highly accurate and interpretable systems to solve hard problems, in particular modeling medical diagnostic decision processes.

The proposed diagnostic system for breast cancer is shown in the figure 1.1. Same is applied for diagnosis of other diagnostic datasets. It consists of a fuzzy system and a threshold unit. The fuzzy system, upon presentation of an input (database entry), proceeds to produce a continuous appraisal value. This value is then passed along to the threshold unit which produces the final binary output e.g. *benign* or *malignant for breast cancer* diagnosis. The threshold value is assigned based on the knowledge of the problem at hand by the user. The fuzzy subsystem's membership functions and rule base, both are optimized. The appraisal value can accompany the final output of the diagnostic system, serving as a confidence measure. This demonstrates an advantage of the ability to output not only a binary classification but also a measure representing the system's confidence in its output.

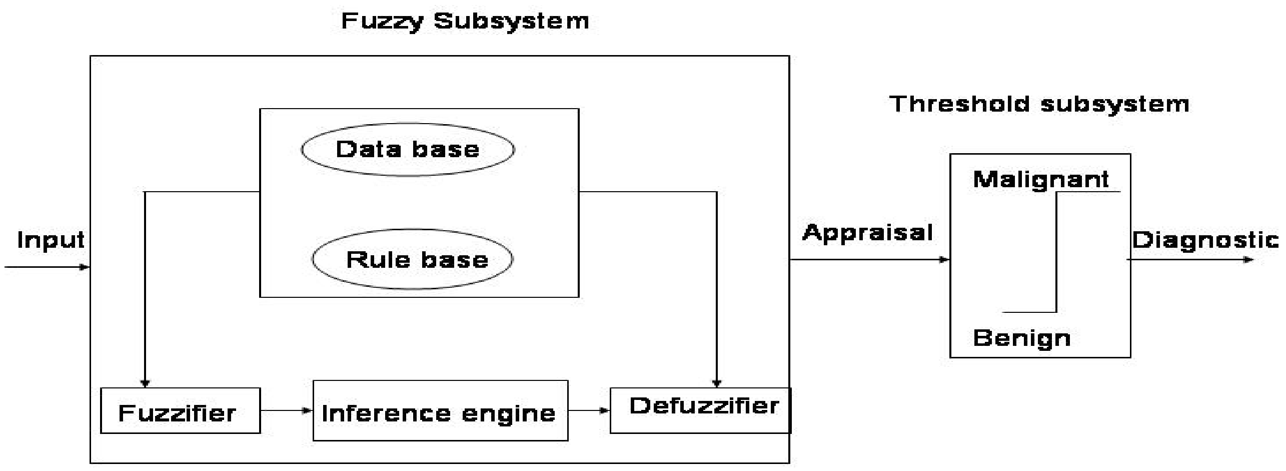


Figure 1.1 Proposed Diagnostic system

1.4 Features of Fuzzy Logic

Fuzzy logic is a computational paradigm that provides a mathematical tool for representing and manipulating information in a way that resembles human communication and reasoning processes. It is based on the assumption that, in contrast to Boolean logic, a statement can be *partially* true (or false), and composed of imprecise concepts. Fuzzy theory provides us linguistic representation such as slow and fast and thus uses truth degrees, which are represented as grades of a membership function. Fuzzy logic is a powerful tool for non-probabilistic and ill-defined structures. Fuzzy inference system is based on the concepts of fuzzy set theory, fuzzy if-then rules, and fuzzy inferences. A fuzzy inference system implements a mapping from its input space to output space by a number of fuzzy if-then rules.

Fuzzy Logic systems are rule-based systems in which an input is first *fuzzified* (i.e., converted from a crisp number to a fuzzy set) and subsequently processed by an inference engine that retrieves knowledge in the form of fuzzy rules contained in a rule-base. The fuzzy sets computed by the fuzzy inference as the output of each rule are then composed and *defuzzified* .The fuzzy subsystem shown in figure 1.1 shows the main components of a typical fuzzy inference system performing the following steps:

1. Fuzzification

It is the process of transformation of crisp measurement into corresponding set of degree of belonging. Before the measurements are fuzzified, however, they should be first normalized to the range of universe of discourse.

2. Inferencing

Inferencing is the process of reasoning the fuzzy set value according to the rules to estimate/recognize the objectives. A rule base is a summary of a set of logic properties with one or more antecedents and inferred conclusions.

3. Defuzzification

At the output of the fuzzy inference there will always be a fuzzy set that is obtained by the composition of the fuzzy sets output by each of the rules .In order to be used in the real world, the fuzzy output needs to be interfaced to the *crisp* domain by the defuzzifier. The output fuzzy set indicates what the output is in fuzzy terms. The *crisp* output corresponding to a certain fuzzy output set should be a number that takes into account all the points in the support of this fuzzy output, weighing the points with high membership degree more than the ones with small or no membership degree. This corresponds to a center of gravity operation. Thus, one widely used defuzzifier is the *centroid* defuzzifier that transforms a fuzzy output set into a number that is the *x*-coordinate of the set's center of gravity.

Advantages of Fuzzy Logic

- Fuzzy logic is conceptually easy to understand. The mathematical concepts behind fuzzy reasoning are very simple. Fuzzy logic is based on natural language. Since fuzzy logic is built atop the structures of qualitative description used in everyday language, fuzzy logic is easy to use.
- Fuzzy logic is tolerant of imprecise data.
- Fuzzy logic can model nonlinear functions of arbitrary complexity.
- Fuzzy logic can be built on top of the experience of experts. Thus fuzzy logic lets you rely on the experience of people who already understand your system.
- Fuzzy logic can be blended with conventional control techniques.

1.5 Features of Genetic Algorithm

Genetic Algorithm is one of the evolutionary optimization method techniques that perform parallel, stochastic, but direct search method to evolve the best solution. It is fundamentally iterative generation and alternation processes operating on a set of candidate solutions in a given problem space. Genetic algorithms are usually applied to spaces which are too large to be exhaustively searched.

Genetic Algorithms simulate the survival of the fittest among individuals, encoding a possible solution, over consecutive generation for solving a problem. The individuals in the population are then made to go through a process of evolution. A *fitness score* is assigned to each solution representing the abilities of an individual to 'compete'. The individual with the optimal fitness score is sought. The entire population evolves towards better candidate solutions via the selection operations and genetic operators such as crossover and mutation. The selection operator decides which candidate solutions move into the next generation, which limits the search space. The cross over and mutation operators generate new candidate solutions from the search space.

In this way it is hoped that over successive generations better solutions will thrive while the least fit solutions die out. Eventually, once the population has converged and is not producing offspring noticeably different from those in previous generations, the algorithm itself is said to have converged to a set of solutions to the problem at hand.

1.6 Dissection of the dissertation

Chapter 1 has identified the problem and the objective of the study. Some features of fuzzy Logic and genetic algorithm are also introduced.

In relation to the work performed in this study, the remainder of the report consists of the following chapters;

Chapter 2 gives the literature review about the work undertaken in the field of fuzzy, genetic algorithms and fuzzy genetics in the medical field.

Chapter 3 presents the fundamentals of fuzzy logic, fuzzy modeling and describes fuzzy logic systems.

Chapter 4 discusses genetic algorithms in detail. It describes major components of genetic algorithm and the effects of genetic operators.

Chapter 5 introduces evolutionary fuzzy modeling techniques. The different learning approaches with genetic algorithm are presented. It also presents the concept of evolutionary computation to solve medical problems.

Chapter 6 deals with the crux of the dissertation. It discusses the breast cancer, liver disorder and diabetes diagnosis problems and presents the approach used for diagnosing these diseases. The solution methodology is shown in a flowchart form.

Chapter 7 analyses the final evolved fuzzy system for each diagnosis. It has the results and discussions related to each diagnosis problem. Finally each optimized fuzzy system is tested to get actual diagnosis in each case.

Chapter 8 is conclusion and future scope of the project.

CHAPTER II

LITERATURE REVIEW

2.1 Introduction

The past few years have witnessed a rapid growth in the number and variety of applications of fuzzy logic. A trend that is growing in visibility relates to the use of fuzzy logic in combination with neuro-computing and genetic algorithms. More generally, fuzzy logic, neurocomputing, and genetic algorithms may be viewed as the principal constituents of what might be called soft computing. Soft computing in the field of medical diagnosis is a technology which extracts information from the medical signal by using expert knowledge. It either seeks to replace physician to perform diagnosis task or it borrows ideas from how biological system solves problems and apply to diagnosis.

2.2 Fuzzy Logic

The concept of Fuzzy Logic was conceived by Lotfi Zadeh, a professor at the University of California at Berkley, as a mathematical way to represent and deal with vagueness in everyday life and presented not as a control methodology, but as a way of processing data by allowing partial set membership rather than crisp set membership or non-membership. This approach to set theory was not applied to control systems until the 70's due to insufficient small-computer capability prior to that time. Professor Zadeh reasoned that people do not require precise, numerical information input, and yet they are capable of highly adaptive control .Other research followed, with the first industrial application, a cement kiln built in Denmark, coming on line in 1975. The Japanese interest in fuzzy systems was sparked by Seiji Yasunobu and Soji Miyamoto of Hitachi, who in 1985 provided simulations that, demonstrated the superiority of fuzzy control systems for the Sendai railway. Their ideas were adopted, and fuzzy systems were used to control accelerating, braking, and stopping when the line opened in 1987 [3]. During an international meeting of fuzzy researchers in Tokyo that year, Takeshi Yamakawa demonstrated the use of fuzzy control, through a set of simple dedicated fuzzy logic chips, in an ‘inverted pendulum’ experiment.

2.2 Genetic algorithm

Proposed by John Holland in the 1960s [8], genetic algorithms are the best known class of evolutionary algorithms. The basic techniques of the genetic algorithm are designed to simulate processes in natural systems necessary for evolution; especially those follow the principles laid down by Charles Darwin of ‘survival of the fittest’. L. J. Fogel’s paper in this area triggered the study and the application of evolutionary techniques .[5]

Research in Genetic Algorithms remained largely theoretical until the mid-1980s, when the first International Conference on Genetic Algorithms was held at the University of Illinois. Kenneth De Jong's important dissertation established the potential of Genetic Algorithm by showing that they could perform well on a wide variety of test functions, including noisy, discontinuous, and multimodal search landscapes. These foundational works established more widespread interest in evolutionary computation. By the early to mid-1980s, genetic algorithms were being applied to engineering issues such as pipeline flow control, pattern recognition and classification, and structural optimization. At first, these applications were mainly theoretical. Today, evolutionary computation is a thriving field, and genetic algorithms are ‘solving problems of everyday interest’ such as stock market prediction and portfolio planning, aerospace engineering, microchip design, biochemistry and molecular biology, and scheduling at airports and assembly lines.

2.3 Fuzzy Genetics

Two of the most successful approaches to hybridize fuzzy systems with learning and adaptation methods have been made in the realm of soft computing. Neural fuzzy systems and genetic fuzzy systems hybridize the approximate reasoning method of fuzzy systems with the learning capabilities of neural networks and evolutionary algorithms. Genetic Fuzzy Systems are genetic fuzzy rule based systems whose genetic process learns or tunes different components of a fuzzy rule-based system. [3]

Voget et al. presented a multi-objective optimization scheme, in which a fuzzy controller regulates the selection procedure and fitness function of the Genetic algorithm. In this fuzzy evolutionary approach a fuzzy system manages the resources and parameters of a Genetic Algorithm such as mutation rate, population size and selective pressure to improve the performance. [24]

Yuhui Shi, et al discussed an evolutionary fuzzy system in which the membership function shapes and types and the fuzzy rule set including the number of rules inside it is evolved using genetic algorithm. [26]

Medical domains of application (eg. medical diagnosis from images data or a typical characteristics, monitoring of an evolving situation, detection of a sudden change of status in a living system, decision making under uncertainty, etc.) seem to comprise a central field of research for observing the effectiveness and usability of applying hybrid computational intelligence architectures.

Genetic algorithms are used for solving fuzzy logical equations in (medical) diagnostic expert systems [17]. Fidelis et al presented a classification algorithm based on genetic algorithm that discovers *If Then* rules .The proposed genetic algorithm has flexible chromosome encoding where each chromosome corresponds to a classification rule. The algorithm has been evaluated on data sets of breast cancer and dermatology [4]. R. Jain et al. presented an application of a genetic-algorithm-based representation of fuzzy rules for the classification of coronary artery disease data and breast cancer data. In this study the concept of fuzzy if-then has been applied for a multi dimensional data classification problem which leads to higher classification power. The classification power on real world data for coronary artery disease and breast cancer was thus demonstrated by computer simulations [9]. Robust ECG R-R wave peak detection using an evolutionary programming-based fuzzy inference system (EPFIS) and its application to assessing brain-gut interaction was demonstrated by Wang, Z.S et al. [25].

There are several studies are based on Wisconsin Breast Cancer Diagnosis database of which Bennet and Mangasarian used linear programming techniques, obtaining a 99.6% classification rate [2]. However, their solution exhibits little understandability, i.e., diagnostic decisions are essentially black boxes, with no explanation as to how they were attained. With increased interpretability in mind as a prior objective, a number of researchers like R. Setino has applied the method of extracting Boolean rules from neural networks[19] [20] . He has designed an algorithm that extracts classification rules from trained neural networks and discussed its application to breast cancer diagnosis. Their results are encouraging, exhibiting both good performance and a reduced number of rules and relevant input variables. The systems defined in the above researches use Boolean rules and are not capable of furnishing the user with a measure of confidence for the decision made.

2.4 Conclusion

This chapter has dealt with the research work done in the area of soft computing viz. fuzzy and genetics, applied to the medical field. Documenting the exhaustive work that has been undertaken in this field is an arduous task and thus this chapter just gives an overview of the literature review.

CHAPTER III

FUZZY MODELING

3.1 Introduction

Fuzzy logic is a fascinating area of research because it does a good job of trading off between significance and precision — something that humans have been managing for a very long time. Fuzzy logic is a convenient way to map an input space to an output space. Everyday language is the cornerstone example of vagueness and is representative of how we assimilate and act upon vague situations and instructions. It may be said that humans assimilate and use (act on) fuzzy data, vague rules, and imprecise information. Just as we are able to make decisions about situations which seem to be governed by an element of chance, accordingly, computational models of real systems should also be able to recognize, represent, manipulate, interpret, and use (act on) both fuzzy and statistical uncertainties. Fuzzy interpretations of data structures are a very natural and intuitively plausible way to formulate and solve various problems.

Fuzzy controllers are particularly suited to applications where it is not necessary to find the global optimum solution, that is, where a near optimum solution is sufficient. Traditional control systems are based on mathematical models in which the control system is described using one or more differential equations that define the system response to its inputs. Such systems are often implemented as P, P+I and P+I+D controllers. Uncertainty and imprecision, caused by non-linearity such as noise and disturbances, are also evident in the operation of real-world systems. The use of mathematical models to develop such systems may result in poor system performance if the dynamics of actual events do not form part of the model. This is because a full and complete mathematical model describing every possible cause and- effect event is not feasible [11]. The required degree of mathematical precision becomes even greater for non-linear system design. In many cases, the mathematical model of the control process may not exist, or may be too "expensive" in terms of computer processing power and memory, and a system based on empirical rules may be more effective. Whereas a fuzzy logic solution is tolerant to the imprecision in the inputs and the model of the system and still produces an output that is desired out of the system.

In this context, Fuzzy Logic is a problem-solving control system methodology that lends itself to implementation in systems ranging from simple, small, embedded microcontrollers to large, networked, multi-channel PC or workstation-based data acquisition and control systems. It can be implemented in hardware, software, or a combination of both. Fuzzy Logic provides a simple way to arrive at a definite conclusion based upon vague, ambiguous, imprecise, noisy, or missing input information.

3.2 Fuzzy Logic Basics

3.2.1 Physical modeling

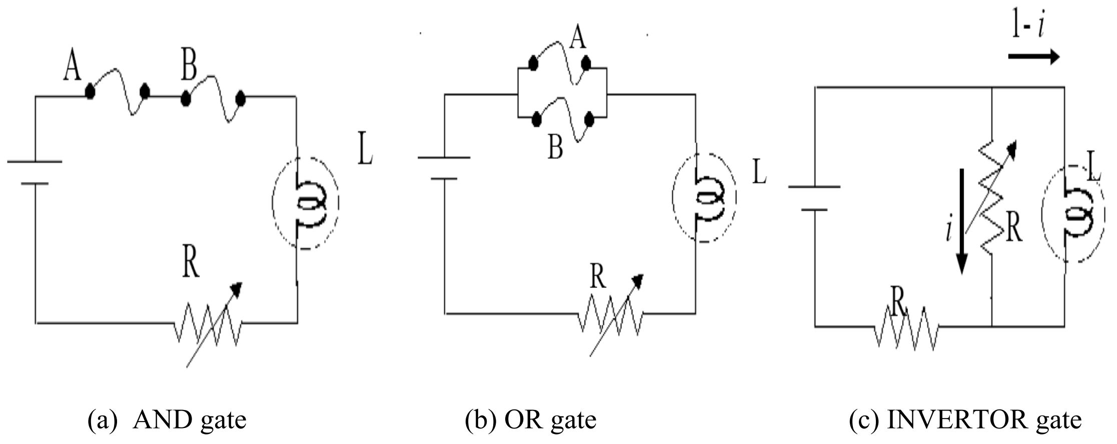


Figure 3.1 Physical Model of fuzzy logic gates

Similar to the case of binary operation fuzzy operations can be illustrated by simple circuits. Instead of wing switches to represent inputs, we use fuses-a wire that burns out if the current through it exceeds certain limit, leading to an open-circuit between its terminals. Figure 3.1 shows circuit that illustrates fuzzy AND, OR and INVERTOR operations. Figure 3.1 (a) shows a simple circuit that illustrated the concept of fuzzy AND. Two fuses with different ratings are arranged in series in the circuit. The fuses with lowest rating determine the maximum current that go through the circuit and hence the brightness of the bulb, i.e.

Brightness ∞ min [A, B]

Figure 3.1(b) illustrates the operation of fuzzy OR. The fuse with highest rating determines the brightness of the light, i.e.

Brightness ∞ max [A, B]

Figure 3.1(c) illustrates the operation of Fuzzy INVERTOR. If the current in the circuit is unity, and current in the parallel resistance is I, then the current through the lamp is (1-I)

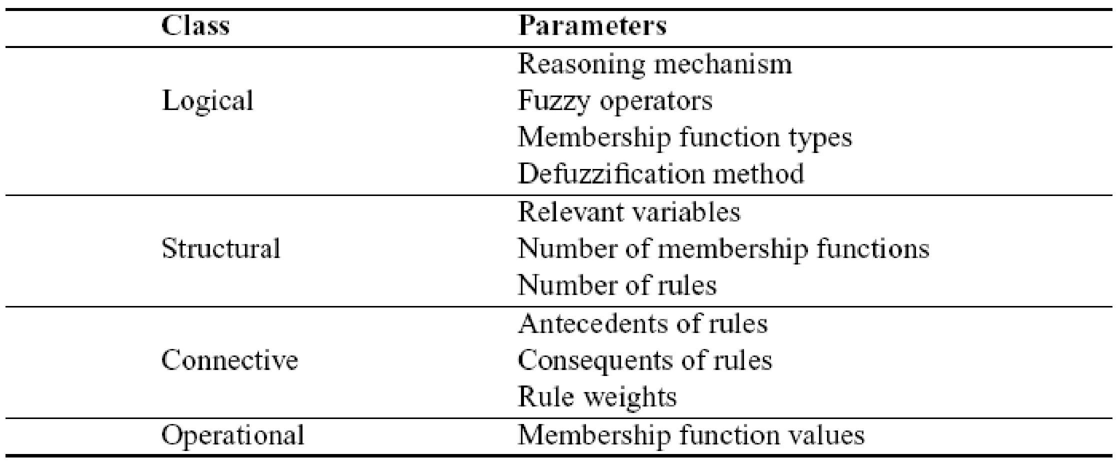
Brightness ∞ (1-I)

3.2.2 The direct approach to fuzzy modeling

Fuzzy modeling techniques of constructing fuzzy rule-based inference systems, is an approach to form a system model using a description language based on fuzzy logic with fuzzy predicates. *Fuzzy modeling* is basically the task of identifying the parameters of a fuzzy inference system so that a desired behavior is attained.

The principles of fuzzy modeling were outlined by Zadeh [28] where he proposed a new approach that "provides an approximate and yet effective means of describing the behavior of systems which are too complex or too ill-defined to admit use of precise mathematical analysis." The models proposed by Zadeh present three distinguishable features: the use of linguistic variables in place or in addition to numerical variables, the description of simple relations between variables by conditional fuzzy statements, and the characterization of complex relations by fuzzy algorithms. The direct approach of fuzzy modeling is still considered as an efficient modeling procedure.

Table 3.1 Fuzzy logic parameter classes



Class	Parameters
Logical	Reasoning mechanism
	Fuzzy operators
	Membership function types
	Defuzzification method
Structural	Relevant variables
	Number of membership functions
	Number of rules
Connective	Antecedents of rules
	Consequents of rules
	Rule weights
Operational	Membership function values

In this approach, the system is first linguistically described, based on the expert's *a priori* knowledge. It is then translated into the formal structure of a fuzzy model following the steps proposed by Zadeh [28]. There are four classes of fuzzy logic parameters as shown in table 1.

1. Selection of the input, state, and output variables (*structural parameters*);

2. Determination of the universes of discourse which is the range of all possible values for an input to a fuzzy system. (*structural parameters*);

3. Determination of the linguistic labels into which these variables are partitioned (*structural parameters*);

4. Definition of the membership functions corresponding to each linguistic label (*operational parameters*);

5. Definition of the rules that describe the model's behavior (*connective parameters*);

6. Selection of an adequate reasoning mechanism (*logic parameters*);

7. Evaluation of the model adequacy.

The problem with this approach is that unless the human expert knows the system well it is very difficult to design a fuzzy rule base and inference system that is workable, let alone efficient. For complex systems (non-linear for example) tuning these membership functions would require the adjustment of many parameters simultaneously. System identification approach was introduced to overcome the difficulties involved in the direct approach of choosing the fuzzy set's membership functions using a search/optimization technique to aid the selection.

3.2.3 Fuzzy sets

These are sets without clear or crisp boundaries. The elements they contain may only have a partial degree of membership. They are therefore not the same as classical sets in the sense that the sets are not closed. Simply put, fuzzy sets are a clever way to deal with vagueness as we often do in our daily life. While mathematically more complicated than classical sets, fuzzy sets provide a more natural representation of the world. Fuzzy set has the following properties;

- It has smooth boundary.
- Membership in a set is a matter of degree or degree of truth ness i.e. in classical set theory, an object either belong to the set or not (temperature is cold/hot, glass full/empty, person is good/bad) but in fuzzy set theory belongingness is a matter of degree (depend up on grade of membership) e.g. A person can be 80% good and 20% bad.
- In classical set theory every individual object is assigned a membership value either 1 or 0 that discriminate between membership and non membership of the crisp set whereas in fuzzy set an object can take a grade of membership between 0 and 1 i.e. [0 1].
- In classical set theory crisp set are based on a two value logic (yes/no) whereas fuzzy set theory is based on multi-value logic.

Fuzzy set is characterized by a mapping from universe of discourse in interval [0 1]. This mapping is the membership function of the set.

A fuzzy set can be represented in two ways:

1) By enumerating membership value of those elements in the set completely or partially. A fuzzy set A can be defined by enumeration using expression

$$A = \sum \frac{\mu(x_i)}{x_i}$$

where the summation and addition operator refers to the union operation and the notation $\frac{\mu(x_i)}{x_i}$ refers to fuzzy set containing exactly one (partial) element *x* with membership degree $\mu_i(x_i)$

2) By defining the membership function mathematically. The term high in terms of speed can be defined mathematically as :

$$\mu_{high} = \begin{cases} 0 & \text{if } 0 < \text{speed} < 30 \\ (S-20)/40 & \text{if } 30 < \text{speed} < 60 \\ 1 & \text{if } \text{speed} > 60 \end{cases}$$

A fuzzy set whose support is a single point in universe U with $\mu_A(u) = 1$ is called a fuzzy singleton.

3.2.4 Membership functions

A membership function (MF or μ) is a curve that defines how each point in the input space is mapped to a membership value (or degree of membership) between 0 and 1. The shape, the overlapping, peak values, and their continuity properties determine how the fuzzy system can be designed and how it behaves. The input space is sometimes referred to as the universe of discourse.

A classical set is defined by a crisp membership for example

$$A = \{x \mid x > 6\}$$

If X is the universe of discourse and its elements are denoted by x, then a fuzzy set A in X is defined as a set of ordered pairs.

$$A = \{x,\ \mu_{A(x)} \mid x \in X\}$$

$\mu_{A(x)}$ is called the membership function of x in A.

Typical membership function shapes include triangular, trapezoidal and Gaussian functions. The shape is chosen on the basis of how well it describes the set it represents. However, most often used one is triangular and trapezoidal and bell-shaped membership functions, because of the ease with which a parametric, functional description of the membership function can be obtained, stored with minimum use of memory and mathematically efficient, in terms of real time requirements by the inference engine [11]. Crossover points also play an important role in mapping, because if there are no crossovers then there would be discontinuities in the control outputs. Also a symmetrical shape of the function can lead to an easier inference and defuzzification.

Criteria for selecting membership functions

Although, the choice of number, range and shapes of membership functions for a variable is ultimately based on subjective design choice and evaluation of resulting system performance, the following points are worth to be considered in selecting the membership function.

- Assume symmetrically distributed Fuzzy sets across the defined universe of discourse.
- Use an odd number of Fuzzy sets for each variable, this ensures that some fuzzy set will be in middle. Five or seven Fuzzy sets for each system variable is fairly typical.
- Overlap adjacent fuzzy sets to ensure that no crisp value fails to correspond to any set, and to help ensure that more than one rule is involved in determining the output. System variables with 5 or 7 with 15 to 20 % overlap of adjacent fuzzy sets tend to work fairly well.

3.2.5 Fuzzy logic operators

Fuzzy logical reasoning is a superset of standard Boolean logic. If we keep the fuzzy values at their extremes of 1 (completely true), and 0 (completely false), standard logical operations will hold.

Intersection (AND) denoted by min or product

$$\mu_{A\cap B}(x) = \min(\mu_A(x), \mu_B(x))$$

Union (OR) by max /algebraic sum

$$\mu_{A\cup B}(x) = \max(\mu_A(x), \mu_B(x))$$

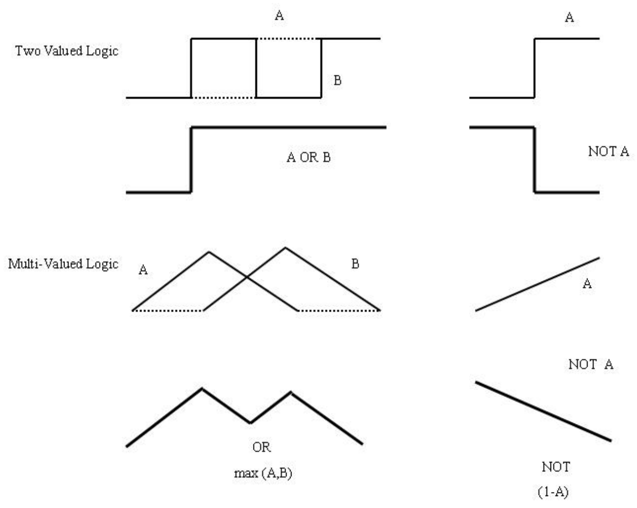


Figure 3.2 Fuzzy Logic Operators

3.2.6 Linguistic variables and rule bases

Linguistic variables are values defined by fuzzy sets. A linguistic variable such as 'High Speeds' for example could consist of numbers that are equal to or between 50km/hr and 80km/hr. The conditional statements that make up the rules govern fuzzy logic behavior using these linguistic variables and have an *if-then* syntax. These if-then rules are what make up fuzzy rule bases. A sample if-then rule where A and B represent linguistic variables could be:

if x is A then y is B

The statement is understood to have both a premise, if '*x is A*', and a conclusion, then '*y is B'*. The premise also known as the *antecedent* returns a single number between 0 and 1 whereas the conclusion also known as the *consequent* assigns the fuzzy set B to the output variable y.

Interpreting these rules involves a number of distinct steps.

1. *Fuzzify the inputs*. To do this all fuzzy statements in the premise are resolved to a degree of membership between 0 and 1. This can be thought of as the degree of support for the rule. At a working level this means that if the antecedent is true to some degree of membership, then the consequent is also true to that same degree.

2. *Apply the fuzzy operators:* For antecedents with multiple parts to yield a single number between 0 and 1. Again this is the degree of support for the rule.

3. *Apply the result to the consequent***.** This step is also known as *implication*. The degree of support for the entire rule is used to shape the output fuzzy set. The outputs of fuzzy sets from each rule are aggregated into a single output fuzzy set. This final set is evaluated (or *defuzzified*) to yield a single number, using the defuzzification methods.

3.3 Fuzzy Relation and Composition

A relation can be considered as a set of ordered pairs (tuples). In the same way a fuzzy relation is a fuzzy set of tuples i.e. each tuple has a membership function between 0 and 1. Fuzzy relation allows a partial membership. Since a relation can be viewed as a set we can easily generalize the classical notion of using fuzzy set .Classical binary relation can be represent relation R on x, y on domain X, Y as a function that maps an ordered pair (x, y) in X×Y to 0 if the relation does not hold between x and y or 1 if relation holds i.e. R= X×Y→{0 ,1}. Fuzzy relation generalizes the classical idea of relation in to a matter of degree. Therefore, fuzzy relation R between variables x and y whose universe of discourse on X,Y is defined by a function that maps ordered pair in X×Y to their degree in the relation, which is numbered between 0 and 1 i.e. R= X×Y→[0 ,1].

For continuous universes it is denoted by

$$R = \int_{X\times Y} \mu_R(x,y) \Big/ (x,y)$$

For discrete Universe it is denoted by

$$R = \sum_{X\times Y} \mu_R(x,y) \Big/ (x,y)$$

A fuzzy relation on X×Y can also be denoted as

$$R = \{(x,y),\ \mu_R(x,y)\} \quad (x,y) \in X \times Y,\ \ \mu_R(x,y) \in [0\ 1]$$

μ_R (x,y) gives the degree of membership of the ordered pair (x,y) in R associating with each pair (x,y) in X×Y in interval [0 1]. The degree of membership indicates the degree to which X is in relation with Y. If A and B are fuzzy sets in the universe of X×Y the fuzzy relation then has membership degree.

$$\mu_R(x,y) = \mu_{A\times B}(x,y) = \min[\mu_A(x), \mu_B(y)] \quad x \in X \text{ and } y \in Y$$

3.3.1 Projection

This operation brings a ternary relation back to a binary relation, or a binary relation to a

Fuzzy set or a fuzzy set to a single crisp value. In binary case it is simple. Let R be defined on X×Y then projection is defined as:

$$\mathrm{Pr}\,oj\ R\ on\ Y = \int_{Y} \frac{(\max \mu_R(x, y))}{y}$$

This is simply done by taking maximum from each column from X ×Y, the same is true for proj R on X but from the row. The projection operation is almost always used in combination with cylindrical extension.

3.3.2 Cylindrical extension

The cylindrical extension is more or less opposite of projection**.** It extends fuzzy set to fuzzy binary relation, fuzzy binary relation to fuzzy ternary relation, etc. it mainly serves the following goal: let A be fuzzy set defined on X and R be a fuzzy relation defined on X ×Y, then it is, of course, not possible to take the intersection of A and R (since X is sub space of X ×Y) but when A is extended to X ×Y this is possible. This extension is done by cylindrical extension operator

Cylindrical extension of S into U is denoted by

$$ce(S) = \int_U \mu_S(x_i, \ldots, x_{ik}) \Big/ (x, \ldots, n)$$

3.4 Types of Fuzzy Systems

There exist two main types of fuzzy systems that differ in the way they define the consequents of their rules: Mamdani and Takagi-Sugeno-Kang fuzzy systems.

3.4.1 Mamdani fuzzy systems.

Mamdani fuzzy systems have fuzzy sets as rule consequents. A typical Mamdani type fuzzy controller is realized using the following steps.

- ***Fuzzification***.

  It is the process of transformation of crisp measurement into corresponding fuzzy set . The value between 0 and 1 each input is given represents the degree of membership or the degree of belongingness. For figure 3.2 the fuzzy sets can be written as:

$$\mu_{Low} = \{(0, 1), (3,1), (4,0.8), (5,0.5), (7,0)\}$$

$$\mu_{High} = \{(0,0), (3,0), (4,0.3), (5,0.5), (7,1)\}$$

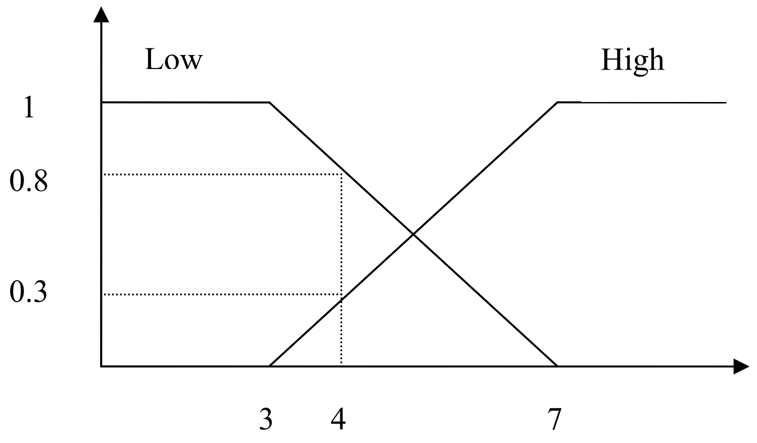


Figure 3.3 Input membership functions for level

- **Application of fuzzy logic operators**

The rule-base is the key factor in fuzzy logic technique. If there are multiple parts to the antecedent of a rule, *apply fuzzy logic operators* and resolve the antecedent to a single number between 0 and 1 that represents the strength of that rule. This is the degree of support for the rule. Use the degree of support for the entire rule to shape the output fuzzy set.

- **Apply implication method**

The consequent of a fuzzy rule assigns an entire fuzzy set to the output. This fuzzy set is represented by a membership function that is chosen to indicate the qualities of the consequent. If the antecedent is only partially true, (i.e., is assigned a value less than 1), then the output fuzzy set is truncated according to the *implication method.* In general, one rule by itself does not do much good. What is needed are two or more rules that can play off one another.

***Example*:** We desire to control the acceleration of a moving object, in order to reach a goal position. Let the input measurements available in order to decide acceleration values at every time instant are the distance of the moving object from the target position and its velocity. One possible rule in the rule-base of a fuzzy logic system could be:

IF *distance* is *big* AND *velocity* is *small* THEN *acceleration* is *big*.

Here the implication method being AND i.e. MIN, hence 0.6 is taken and output membership function is truncated.

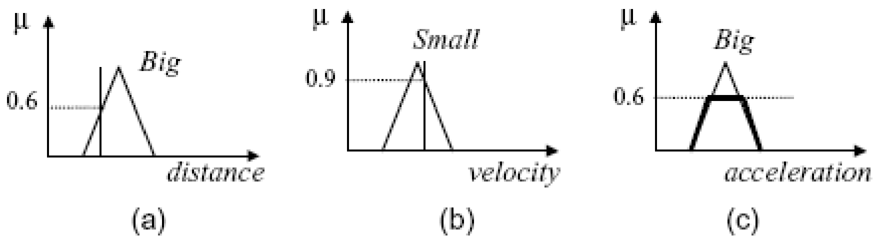


Figure 3.4 Implication method-AND

❖ **Aggregate all outputs**

The output of each rule is a fuzzy set. The fuzzy outputs of each rule need to be combined in a meaningful way to be of any use. *Aggregation* is the method used to perform this by combining each output set into a single output fuzzy set. The order of rules in the aggregation operation is unimportant as all rules are considered. The three methods of aggregation available for use include *sum* (sum of each rules output set), *max* (maximum value of each rule output set) and the *probabilistic OR* method (the algebraic sum of each rules output set).

❖ **Defuzzification**

At the output of the fuzzy inference there will always be a fuzzy set that is obtained by the composition of the fuzzy sets output by each of the rules .In order to be used in the real world, the fuzzy output needs to be interfaced to the *crisp* domain by the defuzzifier. The output fuzzy set indicates what the output is in fuzzy terms. This fuzzy output will be a membership function that provides the degree of membership of several possible crisp outputs. Thus, the point corresponding to the highest degree of membership in the fuzzy output has to be sought. This operation would correspond to a type of defuzzification, called *max* defuzzification. Unfortunately, in most practical cases the situation is not so simple, since there might be many points having the same maximum degree of membership in the fuzzy output, and an indecision on which one of these points to choose arises. Moreover, choosing the maximum point of the membership function is an operation that discards most of the information contained in the membership function itself. There is the need for a technique that *summarizes* the information contained in the membership function [12]. A number of methods of defuzzification are possible and these include the mean of maximum, smallest of maximum and centroid (centre of area)methods.

Given an output fuzzy set $Y = \mu_Y(v)$ defined in the universe V of the variable v, the defuzzified output y for *center of gravity method* is given by the expressions:

$$y_d = \frac{\int_S y\mu_Y(y)dy}{\int_S \mu_Y(y)dy}$$

S is the support of $\mu_Y(y)$. One drawback of this kind of defuzzification is the complexity involved with finding the center of gravity (i.e., integration).

3.4.2 Takagi-Sugeno-Kang (TSK) fuzzy systems.

The Sugeno fuzzy model, or more fully the Takagi-Sugeno-Kang method of fuzzy inference was first introduced in 1985. In many respects it is identical to the Mamdani method except that the output membership functions for the Sugeno method are always linear or constant. The output membership functions can be thought of as singleton spikes that undergo a simple aggregation instead of other aggregation methods such as *max*, *probor* or *sum*. A typical rule in a Sugeno fuzzy model has the form

If Input 1 = x and Input 2 = y, then Output is z = ax + by + c

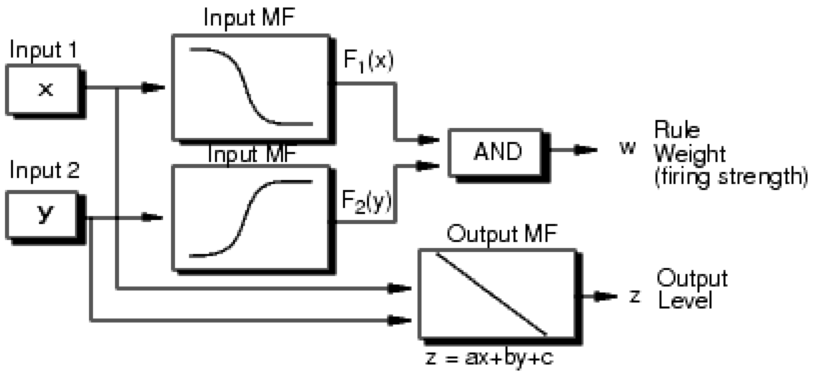


Figure 3.5 Sugeno Rule inference

For a zero-order Sugeno model ,the output level z is a constant (a = b =0).The output level z_i of each rule is weighted by the firing strength w_i of the rule. For example, for an AND rule with Input 1 = x and Input 2 = y, the firing strength is w_i = And Method (F1(x)

, F2(y)) where F1,2 (.) are the membership functions for Inputs 1 and 2 [29]. The final output of the system is the weighted average of all rule outputs, computed as:

$$Final\ \text{Output} = \frac{\sum_{i=1}^{N} w_i z_i}{\sum_{i=1}^{N} w_i}$$

Due to their functional description, TSK-type systems usually exhibit greater accuracy than Mamdani systems. However, the interpretability of their rules is significantly reduced as they no longer represent linguistic concepts and hence not intuitive. The Sugeno system is computationally efficient and its ability to interpolate multiple linear models makes it particularly suited to modeling non-linear systems. The Sugeno model has shown itself to have better potential in controlling non-linear systems that require more stringent output membership functions.

3.5 Fuzzy Logic in Control problems.

Fuzzy Logic offers several unique features that make it a particularly good choice for many control problems.

- It is inherently robust since it does not require precise, noise-free inputs and can be programmed to fail safely if a feedback sensor quits or is destroyed. The output control is a smooth control function despite a wide range of input variations.
- Since the Fuzzy Logic controller processes user-defined rules governing the target control system, it can be modified and tweaked easily to improve or drastically alter system performance. New sensors can easily be incorporated into the system simply by generating appropriate governing rules.
- Fuzzy Logic is not limited to a few feedback inputs and one or two control outputs, nor is it necessary to measure or compute rate-of-change parameters in order for it to be implemented. Any sensor data that provides some indication of a system's actions and reactions is sufficient. This allows the sensors to be inexpensive and imprecise thus keeping the overall system cost and complexity low.
- Because of the rule-based operation, any reasonable number of inputs can be processed (1-8 or more) and numerous outputs (1-4 or more) generated, although defining the rule base quickly becomes complex if too many inputs and outputs

are chosen for a single implementation since rules defining their interrelations must also be defined. It would be better to break the control system into smaller chunks and use several smaller Fuzzy Logic controllers distributed on the system, each with more limited responsibilities.

- Fuzzy Logic can control nonlinear systems that would be difficult or impossible to model mathematically. This opens doors for control systems that would normally be deemed unfeasible for automation.

3.6 CONCLUSION

Fuzzy logic fundamentals have been introduced. Fuzzy modeling techniques are explained and a detailed fuzzy system working has been presented.

CHAPTER IV

GENETIC ALGORITHM

4.1 Introduction

Genetic algorithms are computerized search and optimization algorithms based on the mechanics of natural genetics and natural selection. Essentially, Genetic Algorithm is an optimization technique that performs parallel, stochastic, but direct search method to evolve the fittest population.

The genetic algorithm is a method for solving both constrained and unconstrained optimization problems. According to Goldberg, the simulated evolution of a solution through genetic algorithms is, in some cases, more efficient and robust than the random search, enumerative or calculus based techniques. The main reasons given by Goldberg are the probability of a multi-modal problem state space in non-linear problems, and that random or enumerative searches are exhaustive if the dimensions of the state space are too great [6].

Genetic algorithms are by far the most popular evolutionary technique. This is due in part to their conceptual simplicity, the ease of programming entailed, and the small number of parameters to be defined (apart from the genomic representation and the fitness function, parameters include mainly population size, crossover and mutation probabilities, and termination condition). One can apply the genetic algorithm to solve a variety of optimization problems that are not well suited for standard optimization algorithms, including problems in which the objective function is discontinuous, non-differentiable, stochastic, or highly nonlinear.

The idea is to have a pool of candidate solutions evaluated in parallel, from which the “fittest" solutions are chosen to mate and breed new candidate solutions using genetic operators. This procedure is iterated until the population converges or a preset condition is met. Over successive generations, the population "evolves" toward an optimal solution.

4.1.1 Evolution theory

Human cells are made up of chromosomes. Chromosomes are strings of DNA (deoxyribonucleic acid). A chromosome consists of genes, blocks of DNA. Each gene encodes a particular protein. Basically, it can be said that each gene encodes a trait, for example color of eyes. Possible settings for a trait (e.g. blue, brown) are called alleles. Each gene has its own position in the chromosome. This position is called locus. Complete set of genetic material (all chromosomes) is called genome. Thus a chromosome (solution) is composed of several genes (variables).

Particular set of genes in genome is called *genotype*. The genotype is with later development after birth base for the organism's *phenotype*, its physical and mental characteristics, such as eye color, intelligence etc. During reproduction, *recombination* (or *crossover*) first occurs. Genes from parents combine to form a whole new chromosome. The newly created offspring can then be mutated. *Mutation* means that the elements of DNA are a bit changed. These changes are mainly caused by errors in copying genes from parents. The *fitness* of an organism is measured by success of the organism in its life (survival). In principle, a Genetic Algorithm can be applied to any problem where the variables to be optimized (“genes”) can be encoded to form a string (“chromosome”).

4.2 Classical Vs Genetic Algorithm

- ❖ Genetic Algorithm works with coding of parameter not parameter itself.
- ❖ Seeks optimal solution by searching population of points of the search space in parallel not single point.
- ❖ Uses only value of objective function no derivative or other information is necessary.
- ❖ Uses stochastic (probabilistic) transition not deterministic rule in optimization procedure.
- ❖ It is robust – eliminates costly redesigning, adaptation to the environment – existing system can perform their function longer and better.

4.3 Strengths of Genetic Algorithm

- ❖ It evaluates many possible solutions simultaneously. Hence it is intrinsically parallel. Most other algorithms are serial and can only explore the solution space to a problem in one direction at a time, and if the solution they discover turns out to be suboptimal, there is nothing to do but abandon all work previously completed and start over. However, since Genetic Algorithms have multiple offspring, they can explore the solution space in multiple directions at once. If one path turns out to be a dead end, they can easily eliminate it and continue work on more promising avenues, giving them a greater chance each run of finding the optimal solution.

- Genetic Algorithms have the quality of robustness, that is, while special case algorithms may find more optimal solutions to specific problems, genetic algorithms perform very well over a large number of problem categories. Because of this, Genetic Algorithms are not caught by local minima. Genetic Algorithms are stochastic and less likely to get trapped in local optima, which inevitably are present in any practical optimization applications.

- Genetic Algorithms also perform well on problems whose complexity increases exponentially with the number of input parameters, since these types of problems are extremely inefficient to solve using traditional approaches.

4.4 Weakness of Genetic Algorithm

- Although Genetic Algorithms are easy to implement and are powerful tools to solve difficult problems featuring complex search spaces, they usually require human supervision to be exploited successfully.
- There are practical limit on the hypothetically unlimited number of iteration or generation and population size in genetic algorithm.
- The key disadvantage of the Genetic Algorithms is that its convergence speed near the global optimum becomes slow.

4.5 Basic Working Principle

A Genetic Algorithm is an iterative procedure that involves a population of individuals, each one represented by a finite string of symbols, known as the genome, encoding a possible solution in a given problem space. A general outline of the algorithm is described below. This can yield solution to the problem, whatever the form of objective function. The search process for the solution of an optimization problem, in the form of a pseudo code, is summarized as below.

4.5.1 A General Outline of Genetic Algorithm

- An initial population of individuals is generated at random or heuristically.

- Every evolutionary step, known as a generation, the individuals in the current population are decoded and evaluated according to some predefined quality criterion, referred to as the fitness, or fitness function. To form a new population (the next generation), individuals are selected according to their fitness.
- New individuals are introduced into the population, (i.e. to find new points in the search space) by genetically inspired operators- Crossover and mutation.
    - Crossover is performed with probability p_c (the 'crossover probability' or 'crossover rate') between two selected individuals, called *parents*, by exchanging parts of their genomes (i.e. encodings) to form two new individuals, called *offspring*.
    - The mutation operator is introduced to prevent premature convergence to local optima by randomly sampling new points in the search space. It is carried out by flipping bits at random, with some (usually small) probability p_m.
- Replaces the current population with the children to form the next generation.
- The algorithm stops when one of the stopping criteria is met.
    - Runs prescribed number of generation.
    - If acceptable solution has been found.
    - There is no improvement in fitness value.

4.5.2 The Pseudo Code

```
Begin GA
    g=0;
    Initialize population P (g)
    Evaluate population P (g): Compute fitness value
while not done
    g = g+1
    Select P (g) from P (g-1)
    Crossover P (g)
    Mutate P (g)
    Evaluate P (g)
```

```
end while
end GA
```

4.6 Development of Genetic Algorithm

4.6.1 Encoding Schemes

The first decision to take when applying such an algorithm is how to encode candidate solutions within the genome. The representation must allow for the encoding of all possible solutions while being sufficiently simple to be searched in a reasonable amount of time. So that the genetic algorithm may converge to good solutions, the representation must be carefully designed to minimize redundancy. (i.e. several genotypes encoding the same phenotype) and to avoid invalid representation (i.e. a genotype encoding a phenotype which is not a possible solution to the problem in hand) .Encoding transforms points in parameter space into bit string representation. Binary encoding is the most common one, mainly because the first research of Genetic Algorithm used this type of encoding and because of its relative simplicity. In binary encoding, every chromosome is a string of bits {0, 1}.

For instance, a point (11, 5, 8) in a three-dimensional parameter space can be represented as a concatenated binary string 1011 0101 1000 in which each coordinate value is encoded as a "gene" composed of four binary bits using binary coding.

If each variable X_i, with real values, is coded as a binary string of length l_i, then the relation between the initial value and coding information is;

$$X_i = x_i^{(L)} + \frac{x_i^{(U)} - x_i^{(L)}}{2^{l_i} - 1} \times \text{decoded value}(s_i) \qquad \text{...... (4.1)}$$

where the variable X_i can take the value from a domain $D_i = [x_i\ (L), x_i\ (U)]$ and is coded in substring S_i of length l. The decoded value of a binary substrings S_i is calculated as

$$S_i = \sum_{i=0}^{l-1} 2^i s_i$$

where $S_i \in (0,1)$

Generalizing this concept, we may say that with a l_i–bit coding for a variables, the obtainable accuracy in that variable is approximately $\frac{x_i^{(U)} - x_i^{(L)}}{2^{l_i}}$. Once the coding of the variables has been done, the corresponding point $x = (x_1, x_2, \ldots\ldots x_N)^T$ can be found using equation (4.1). Thereafter, the function value at the point 'x' can also be calculated by substituting x in the given objective function f(x).

4.6.2 Fitness Function

The first step after creating a generation is to calculate the fitness value of the each member in the population. The fitness/objective function is chosen depending on the problem in hand such that the individuals having high fitness values are the good solution candidates for the optimization Therefore, reproduction of the next generation will strictly be dependent on the fitness measure.

In general fitness function is *F(x)* is first derived from objective function and used in successive genetic generations. For maximization problems the fitness function can be considered to be the same as objective function or *F(x) =f(x).* For minimization problems, the fitness function is an equivalent maximization problem chosen in such a way that the optimum point remains unchanged.

A number of such transformations are possible. The following fitness function is often used; where *f*(x) is the objective function

$$F(x) = \frac{1}{1 + f(x)} \qquad or \qquad F(x) = -f(x)$$

This transformation does not alter the location of optimum point but converts maximization problem in to equivalent minimization problem. The fitness function value is called string fitness for survival.

Fitness is an important concept for the operation of the Genetic Algorithm. The fitness of a string is a measure of the quality of the trial solution represented by the string with respect to the function being optimized. Thus, high fitness corresponds to a high value (in a maximization problem) or a low value (in a minimization problem) of the function. Fitness is important in determining the likelihood of an individual taking part in crossover and also in deciding which individuals will survive into the next generation.

4.6.3 Selection

After evaluation, a new population is created from the current generation. Selection operation determines which parents participate in producing offspring for the next generation, and it is analogous to "survival of the fittest" in natural selection. There are many different techniques which a genetic algorithm can use to select the individuals to be copied over into the next generation, but listed below are some of the most common methods.

4.6.3.1 Elitist selection

The fit members of each generation are guaranteed to be selected. (Most Genetic Algorithms do not use pure elitism, but instead use a modified form where the single best or a few of the best, individuals from each generation are copied into the next generation just in case nothing better turns up).

4.6.3.2 Fitness-proportionate selection

Individuals are selected with a probability proportional to their relative fitness. More fit individuals are more likely, but not certain, to be selected. This ensures that the expected number of times an individual is chosen is approximately proportional to its relative performance in the population. Thus, high-fitness (‘good’) individuals stand a better chance of ‘reproducing’, while low-fitness ones are more likely to disappear

4.6.3.3 Roulette-wheel selection

A form of fitness-proportionate selection in which the chance of an individual's being selected is proportional to the amount by which its fitness is greater or less than its competitor’s fitness. (Conceptually, this can be represented as a game of roulette - each individual gets a slice of the wheel, but more fit ones get larger slices than less fit ones. The wheel is then spun, and whichever individual "owns" the section on which it lands each time is chosen.)

4.6.3.4 Tournament selection

Subgroups of individuals are chosen from the larger population, and members of each subgroup compete against each other. Only one individual from each subgroup is chosen to reproduce.

4.6.3.5 Rank selection

Each individual in the population is assigned a numerical rank based on fitness, and selection is based on these ranking rather than absolute differences in fitness. The advantage of this method is that it can prevent very fit individuals from gaining dominance early at the expense of less fit ones, which would reduce the population's genetic diversity and might hinder attempts to find an acceptable solution.

4.6.3.6 Steady-state selection

The offspring of the individuals selected from each generation go back into the pre-existing gene pool, replacing some of the less fit members of the previous generation. Some individuals are retained between generations.

4.6.5 Crossover

To exploit the potential of the current population, the crossover operator generates new chromosomes. Crossover is usually applied to selected pairs of parents with a probability equal to a given crossover rate. Crossover techniques are described below:

- ❖ One point crossover is the most basic crossover operator, where a crossover point on the genetic code is selected at random and two parent chromosomes are interchanged at this point as shown in figure 4.1
- ❖ In two-point crossover, two crossover points are selected and the part of the chromosome string between these two points (P1 , P2) is then swapped to generate two children as shown in figure 4.2

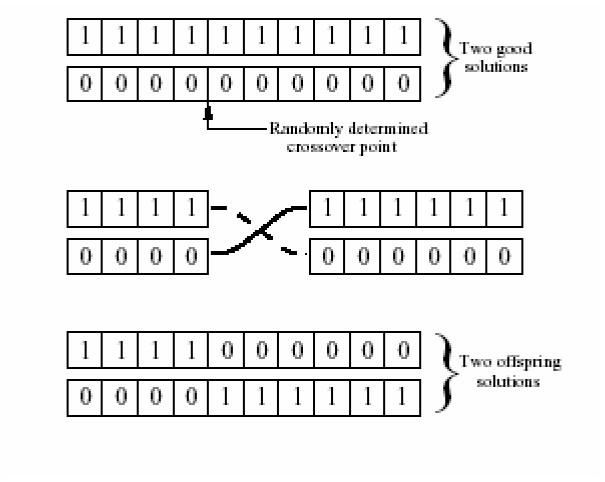


Figure 4.1 One point Crossover

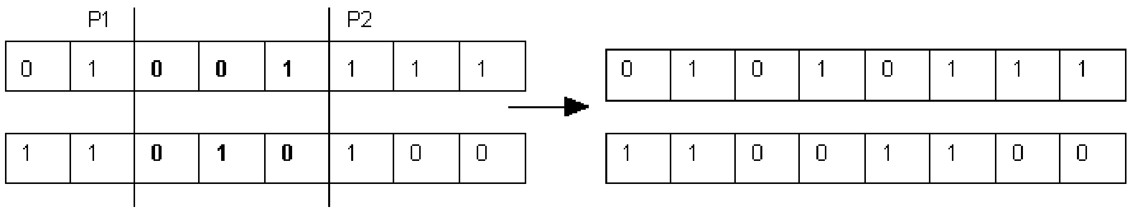


Figure 4.2 Two point Crossover

4.6.6 Mutation

While crossover exploits current gene potentials, mutation operators are capable of spontaneously generating new chromosomes. A mutation operator can prevent any single bit from converging to a value throughout the entire population, and more important, it can prevent the population from converging and stagnating at any local optima. For example, the string 1011, when the mutation operator applied to the last bit of the string, becomes 1010. A similar example is depicted in figure 4.3. Mutations are needed in Genetic Algorithms because while reproduction and crossover explore the search space well, occasionally they lose information that is essential to the solution. Moreover the Genetic Algorithm may converge on sub-optimum strings due to a bad choice of initial population. It is for this reason mutation is used to explore the search space as efficiently as possible. Mutation is the occasional random alteration of a value of a string position. Mutation is applied randomly to an entire population and with low probability (0.001→0.1).

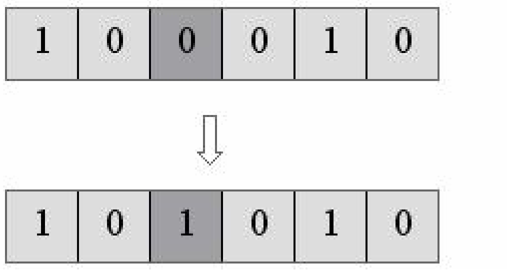


Figure 4.3 Mutation

4.7 Effect of Genetic Operators

4.7.1 Crossover Probability

The crossover operator determines the rate of convergence and how often crossover will be performed. This operator tends to enable the evolutionary process to move toward ‘promising’ regions of the search space. If there is no crossover, offspring are exact copies of parents. If there is crossover, offspring are made from parts of both parent's chromosome. If crossover probability is *100%,* then all offspring are made by crossover. If it is *0%,* whole new generation is made from exact copies of chromosomes from old population Crossover is made in hope that new chromosomes will contain good parts of old chromosomes and therefore the new chromosomes will be better.

4.7.2 Mutation Probability

Using mutation alone induces a random walk through the search space. The probability of mutation is normally low because a high mutation rate would destroy fit strings and degenerate the genetic algorithm into a random search, and it becomes difficult to quickly converge to the global optimum. Mutation probability values of around 0.1% or 0.01% are common, these values represent the probability that a certain string will be selected for mutation i.e. for a probability of 0.1%; one string in one thousand will be selected for mutation. If there is no mutation, offspring are generated immediately after crossover (or directly copied) without any change. If mutation is performed, one or more parts of a chromosome are changed. If mutation probability is *100%,* whole chromosome is changed, if it is *0%*, nothing is changed. Mutation generally prevents the Genetic Algorithm from falling into local extremes.

4.7.3 Population Size

It is the number of chromosomes in population (in one generation). Population size is a factor that affects the Genetic Algorithm performance. Increasing the population means a longer computation time, while if the population size is decreased; the accuracy of the solution is also decreased because of reduced variation of chromosomes. If there are too few chromosomes, Genetic Algorithm has few possibilities to perform crossover and only a small part of search space is explored. On the other hand, if there are too many chromosomes, Genetic Algorithm slows down. Research shows that after some limit (which depends mainly on encoding and the problem) it is not useful to use very large populations because it does not solve the problem faster than moderate sized populations.

In Genetic Algorithm design there must be a balance between generation numbers and population size. For e.g. a population of 100 chromosomes can reach a solution in 10 generations. However, the solution in a population of 20 chromosomes and 20 generations can take four times shorter time than the first one. Population size also reduces the effect of highest fitness valued chromosomes. For e.g. in a population of 10 chromosomes, if one of the chromosomes has fitness value of 9 while others have value of 1, half of the parents chosen from among the relatively low fitness valued chromosomes though the best fitness valued chromosome is nine times better.

4.8 CONCLUSION

The genetic algorithm optimizing technique has been discussed in detail. The algorithm outline has been dealt and each of its components has been explained. The effect of its parameters on the algorithm functioning has also been discussed.

CHAPTER V

EVOLUTIONARY FUZZY MODELING

The domain of evolutionary computation involves the study of the foundations and the applications of computational techniques based on the principles of natural evolution. Evolutionary algorithms are used to search large, and often complex, search spaces. They have proven worthwhile on numerous diverse problems; able to find near-optimal solutions given an adequate performance (fitness) measure .Genetic algorithm is a well known evolutionary algorithm. Fuzzy modeling can be considered as an optimization process where part or all of the parameters of a fuzzy system constitute the search space. Evolutionary fuzzy modeling has since been applied to an ever-growing number of domains, branching into areas as diverse as chemistry, medicine, telecommunications, biology, and geophysics. When designing a fuzzy system using a Genetic Algorithm, the first important consideration is the representation strategy that is how to encode the fuzzy system into the chromosome. To completely represent a fuzzy system, each chromosome must encode all the needed information about the rule set and the membership function parameters.

Below one can classify evolutionary fuzzy modeling techniques based on three types of parameters which compose the search space: structural, connective, and operational.

5.1. Applying Evolution to Fuzzy Modeling

Depending on several criteria—including the available a priori knowledge about the system, the size of the parameter set, and the availability and completeness of input: output data—artificial evolution can be applied in different stages of the fuzzy parameters search. From the viewpoint of optimization, the task of finding an appropriate knowledge base for a particular problem, is equivalent to parameterize the fuzzy knowledge base (rules and membership functions), and to find those parameter values that are optimal with respect to the design criteria. The knowledge base parameters constitute the optimization space, which is transformed into a suitable genetic representation on which the search process operates.

First of all, it is important to distinguish between tuning (alternatively, adaptation) and learning problems:

• ***Tuning*** is concerned with optimization of an existing Fuzzy Rule Base System. Tuning processes assume a predefined rule base and have the objective to find a set of optimal parameters for the membership and/or the scaling functions, data base parameters.

• ***Learning*** constitutes an automated design method for fuzzy rule sets that starts from scratch. The processes perform a more elaborated search in the space of possible rule bases or whole knowledge base and do not depend on a predefined set of rules. A *genetic learning* process faces a much more difficult task as it has to establish the proper relationship between input and output states from scratch, rather than optimizing the performance of a fuzzy system that already operates at least approximately correct.

Three of the four types of fuzzy parameters in Table 3.1 can be used to define targets for evolutionary fuzzy modeling: structural parameters, connective parameters, and operational parameters. Logical parameters are usually predefined by the designer based on experience.

5.1.1 Knowledge Tuning (Operational Parameters)

The evolutionary algorithm is used to tune the knowledge contained in the fuzzy system by finding membership function values. An initial fuzzy system is defined by an expert. Then, the membership function values are encoded in a genome, and an evolutionary algorithm is used to find systems with high performance. Evolution often overcomes the local-minima problem present in gradient descent-based methods.

Genetic Algorithms are applied to modify the membership functions. When modifying the membership functions, these functions are parameterized with one to four coefficients (Figure 5.1), and each of these coefficients will constitute *a gene of the chromosome for the Genetic Algorithm*. The basic idea is to represent the complete set of membership functions by an individual and to evolve shape and location of the triangles. Each triangle may be described by its anchor points on the abscissa axis, and the gaussian membership functions are characterized by center c and sigma σ.

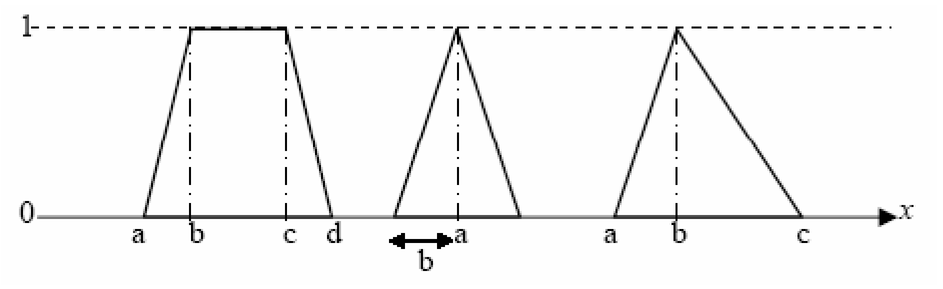


Figure: 5.1 some parameterized membership functions

Using as example the WBCD (Wisconsin Breast Cancer Diagnosis) problem, an initial fuzzy rule base is defined by an expert. An example fuzzy rule in this case would be:

if (input$_2$ is Low) and (input$_6$ is Low) then (output is benign).

The evolutionary algorithm then fine-tunes the membership functions, i.e. the membership function parameter values .One of the major shortcomings of knowledge tuning is its dependency on the initial setting of the knowledge base.

5.1.2 Behavior Learning (Connective Parameters)

In this approach, one supposes that extant knowledge is sufficient in order to define the membership functions; this determines, in fact, the maximum number of rules [27]. The genetic algorithm is used to find either the rule consequents, or an adequate subset of rules to be included in the rule base. As the membership functions are fixed and predefined, this approach lacks the flexibility to modify substantially the system behavior. Furthermore, as the number of variables and membership functions increases, the curse of dimensionality becomes more pronounced and the interpretability of the system decreases rapidly.

5.1.3 Structure Learning (Structural Parameters)

In many cases, the available information about the system is composed almost exclusively of input: output data and specific knowledge about the system structure is scant. In such a case, evolution has to deal with the simultaneous design of rules, membership functions, and structural parameters. Some methods use a fixed-length genome encoding a fixed number of fuzzy rules along with the membership function values. In this case the designer defines some structural constraints according to the available knowledge of the problem characteristics. Other methods use variable-length genomes to allow evolution to discover the optimal size of the rule base. In this thesis, evolutionary structure learning is carried out by encoding within the genome an entire fuzzy system (this is known as the Pittsburgh approach).

Structure learning permits to specify other criteria related to the interpretability of the system, such as the number of membership functions and the number of rules. On the other hand, the strong interdependency among the parameters involved in this form of learning may slow down, or even prevent altogether, the convergence of the genetic algorithm.

5.2 Learning with Genetic Algorithm

Both connective and structural parameters modeling can be viewed as rule base learning processes with different levels of complexity. They can thus be assimilated within other methods from machine learning, taking advantage of experience gained in this latter domain. In the evolutionary algorithm community there are two major approaches for evolving such rule systems: the Michigan approach and the Pittsburgh approach. The three approaches are presented below.

5.2.1 Michigan Approach

Each individual represents a *single* rule. The fuzzy inference system is represented by the entire population. Since several rules participate in the inference process, the rules are in constant competition for the best action to be proposed, and cooperate to form an efficient fuzzy system. The cooperative–competitive nature of this approach renders difficult the decision of which rules are ultimately responsible for good system behavior. It necessitates an effective credit assignment policy to ascribe fitness values to individual rules. The method is suitable for on-line learning tasks as the evolutionary algorithm incrementally improves the performance of the fuzzy controller constituted by the population of rules. In *Michigan approach,* data base and rule base are two clearly separate entities. All rules share the same membership functions defined in a common data base.

5.2.2 Pittsburgh Approach

Here, the evolutionary algorithm maintains a population of candidate fuzzy systems, each individual representing an entire fuzzy system. In this approach each chromosome encodes a whole rule base or knowledge base. Selection and genetic operators are used to produce new generations of fuzzy systems. Since evaluation is applied to the entire system, the credit assignment problem is eschewed. This approach allows including of additional optimization criteria in the fitness function, thus affording the implementation of multi-objective optimization. In a *Pittsburg* approach fuzzy system, each rule operates with its own fuzzy sets. Data base and rule base merge as the rule itself contains the parameters of the underlying membership functions. Such a representation possesses more degrees of freedom which permits a more accurate approximation of the desired input-output relationship, hence the term approximate.

The price of increased accuracy is that the resulting fuzzy system becomes more difficult to analyze as an individual fuzzy set no longer coincides with a linguistic concept. The main shortcoming of this approach is its computational cost, since a population of full-fledged fuzzy systems has to be evaluated each generation [22].

5.2.3. Iterative Rule Learning Approach

As in the Michigan approach, each individual encodes a single rule. An evolutionary algorithm is used to find a single rule, thus providing a partial solution. The evolutionary algorithm is used iteratively for the discovery of new rules, until an appropriate rule base is built. To prevent the process from finding redundant rules (i.e. rules with similar antecedents), a penalization scheme is applied each time a new rule is added.

This approach combines the speed of the Michigan approach with the simplicity of fitness evaluation of the Pittsburgh approach. However, as with other incremental rule base construction methods, it can lead to a non-optimal partitioning of the antecedent space.

5.3 Evolutionary Computation to Solve Medical Problems

Most medical decisions can be formulated as a search in some appropriate space. For example, a pathologist analyzing biopsies to decide whether they are malignant or not, is searching in the space of all possible cell features for a set of features permitting him to provide clear diagnosis. A radiologist planning a sequence of radiation doses is searching for the best treatment in the space of all possible treatments [15]. Medical search spaces are usually very large and complex. Medical decisions are based on clinical tests which provide huge amounts of data. Based on these data one must ultimately make a single decision (e.g. malignant or benign). Given the tight interdependency among the domain variables, and the inherent non-linearity of most real-world problems, neighboring points in the search space may have widely differing qualities, turning the search into a complex task. Evolutionary computation provides powerful techniques for searching such complex spaces.

The construction of accurate models of medical decision from extant knowledge is a hard task. On one hand, the models involve too many non-linear and uncertain parameters to be treated analytically. On other hand, medical experts are usually not available, or simply do not collaborate in translating their experience into a usable decision tool. Evolutionary computation is applied in medicine to perform several types of tasks. Whenever a decision is required in medicine, it is usually possible to fine a niche for evolutionary techniques. The tasks performed by evolutionary algorithms in the medical domain can be divided into two groups:

1. Data mining mainly applied to diagnosis and prognosis.

2. Medical imaging and signal processing.

Evolutionary algorithms (EA) for discovering fuzzy classification rules can be divided into three categories.

- ***Fixed membership functions***: This approach has Evolutionary algorithm search for good combinations of attribute values, which will compose fuzzy rules. However, the membership functions of the attribute values are predefined (either manually or by another algorithm), rather than being evolved by the Evolutionary algorithms.
- ***Fixed rules***: Evolutionary algorithms are used for tuning the membership functions associated with attributes being fuzzified. This approach is typically used when crisp rules have already been discovered by another algorithm, and we just want to use an Evolutionary algorithm to fuzzify the discovered crisp rules.
- ***Evolutionary algorithms for both generating fuzzy rules and tuning membership functions.*** This approach has Evolutionary algorithm to optimize both the contents of fuzzy rules and the membership functions of the linguistic values of the attributes being fuzzified.

5.4 CONCLUSION

The concept of evolutionary fuzzy modeling and its different techniques has been explained. The different approaches for evolutionary rule based systems have been discussed and it's found that evolutionary computation is a good tool for medical diagnosis and thus being widely used.

CHAPTER VI

MEDICAL DIAGNOSIS USING FUZZYGENETICS

6.1 Breast Cancer Diagnosis

Breast cancer is the most common cancer among women, excluding skin cancer. The presence of a breast mass is an alert sign, but it does not always indicate a malignant cancer. The Wisconsin Breast Cancer Diagnosis (WBCD) problem, involves classifying a presented case of putative cancer as to whether it is benign or malignant. Fine needle aspiration (FNA) of breast masses is a cost-effective, non-traumatic, and mostly non-invasive diagnostic test that obtains information needed to evaluate malignancy. The Wisconsin breast cancer diagnosis database is the result of the efforts made by Dr. W.H. Wolberg, at the University of Wisconsin Hospital for accurately diagnosing breast masses based solely on an FNA test and is available at the University of California- Irvine, machine learning repository [30].

Nine visually assessed characteristics of an FNA sample considered relevant for diagnosis were identified, and assigned an integer value between 1 and 10 as shown in Table 6.1. Each case has nine input attributes, as shown in Table 6.2; and a binary diagnostic output, whether the case is benign or malignant. But the diagnostics do not provide any information about the degree of benignity or malignancy. It admits a relatively high number of variables and consequently a large search space. WBCD database consists of 683 cases of breast biopsy evaluations. The output diagnostics in the WBCD database were furnished by specialists in the field. The objective is to evolve an optimized fuzzy inference system for this dataset, such that it can diagnose benignancy or malignancy for any breast cancer patient.

Table 6.1 WBCD dataset

Case	υ_1	υ_2	υ_3	...	υ_9	*diagnostic*
1	***5***	***1***	***1***	**…**	***4***	***Benign***
2	***5***	***4***	***4***	**…**	***1***	***Benign***
:	***:***	***:***	***:***	**…**	***:***	***:***
683	***4***	***8***	***8***	**…**	***1***	***Malignant***

Table: 6.2 Input attribute information

Input attribute name	***Input attribute symbol***

Clump thickness	υ_1
Uniformity of cell size	υ_2
Uniformity of cell shape	υ_3
Marginal adhesion	υ_4
Single epithelial cell size	υ_5
Bare nuclei	υ_6
Bland chromatin	υ_7
Normal nucleoli	υ_8
Mitosis	υ_9

Output diagnostic classes

1. Benign
2. Malignant

6.1.1 Evolving fuzzy systems for the WBCD problem

The solution scheme for the WBCD problem depicted in Fig 1.1 is referred now. In order to evolve the fuzzy model we must make some preliminary decisions about the fuzzy system itself and about the genetic algorithm encoding.

Previous knowledge about the WBCD problem and about some of the extant rule-based models represents valuable information are used for the choice of fuzzy parameters. When defining the set up, following results, described in previous works are taken into consideration.

- ❖ *Small number of rules*. Systems with no more than four rules have been shown to obtain high performance [13, 20].
- ❖ *Small number of variables*. Rules with no more than four antecedents have proven adequate [13, 21, 23].

- *Monotonicity of the input variables*. Simple observation of the input and output spaces shows that higher-valued variables are associated with malignancy. Some fuzzy models forgo interpretability in the interest of improved performance. Where medical diagnosis is concerned, interpretability—also called linguistic integrity—is the major advantage of fuzzy systems[13].

- *Justifiable number of elements*. The number of membership functions of a variable should be compatible with the number of conceptual entities a human being can handle. The same criterion is applied to the number of variables in the rule antecedent. For example, the following would be considered an adequate rule:

  if (υ_1 **is** *High*) **and** (υ_2 **is** *High*) **and** (υ_4 **is** *High*) **and** (υ_6 **is** *Low*) **and** (υ_8 **is** *Low*) **then** (*output* **is** *benign*).

- *Coverage*. Any element from the universe of discourse should belong to at least one of the fuzzy sets. That is, its membership value must be different than zero for at least one of the linguistic labels. Referring to Fig. 6.1, we see that any value along the *x*-axis belongs to at least one fuzzy set (*Low*, *High*, or both); no value lies outside the range of all sets.

- *Normalization*. Since all labels have semantic meaning, then, for each label, at least one element of the universe of discourse should have a membership value equal to one. In Fig. 6.1, we observe that both *Low* and *High* have elements with membership value equal to 1.

- *Orthogonality*. For each element of the universe of discourse, the sum of all its membership values should be equal to one (e.g. in Fig.6.1 a *Low* membership value of 0.8 entails a *High* membership value of 0.2).

Taking into account the above criteria, we delineate the fuzzy system setup.

6.1.1.1 Fuzzy system parameters set up.

- ***Fuzzy system type* :** Mamdani
- ***Logical parameters***
  - Reasoning mechanism: singleton-type fuzzy system, meaning that consequents of rules (i.e. output membership functions) are real values (also called singletons), rather than fuzzy ones.
  - Fuzzy operators: min

- Input membership function type: Trapezoidal. (See Fig. 6.1). *Though triangular was also tried, but it gave less fitness value as compared to trapezoidal for WBCD.*
- Defuzzification method: Weighted average.

❖ ***Structural parameters***

- Number of input membership functions: Two membership functions, denoted *Low* and *High* are used (Fig.6.1). Here P and d are the parameters used to describe the membership functions.

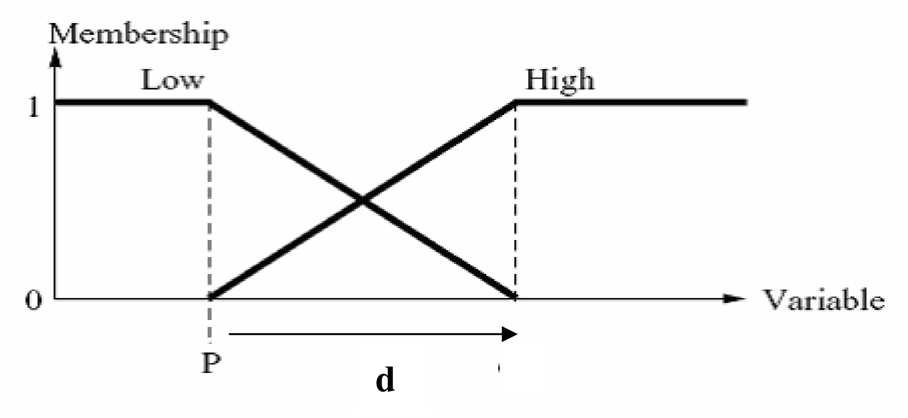


Figure 6.1 Trapezoidal input membership function

- Number of output membership functions: Two singletons are used, corresponding to the *benign* and *malignant* diagnostics.
- Number of rules: This is a user-configurable parameter. It can be limited the number of rules to be between 1 and 5. Number of rules has been chosen to be *three or five.* The rules themselves are to be found by the genetic algorithm.

❖ ***Connective parameters***

- Antecedents of rules: to be found by the genetic algorithm.
- Consequent of rules: the algorithm finds rules for the *benign* diagnostic; the *malignant* diagnostic is an else condition.
- Rule weights: active rules have a weight of value 1 and the *else* condition has a weight of 0.25.

❖ ***Operational parameters***

- Input membership function values: to be found by the genetic algorithm.
- Output membership function values: following the WBCD database, a value of 2 for *benign* and 4 for *malignant* is used.

6.1.1.2 Genetic algorithm parameters

The Genetic algorithm parameters taken for the experiments are as shown in Table 6.3

Table 6.3: GA parameters

Population size	20, 30
Maximum number of Generations	20, 50
Mutation probability	0.01
Crossover probability	1
Termination criteria	Max number of generations reached or when the increase in fitness of the best individual over five successive generations falls below a threshold of 0.001
Selection Procedure	Fitness proportionate

6.1.2 Encoding the genome

Pittsburgh-style structure learning is applied, using a genetic algorithm to search for two parameters. The genome, encoding input membership function values and antecedents of rules is constructed as follows [14]:

6.1.2.1 Membership function parameters.

There are nine input variables ($\upsilon_1 - \upsilon_9$), each with two parameters *P* and *d*, defining the start point and the end point of the membership function edges, respectively (Fig.6.1).

6.1.2.2 Antecedents

The i-th rule has the form:

if (υ_1 **is** $A_1{}^i$) **and**…**and** (υ_9 **is** $A_9{}^i$) **then** (output **is** benign),

where $A_j{}^i$ represents the membership function applicable to variable υ_j. $A_j{}^i$ can

take on the values: 1 (*Low*), 2 (*High*), or 0 or 3 (*Other*).

The total number of bits required for encoding the genome is as shown in table 6.3.

Table 6.4: Parameter encoding of an individual's genome

Parameters	Values	Bits	Quantity	Total Bits
P	{1,2…8}	3	9	27
d	{1,2…8}	3	9	27
A	{0,1,2,3}	2	$9\times N_r$	$18 \times N_r$

The total genome length is 54 + $18N_r$, where N_r denotes the number of rules (set to 3 or 5), which is fixed during the genetic run.

An example of a simple one rule genome encoding is shown in the figure 6.2.

P_1	d_1	P_2	d_2	P_3	d_3	P_4	d_4	P_5	d_5	P_6	d_6	P_7	d_7	P_8	d_8	P_9	d_9
4	3	1	5	2	7	1	1	1	6	3	7	4	6	7	1	1	5

A^1_1	A^1_2	A^1_3	A^1_4	A^1_5	A^1_6	A^1_7	A^1_8	A^1_9
0	1	3	3	2	3	1	3	1

(a) Database

	υ_1	υ_2	υ_3	υ_4	υ_5	υ_6	υ_7	υ_8	υ_9
P	**4**	***1***	***2***	***1***	***1***	***3***	***4***	***7***	***1***
d	***3***	***5***	***7***	***1***	***6***	***7***	***6***	***1***	***5***

Rule 1:	if (υ_2 is low) and (υ_5 is High) and (υ_7 is low) and (υ_9 is low) then (output is benign).
Default:	else output is malignant.

(b) Rule base

Figure 6.2: Example of a genome for a single rule system

Figure 6.2 shows:

(a) *Genome encoding*. The first 18 positions encode the parameters P and d for the 9 input variables. The rest encode the membership functions applicable to the nine antecedents for each rule: 1 (*Low*), 2 (*High*), or 0 or 3 (*Other*).

(b) *Interpretation.* Data base and rule base of the single rule system encoded by (a)

6.1.3 Fitness function

As the main function of the proposed system is the assessment of a medical diagnosis, the fitness function takes into account the criteria of classification performance, computed as the ratio of cases correctly diagnosed (N_C) to the total cases.

$$\text{Fitness} = N_C \,/\, \text{total number of patient cases.}$$

6.2 Liver Disorder Diagnosis

Liver is the largest internal human organ and one of the most important one too. Liver disorders can be congenital, injury-related, viral-induced, or alcohol-induced. BUPA Medical Research Ltd has recorded these data evaluations of patients complaining for liver disorders. The data of this dataset is available at the University of California-Irvine machine learning repository [31] .The dataset consists of 345 patient records with six input variables (Table 6.4). There are two output diagnostic classes. Each record contains data for one male patient. The first five variables are all blood tests which are thought to be sensitive to liver disorders that might arise from excessive alcohol consumption. This data set was donated by R. S. Forsyth. The problem is to predict whether or not a male patient has a liver disorder based on blood tests and alcohol consumption.

Table 6.5 Liver disorder input attribute information

Input attribute name	***Input attribute symbol***	***Range [Min Max]***
Mean Corpuscular Volume	MCV(v_1)	[65 103]
Alkaline Phosphotase	ALKPHOS(v_2)	[23 138]
Alamine Aminotransferase	SGPT(v_3)	[4 155]

Aaspartate Aminotransferase	SGOT(v_4)	[5 82]
Gamma-Glutamyl Transpeptidase	GAMMA GT(v_5)	[5 297]
Number of half-pint equivalents of alcoholic beverages drunk per day	Drinks(v_6)	[0 20]

Output diagnostic classes

1. Liver disorder
2. No liver disorder

6.2.1 Evolving fuzzy systems for the Liver disorder problem

6.2.1.1 Fuzzy system parameters set up

- **Fuzzy system type :** Mamdani
- **Logical parameters**
  - Reasoning mechanism: singleton-type fuzzy system.
  - Input membership function type: Triangular. (Fig. 6.3). *Though trapezoidal was also tried, but it gave less fitness value as compared to triangular for liver dataset.*
  - Defuzzification method: Weighted average.
- ***Structural parameters***
  - Number of input membership functions:
    - Three Membership functions, denoted as *Low, Medium and High* (Figure 6.3). Here the membership parameters P_1, P_2, P_3, P_4 and P_5 are encoded into the genome. So a total of five parameters (P = 5) denote the three membership functions in this case, instead of two as in the previous case. L_L and U_L are the lower and upper ranges of the input.

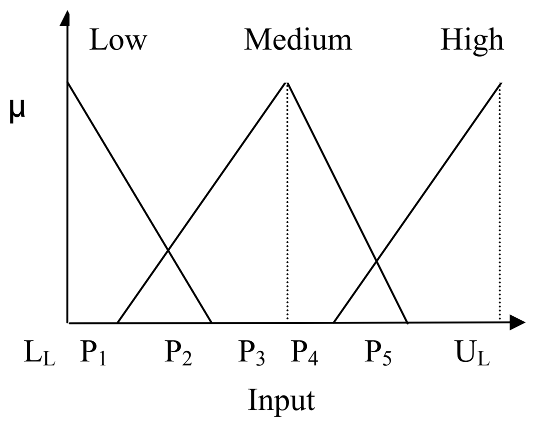


Figure 6.3Three triangular input membership function

- Five membership functions, denoted by *Vlow, Low, Medium, High and Vhigh* are used (Fig. 6.4). Here the membership parameters P_1, P_2, P_3 and the three centers (C_1, C_2, and C_3) are encoded into the genome. So a total of six parameters (P = 6) denote the five membership functions in this case.

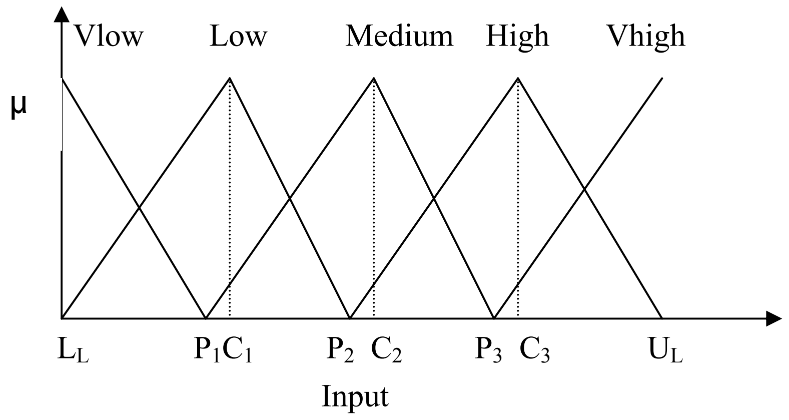


Figure 6.4 Five triangular input membership functions

- Number of output membership functions: two singletons are used, corresponding to the *Liver disorder* and *No liver disorder* diagnostics.
- Number of rules: It is chosen to be *three* and *five*.

❖ ***Connective parameters***

- Antecedents of rules: to be found by the genetic algorithm.
- Consequent of rules: the algorithm finds rules for the *Liver disorder* diagnostic; the *No disorder* diagnostic is an else condition.

- Rule weights: active rules have a weight of value 1 and the *else* condition has a weight of 0.25.

❖ ***Operational parameters***

- Input membership function values: to be found by the genetic algorithm.
- Output membership function values: following the database, a value of 1 for *Liver disorder* and 2 for *No liver disorder* is used.

6.2.1.2 GA parameters

The same parameter values were used as shown in Table 6.3

6.2.2 Encoding the genome

6.2.2.1 Membership function parameters.

The number of input variables (N_I) is six, each with five or six parameters P, defining the membership functions. (Fig. 6.3 and 6.4). The bits required to encode each input parameter is shown in table 6.5. Hence, total number of bits to encode the membership function parameters would be (5+7+7+6+8+4) $\times$ P = 37$\times$P bits .

6.2.2.2 Antecedents.

Antecendents can take on values:

❖ For 3 Membership functions: 1 (Low), 2 (Medium), 3(High). It would require 2 bits (B = 2) to encode each of these values.

❖ For 5 Membership functions: 1 (VLow), 2 (Low), 3 (Medium), 4(High), 5 (Vhigh). It would require 3 bits (B = 3) to encode each of these values

Table 6.6 Parameter encoding of an individual's genome for liver dataset

Input	Range of Values [Min Max]	Bits required for each input parameter	Total Bits for each input variable encoding
MCV	[65 103]	5	5×P
ALKPHOS	[23 138]	7	7×P
SGPT	[4 155]	7	7×P
SGOT	[5 82]	6	6×P
GAMMA GT	[5 297]	8	8×P

Drinks	[0 20]	4	$4 \times P$

Hence, for 5 triangular Membership functions, 3 rule system , total genome length is :

$(37 \times P) + (N_r \times B \times N_I) = (37 \times 6) + (3 \times 3 \times 6) =$ 276 bits

6.2.3 Fitness function

Fitness = N_C / total number of patient cases.

6.3 Diabetes diagnosis

Diabetes is a disease in which blood glucose levels are above normal. People with diabetes have problems converting food to energy. Cells use insulin, a hormone made in the pancreas, to help them convert blood glucose into energy. People develop diabetes because the pancreas does not make enough insulin or because the cells in the muscles, liver, and fat do not use insulin properly, or both. As a result, the amount of glucose in the blood increases while the cells are starved of energy. Diabetes is caused by genetic and environmental factors.

Women with gestational diabetes are an ideal group to target for primary prevention. Gestational diabetes develops in some women during the late stages of pregnancy caused by the hormones of pregnancy or by a shortage of insulin. Pregnant women who develop glucose intolerance during pregnancy are at lifetime risk of developing diabetes after a varying period of life. Their children face the same fate[18]. The data of the epidemic of diabetes in Pima Indians has been used extensively for diabetes research. A population of women who were at least 21 years old, of Pima Indian heritage and living near Phoenix, Arizona, was tested for diabetes according to World Health Organization criteria. The data were collected by the US National Institute of Diabetes and Digestive and Kidney Diseases and made into the pima-Indian diabetes dataset.[32]

The diagnostic, binary-valued variable investigated is whether the patient shows signs of diabetes according to World Health Organization criteria (i.e., if the 2 hour post-load Plasma glucose was at least 200 mg/dl at any survey examination or if found during routine medical care). The dataset consists of eight inputs and one diagnostic class variable. It has 768 female patient data.

The objective is to evolve an optimized fuzzy inference system for this dataset, such that it can diagnose whether a female patient suffers from diabetes or not and thus whether she is at a risk to pass it to her offspring.

Table 6.7 Diabetes input attribute information

Input attribute name	***Range [Min Max]***	***Symbol***
Number of times pregnant	[0 17]	υ_1
Plasma glucose concentration a 2 hours in an oral glucose tolerance test	[0 199]	υ_2
Diastolic blood pressure	[0 122]	υ_3
Triceps skin fold thickness	[0 99]	υ_4
2-Hour serum insulin	[0 846]	υ_5
Body mass index	[0.0 67.1]	υ_6
Diabetes pedigree function	[0.078 2.42]	υ_7
Age	[21 81]	υ_8

Output classes

1 . Tested positive for diabetes

2. Tested negative for diabetes

6.3.1 Evolving fuzzy systems for the pima Indian diabetes problem

6.3.1.1 Fuzzy system parameters set up

The logical, connective, structural and Operational parameters are kept same as in the case of liver disorder problem.

6.3.1.2 GA parameters

The genetic algorithm parameters are same as shown in table 6.3

6.3.2 Encoding the genome

6.3.2.1 Membership function parameters.

There are six input variables, each with five or six parameters (P = 5 or P = 6), defining the membership functions (Fig. 6.3 and Fig 6.4). The bits required to encode each input parameter is shown in table 6.7. Hence, total number of bits to encode the membership function parameters would be (4+8+7+7+10+6+2+6) $\times$ P = 50 $\times$P bits .

6.3.2.2 Antecedents.

- For 3 Membership functions: Antecedents can take on values: 1 (Low), 2 (Medium), 3(High). It would require 2 bits (B = 2) to encode each of these values.
- For 5 Membership functions: Antecedents can take on values: 1 (VLow), 2 (Low), 3 (Medium), 4(High), 5 (Vhigh). It would require 3 bits (B = 3) to encode each of these values

Table 6.8 Parameter encoding of an individual's genome for pima-indian diabetes dataset

Input	Range of Values [Min Max]	Bits required for each input parameter	Total Bits for each input variable encoding
υ_1	[0 17]	4	4$\times$P
υ_2	[0 199]	8	8$\times$P
υ_3	[0 122]	7	7$\times$P
υ_4	[0 99]	7	7$\times$P
υ_5	[0 846]	10	10$\times$P
υ_6	[0.0 67.1]	6	6$\times$P
υ_7	[0.078 2.42]	2	2$\times$P
υ_8	[21 81]	6	6$\times$P

- For 5 triangular Membership functions, 3 rule system , total genome length is : $(50 \times P) + (N_r \times B \times N_I) = (50 \times 6) + (3 \times 3 \times 8) =$ 372 bits,

  where N_I is the number of inputs, B is the bits required to code antecedent and N_r is the number of rules.

- For 3 triangular Membership functions, 3 rule system , total genome length is : $(50 \times P) + (N_r \times B \times N_I) = (50 \times 5) + (3 \times 2 \times 8) =$ 298 bits

6.3.3 Fitness function

As the main function of the proposed system is the assessment of a medical diagnosis, the fitness function takes into account the criteria of classification performance, computed as the ratio of cases correctly diagnosed (N_C) to the total cases.

Fitness = N_C / total number of patient cases.

6.4 A general Flowchart

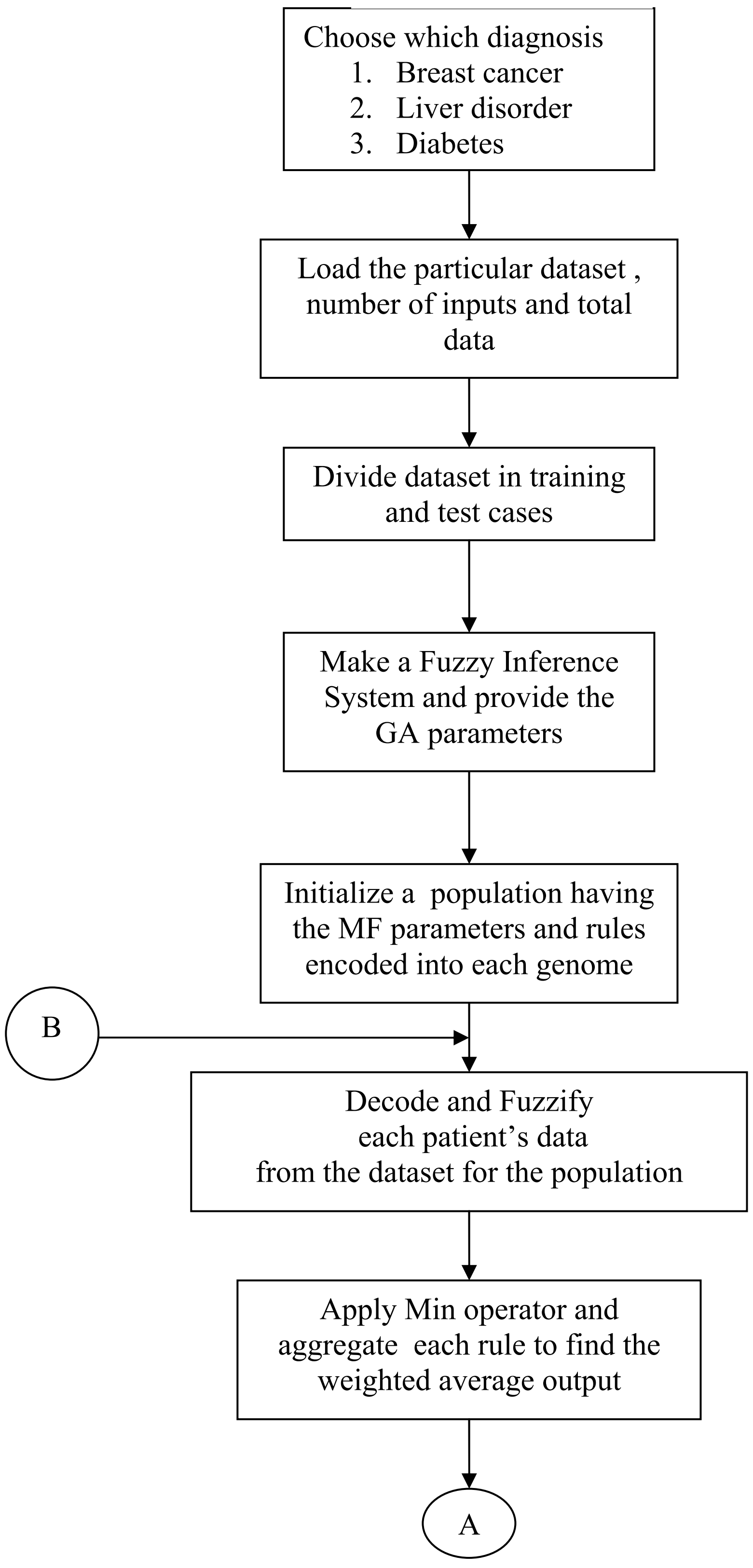
Referring to the proposed diagnostic system shown in figure 1.1, a fuzzy inference system needs to be modeled first. A random initial population is taken with the membership function parameters and rule encoded into it. Each patient's data from the dataset is applied to the fuzzy subsystem. Fitness value for each individual (chromosome-entire fuzzy system here) is found out. Thus the entire fuzzy inference system is evolved from scratch. The final evolved fuzzy system is then tested with 50% of the dataset (validation).

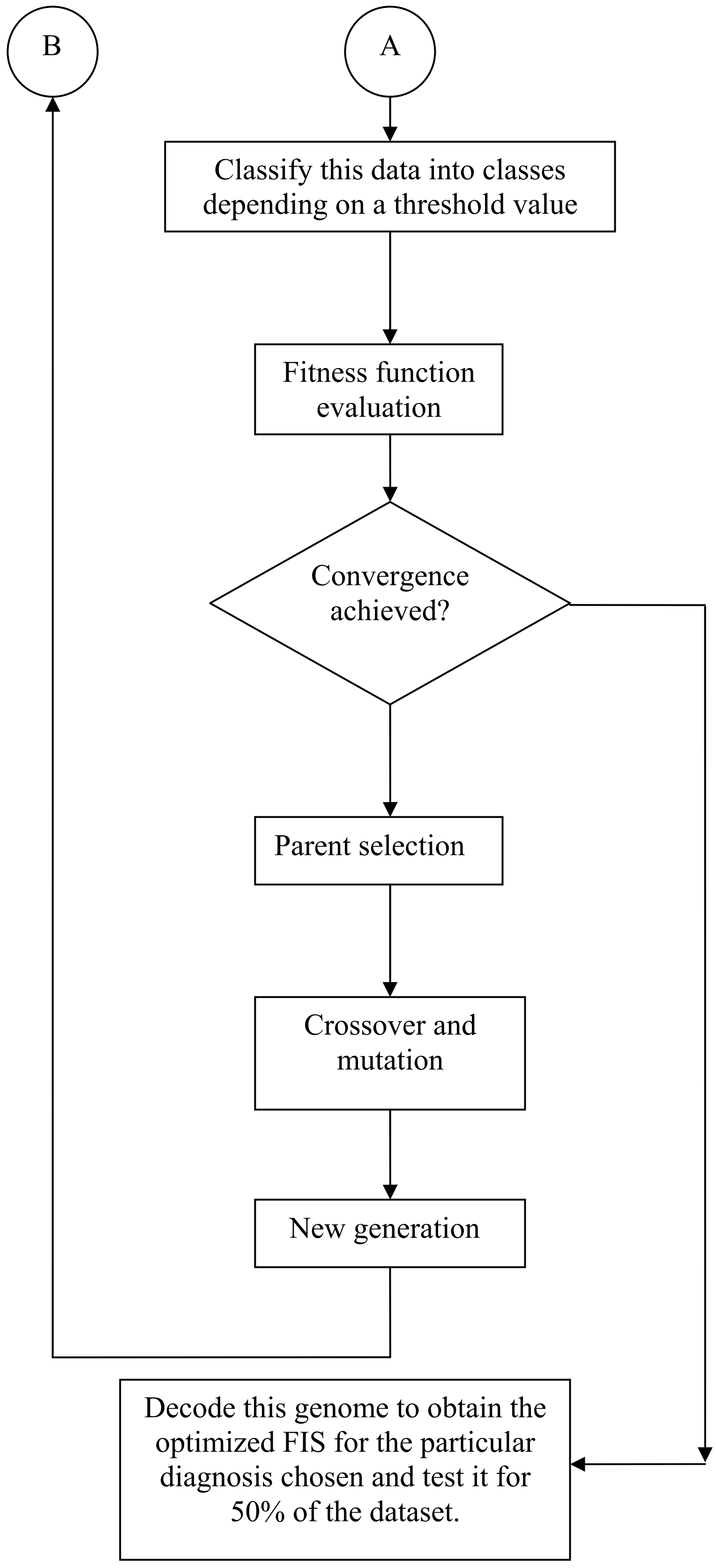
The code has been implemented in MATLAB 7.0. Fuzzy logic toolbox and the genetic optimization functions are the platform on which the code functions. The diagnostic code is takes care of :

- ❖ The number and the type of membership functions suited for each diagnosis.
- ❖ Implementing the weighted average function.
- ❖ The length of chromosome as different inputs has different ranges and different number of rules.

A general flowchart is shown as follows.

A general flowchart


Choose which diagnosis
1. Breast cancer
2. Liver disorder
3. Diabetes
Load the particular dataset , number of inputs and total data
Divide dataset in training and test cases
Make a Fuzzy Inference System and provide the GA parameters
Initialize a population having the MF parameters and rules encoded into each genome
B
Decode and Fuzzify each patient's data from the dataset for the population
Apply Min operator and aggregate each rule to find the weighted average output
A


B
A
Classify this data into classes depending on a threshold value
Fitness function evaluation
Convergence achieved?
Parent selection
Crossover and mutation
New generation
Decode this genome to obtain the optimized FIS for the particular diagnosis chosen and test it for 50% of the dataset.

6.5 CONCLUSION

The fuzzy genetic approach to solve each of the diagnosis problems has been dealt with in detail. The fuzzy subsystem modeling of the proposed solution is explained. The methodology to encode the genome, the fitness function which is to be maximized has been presented.

CHAPTER VII

RESULTS AND DISCUSSION

The proposed diagnostic system consisting of the fuzzy system and the threshold unit diagnoses all three medical problems, providing excellent performance, i.e. diagnose and classify the presented cases correctly. The entire fuzzy system is optimized using genetic algorithm. The designed system has been implemented in the MATLAB 7.0 programming environment.

The evolutionary experiments performed for each problem with proper validation has the training set containing 50% of the dataset cases and the test set containing the remaining 50% of the cases. In each case, the membership function which works best has been analyzed. The results are tabulated for trapezoidal and triangular membership function cases. Also the results obtained by encoding three and five triangular input membership functions into the genome, have been shown.

The best evolved fuzzy system is the one which has the best diagnosis–classification performance. The percentage of correctly classified data is obtained on providing the testing dataset to the optimized fuzzy system. Average number of variables per rule has been considered, as more of this number, more is the complexity and more difficult is the interpretability. Computation time is also noted and it's seen that as the genome length increases the time taken is more. The number of rules needs to be initialized beforehand. Three and five rule base system results are shown. For each medical problem, the evolved fuzzy inference system i.e. the optimized input membership function plots and the rule base, has been plotted.

The diagnostic system is also tested to find out the diagnostic confidence measure, thus going beyond a mere binary classification instead giving a better understanding about the system output. This is done by giving a patient's data information and verifying the output from the dataset.

7.1 Breast Cancer diagnosis

7.1.1 The best membership function

The results for trapezoidal and triangular membership function have been tabulated as in table 7.1 and table 7.2 respectively.

Table: 7.1 Results for WBCD: trapezoidal membership function

Number of Rules	Fitness	% of correctly classified data	Average number of variables per rule	Computation time (sec)
3	**0.947368**	**97.9412**	**4.33**	**160**
3	0.906433	97	4	110
3	0.932749	97.3529	4.66	99
2	0.862573	90.8824	3.55	96
2	0.859649	91.7647	4	100
2	0.763158	92.3529	4	103

Table: 7.2 Results for WBCD: triangular membership function

Number of Rules	Fitness	% of correctly classified data	Average number of variables per rule	Computation time (sec)
3	0.847953	92.3529	7	426
3	0.877193	93.2353	5.66	430
3	0.880117	96.4706	6.66	427

Result: As seen from the table 7.1 and table 7.2 for WBCD problem the trapezoidal membership function works better.

Hence the best evolved fuzzy inference system with classification performance of 97.9412% is as below. The Fitness Vs generation plot and the nine input membership function plots are shown in figure 7.1 and figure 7.2 respectively.

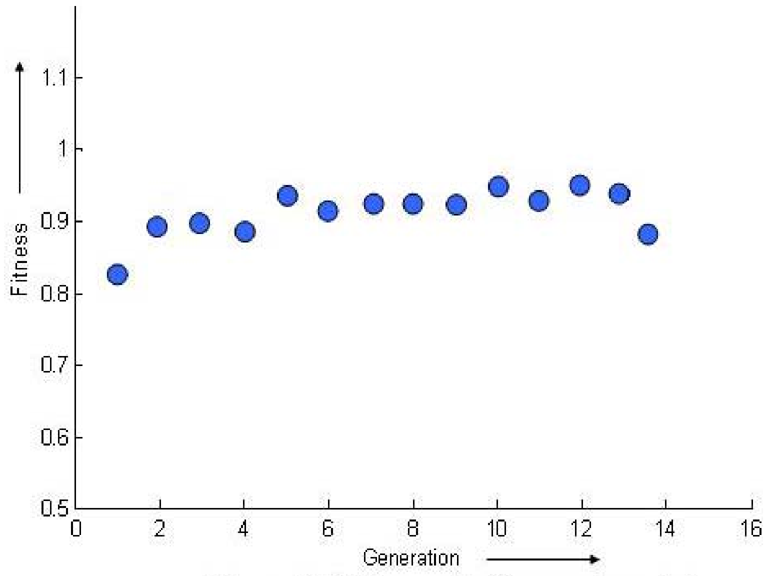
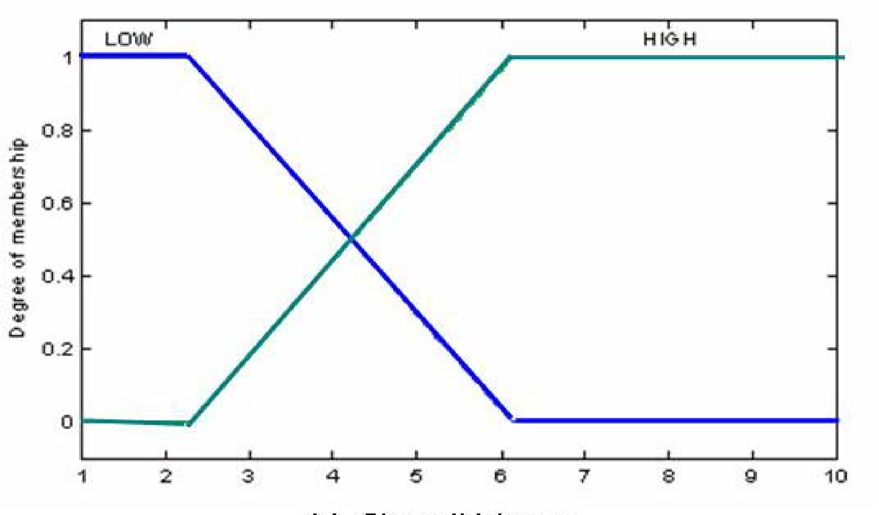
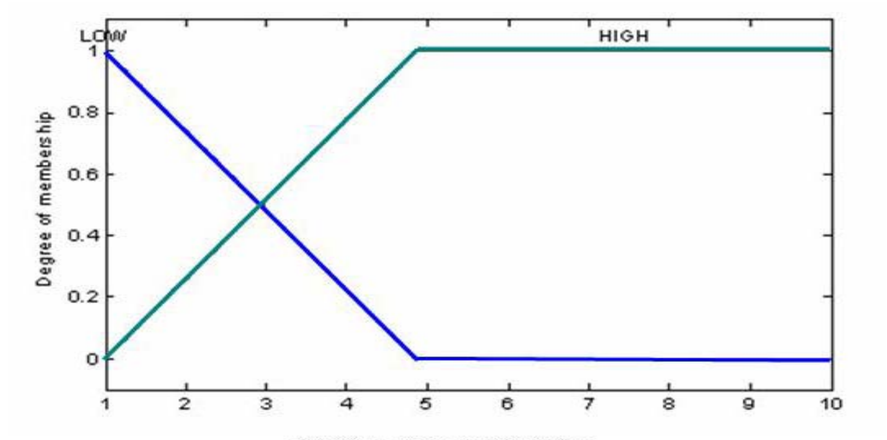


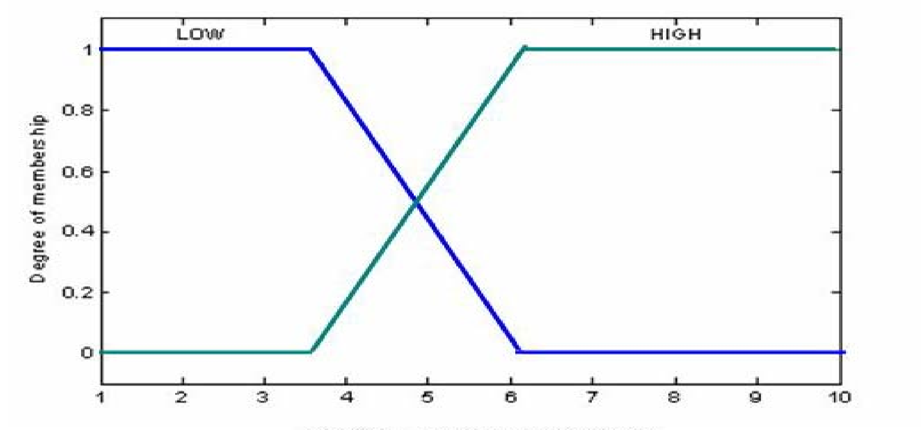
Figure 7.1 Fitness Vs Generation plot



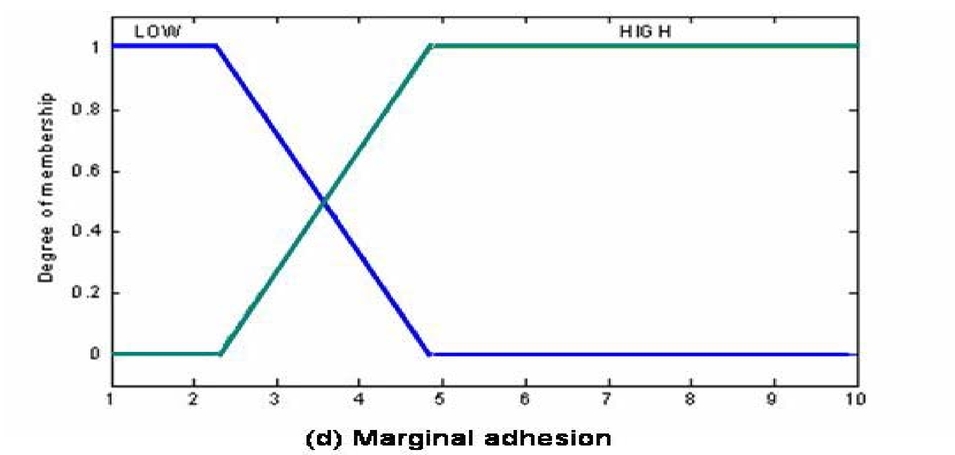
(a) Clump thickness



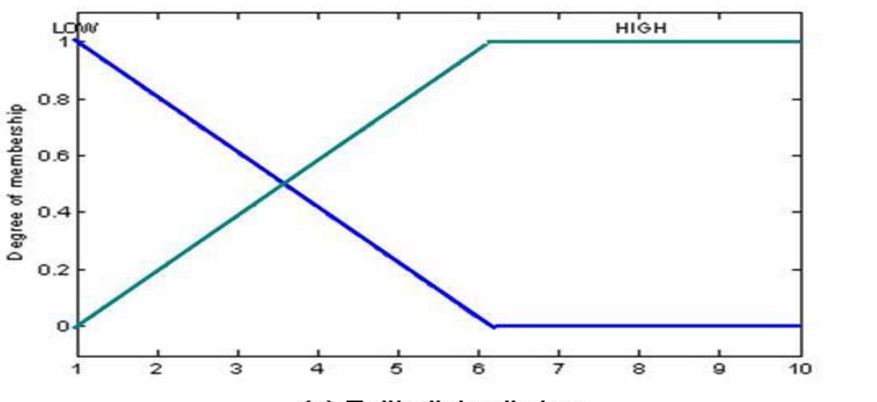
(b) Cell size Uniformity



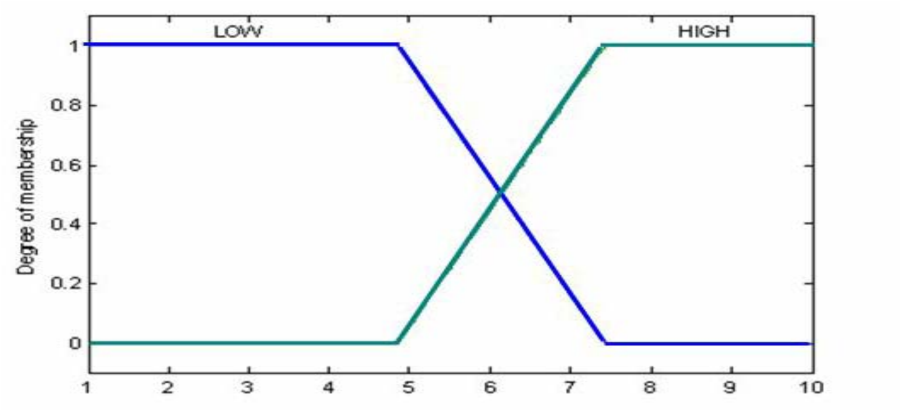
(c) Cell shape Uniformity



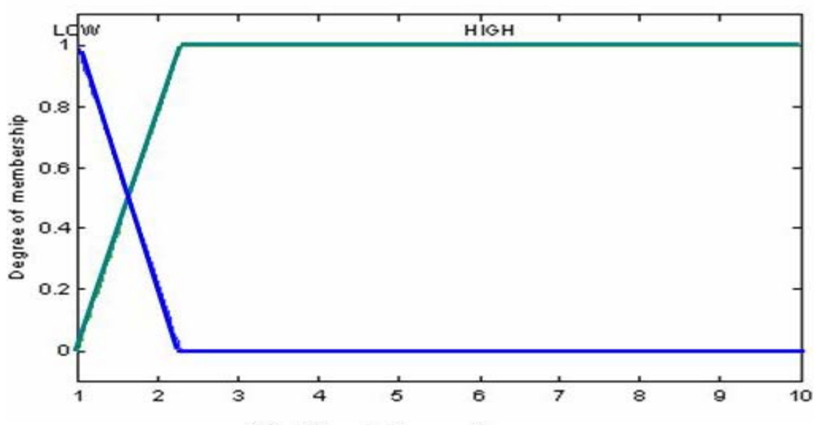
(d) Marginal adhesion



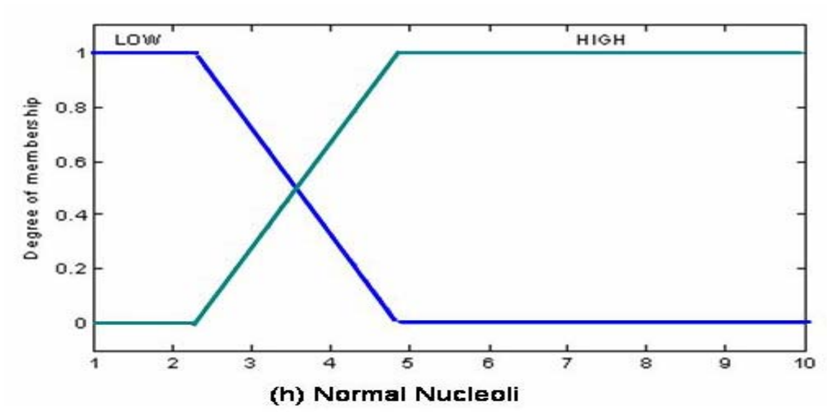
(e) Epithelial cell size



(f) Bare Nuclei



(g) Bland chromatin



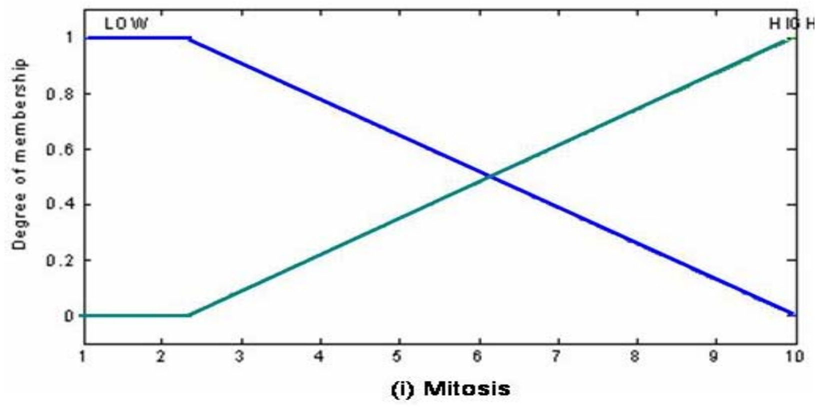


Figure 7.2 Optimized input membership function plots for WBCD-3 rule base

Optimized 3 rule base

1. If *(Cell size Uniformity is LOW)* and *(Epithelial cell size is LOW)* and *(Bare nuclei is LOW)* then *(Diagnosis is BENIGN)*

2. If *(Clump thickness is HIGH)* and *(Cell size Uniformity is HIGH)* and *(Cell shape Uniformity is LOW)* and *(Marginal adhesion is HIGH)* and *(Bare nuclei is HIGH)* and *(Bland chromatin is LOW)* and (*Mitosis is HIGH)* then *(Diagnosis is BENIGN)*

3. If *(Marginal adhesion is LOW)* and *(Bland chromatin is LOW)* and *(Mitosis is LOW)* then *(Diagnosis is BENIGN)*

4. else *(Diagnosis is MALIGNANT)*

7.1.2 Evolved fuzzy system for two trapezoidal membership functions and five rule base.

Table 7.3 Results for WBCD: trapezoidal MF and five rule base

Number of Rules	Fitness	% of correctly classified data	Average number of variables per rule	Computation time (sec)
5	**0.938596**	**98.5294**	**5.2**	**244**
5	0.947368	97.0588	5.6	236
5	0.935673	94.7059	4.8	232

Hence the best evolved five rules fuzzy inference system with classification performance of 98.5294% is as below. The Fitness Vs generation plot and the nine input membership function plots are shown in figure 7.3 and figure 7.4 respectively.

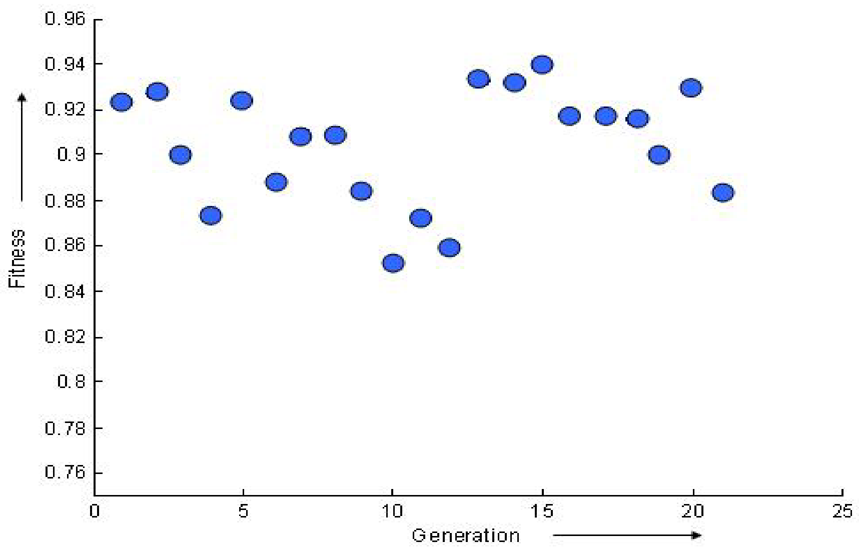
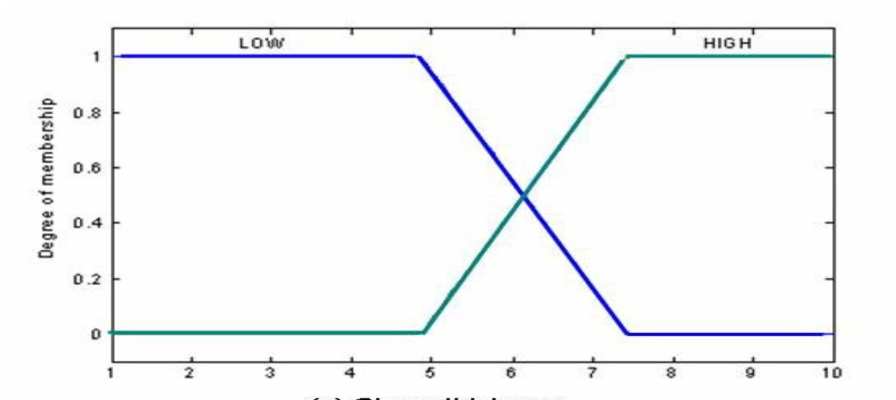
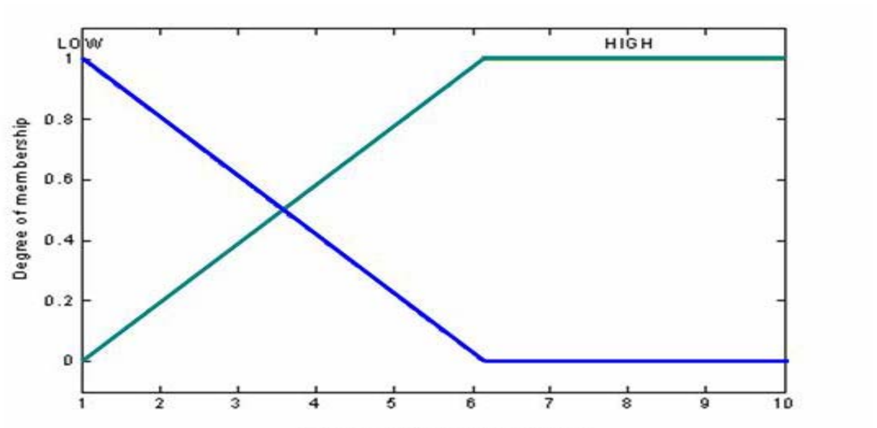


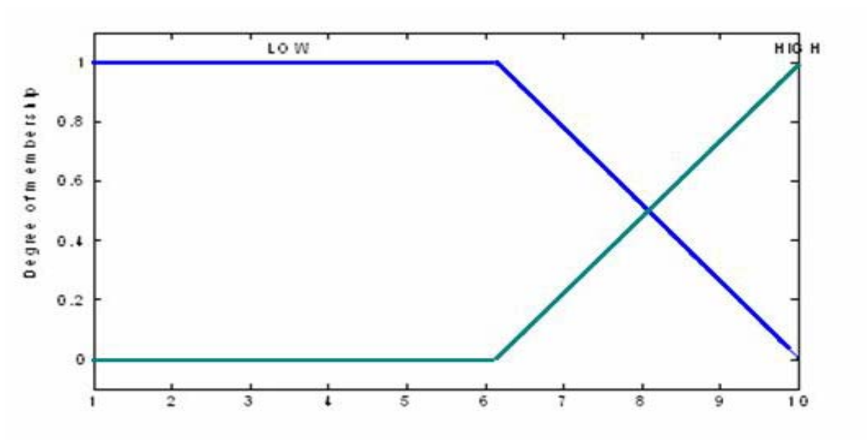
Figure 7.3 Fitness Vs Generation plot



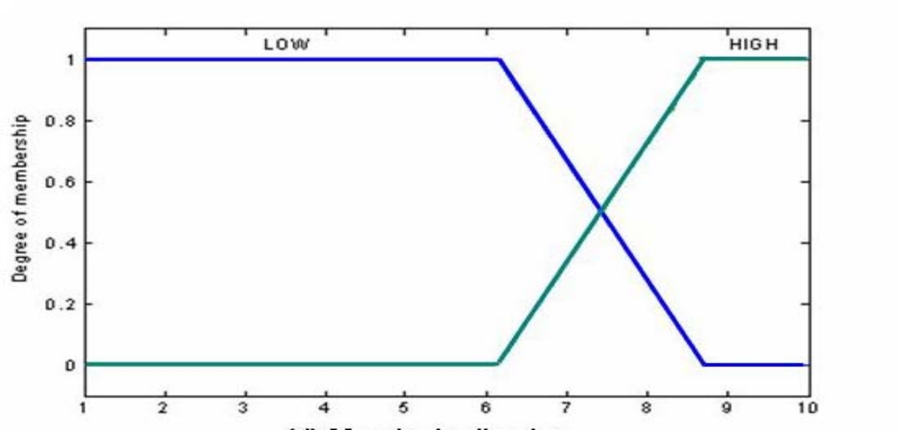
(a) Clump thickness



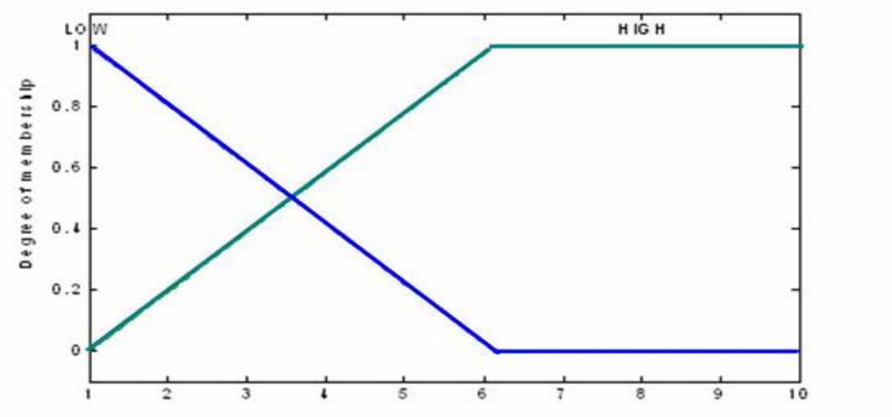
(b) Cell size Uniformity



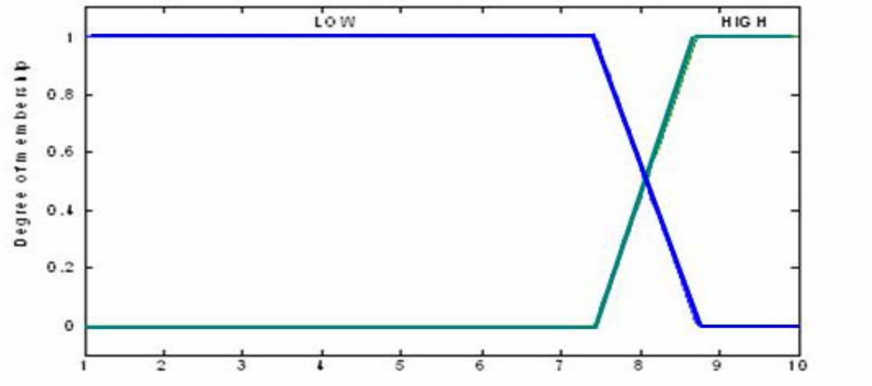
(c) Cell shape Uniformity



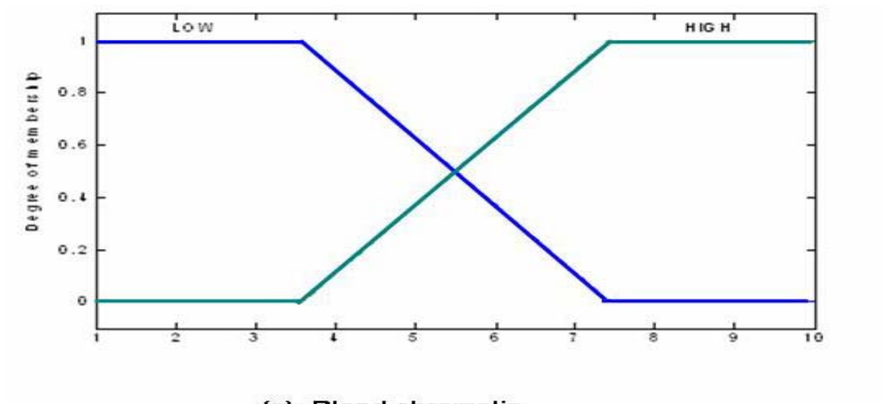
(d) Marginal adhesion



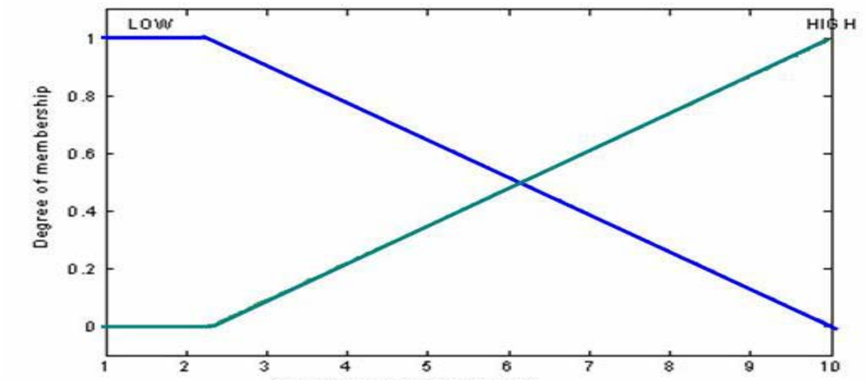
(e) Epithelial cell size



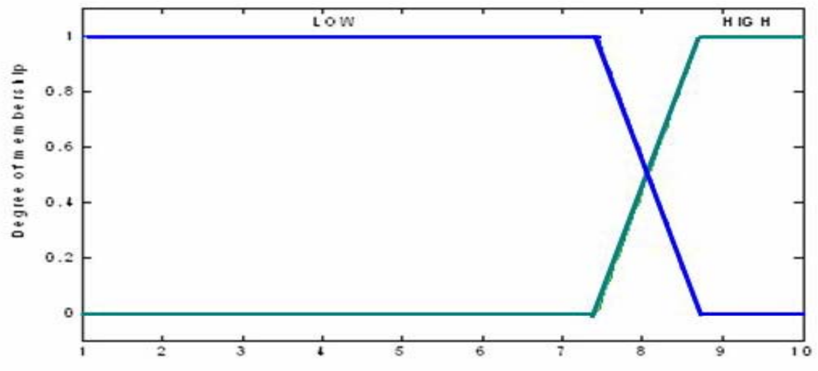
(f) Bare Nuclei



(g) Bland chromatin



(h) Normal Nucleoli



(i) Mitosis

Figure 7.4 Optimized input membership function plots for WBCD-5 rule base

The optimized five rule base

1. If *(Cell size Uniformity is HIGH)* and *(Cell shape Uniformity is LOW)* and *(Epithelial cell size is LOW)* and *(Bland chromatin is LOW)* and *(Normal nucleoli is LOW)* and *(Mitosis is LOW)* then *(Diagnosis is BENIGN)*

2. If *(Clump thickness is LOW)* and *(Cell size Uniformity is LOW)* and *(Cell shape Uniformity is HIGH)* and *(Marginal adhesion is HIGH)* and *(Epithelial cell size is LOW)* and *(Bare nuclei is LOW)* and *(Normal nucleoli is LOW)* and *(Mitosis is LOW)* then *(Diagnosis is BENIGN)*

3. If *(Cell shape Uniformity is LOW)* and *(Normal nucleoli is LOW)* then *(Diagnosis is BENIGN)*

4. If *(Cell shape Uniformity is HIGH)* and *(Epithelial cell size is LOW)* and *(Mitosis is HIGH)* then *(Diagnosis is BENIGN)*

5. If *(Clump thickness is LOW)* and *(Cell size Uniformity is LOW)* and *(Cell shape Uniformity is LOW)* and *(Marginal adhesion is LOW)* and *(Epithelial cell size is LOW)* and *(Bland chromatin is LOW)* and *(Normal nucleoli is LOW)* then *(Diagnosis is BENIGN)*

6. else (*Diagnosis is MALIGNANT)*

7.1.3 Testing the evolved fuzzy system-diagnostic confidence

Suppose the following input data of patient no #145 from the WBCD dataset is given to the evolved fuzzy system of topic 7.1.2.

υ_1	υ_2	υ_3	υ_4	υ_5	υ_6	υ_7	υ_8	υ_9
4	3	1	1	2	1	4	8	1

The membership value of each variable is then computed in accordance with the evolved database of 7.1.2.

Table: 7.4 Fuzzification using the evolved database for WBCD

	υ_1	υ_2	υ_3	υ_4	υ_5	υ_6	υ_7	υ_8	υ_9
μ_{Low}	1	0.6	1	1	0.8	1	0.9	0.25	1
μ_{High}	0	0.4	0	0	0.2	0	0.1	0.75	0

Rule 1: $Min(0.4\ ,1\ ,0.8\ ,0.9\ ,0.25\ ,1\ ) = 0.25$

Rule 2: $Min(1\ ,0.4\ ,0\ ,0\ ,0.8, 1,\ 0.25,\ 1\ ) = 0$

Rule 3: $Min(0.4\ ,\ 0.25\ ) = 0.25$

Rule 4: $Min(0\ ,0.8\ ,0) = 0$

Rule 5: $Min(1\ ,0.6\ ,1\ ,1\ ,0.8\ ,0.9\ ,0.25\ ) = 0.25$

Each of the above rules has rule weight of 1 and the malignant rule has rule weight of 0.25. The output functions are singletons at 2 and 4 for benign and malignant respectively.

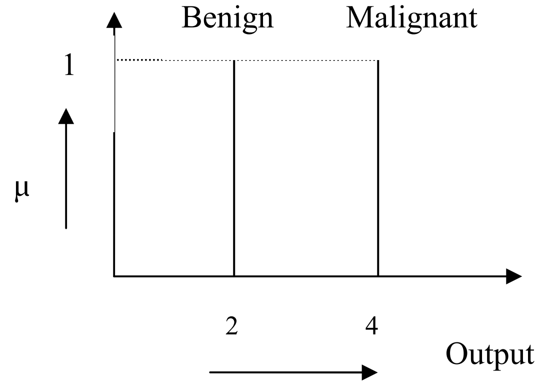


Figure 7.5 Output singletons membership functions

Weighted average for defuzzification:

Defuzzified output = $\dfrac{(0.25\times 2)+(0.25\times 2)+(0.25\times 2)+(0.25\times 4)}{0.25+0.25+0.25+0.25} = 2.5$

Discussion :

- This appraisal (defuzzified) output from the evolved fuzzy system is then passed on to the threshold system having a threshold of 3. As 2.5 is less than 3, we have the diagnosis that it benign. *This is correct in relation to the output given in the output dataset of the WBCD.*
- The best diagnostic classification performance obtained is that of **98.5294 %**

7.2 The Liver disorder problem.

7.2.1 The best membership function.

The results for trapezoidal and triangular membership function have been tabulated as in table 7.5 and table 7.6 respectively.

Table 7.5 Results for Liver disorder: trapezoidal membership function

Number of Rules	Fitness	% of correctly classified data	Average number of variables per rule	Computation time (sec)
3	0.601156	58.4795	2.66	123
3	0.606936	59.0643	3.33	25
3	0.612717	59.0643	2.33	127

Table 7.6 Results for Liver disorder: triangular membership function

Number of Rules	Fitness	% of correctly classified data	Average number of variables per rule	Computation time (sec)
3	**0.630058**	**64.3275**	**4.33**	**141**
3	0.647399	63.7427	5	144
3	0.630058	62.5731	3.33	140
2	0.580913	61.165	2.667	128
2	0.585062	60.194	3	124
2	0.589212	61.165	2.33	124

Result: As seen from the table for liver disorder problem the triangular membership function works well.

Hence the best evolved three rules fuzzy inference system with classification performance of 64.3275% is as below. The Fitness Vs generation plot and the six input membership function plots are shown in figure 7.6 and figure 7.7 respectively.

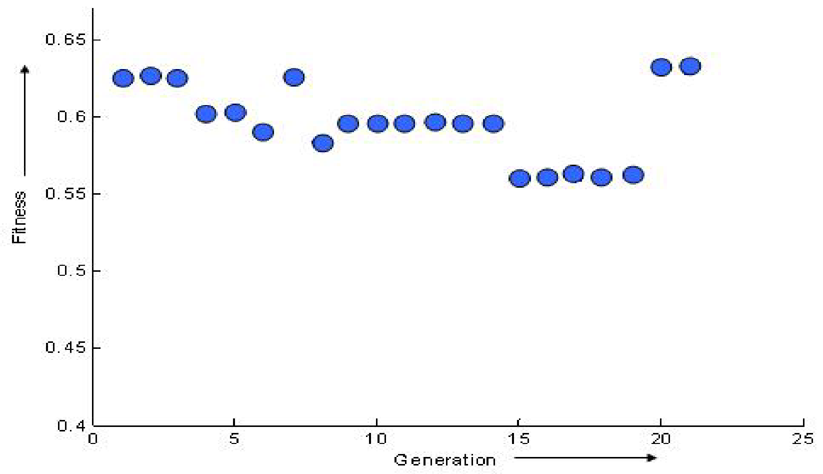
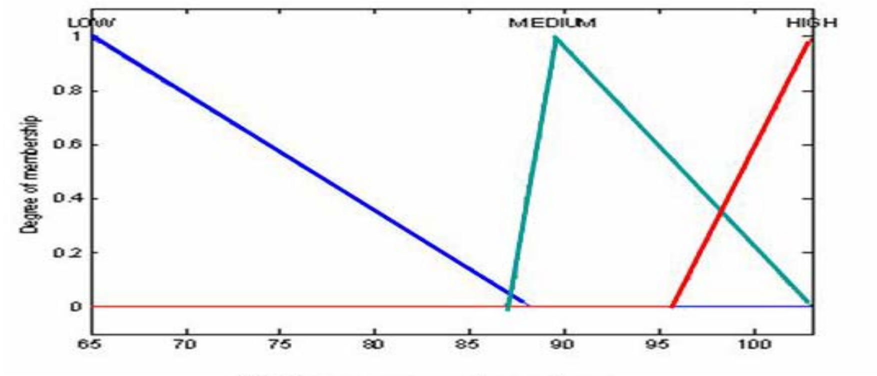
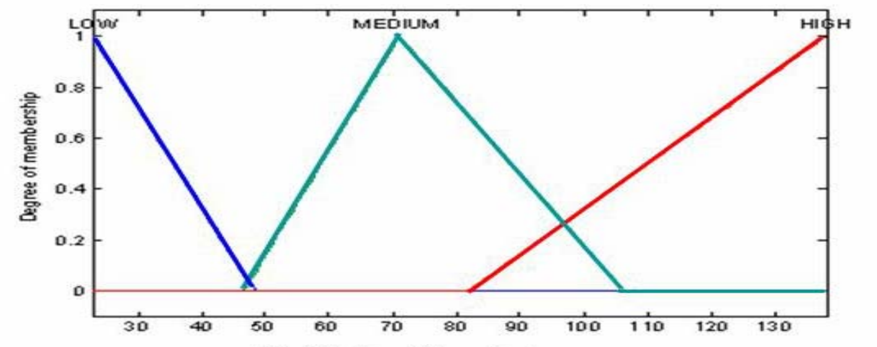


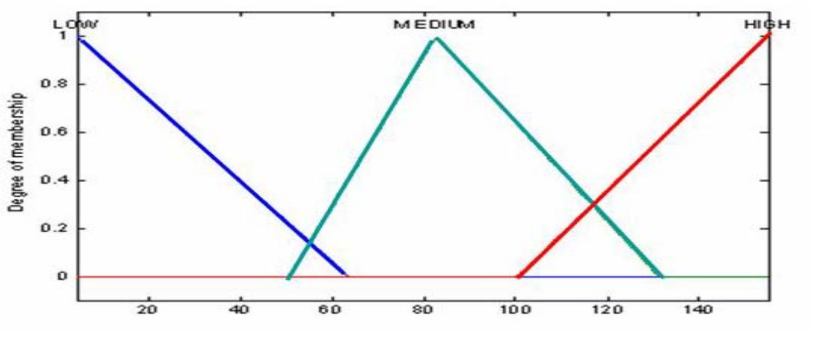
Figure 7.6 Fitness Vs Generation plot



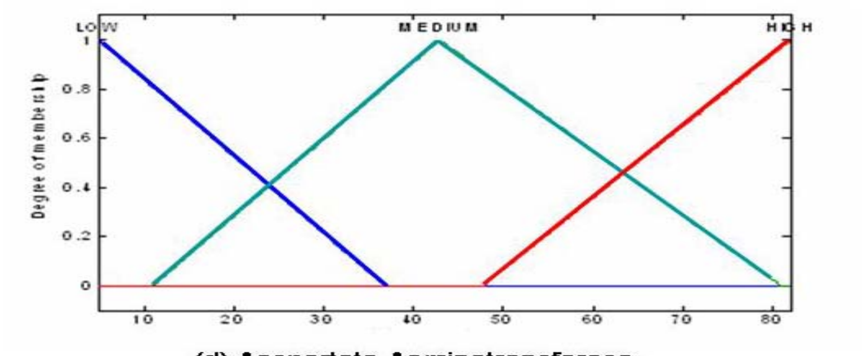
(a) Mean corpuscular volume



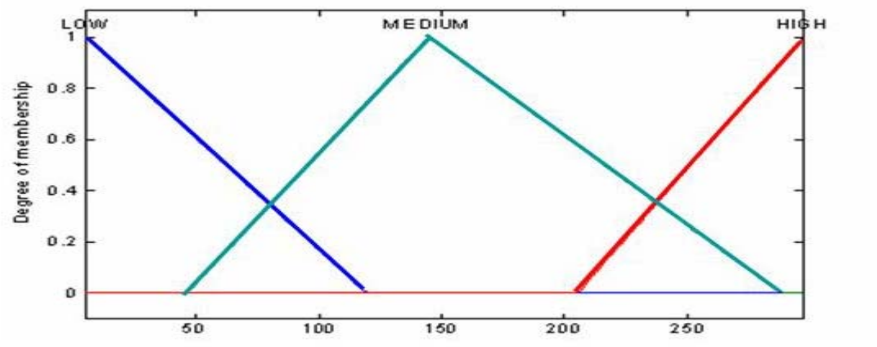
(b) Alkaline Phosphotase



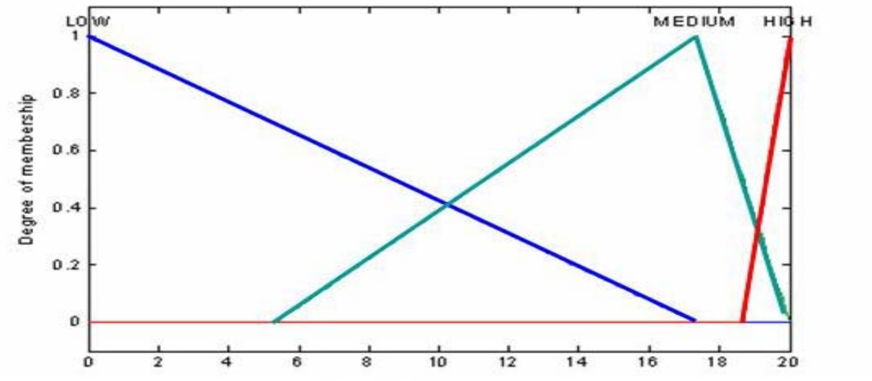
(c) Alamine Aminotransferase



(d) Aaspartate Aaminotransferase



(e) G-G transpeptidase



(f) Drinks

Figure 7.7 Optimized input membership function plots for Liver disorder3 triangular MF and 3 rule base

The optimized 3 rule base

1. If *(Alkaline phosphotase is LOW)* and *(Aspartate aminotransferase is HIGH)* and *(Drinks is MEDIUM)* then *(Diagnosis is LIVER DISORDER)*

2. If *(Mean Corpuscular Volume is HIGH)* and *(Alkaline phosphotase is HIGH)* and *(Alamine aminotransferase is MEDIUM)* and *(G-G transpeptidase is LOW)* and *(Drinks is MEDIUM)* then *(Diagnosis is LIVER DISORDER)*

3. If *(Mean Corpuscular Volume is HIGH)* and *(Alkaline phosphotase is MEDIUM)* and *(Aspartate aminotransferase is HIGH) and (G-G transpeptidase is MEDIUM)* and *(Drinks is HIGH)* then *(Diagnosis is LIVER DISORDER)*

4. else *Diagnosis is (HEALTHY LIVER)*

7.2.2 Evolved fuzzy system for five triangular membership functions and three rule base.

Table 7.7 Results for Liver disorder: five triangular MF and 3 rule base

Number of Rules	Fitness	% of correctly classified data	Average number of variables per rule	Computation time (sec)
3	0.589595	60.2339	4.66	155
3	0.630058	60.2339	4.33	149
3	**0.630058**	**64.3275**	**3**	**152**

Hence the best evolved three rules fuzzy inference system with classification performance of 64.3275 is as below. The Fitness Vs generation plot and the six input membership function plots are shown in figure 7.8 and figure 7.9 respectively.

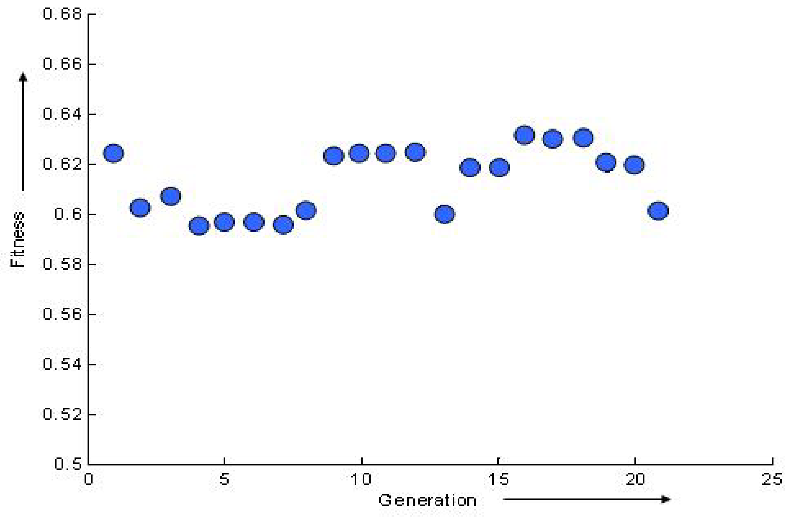
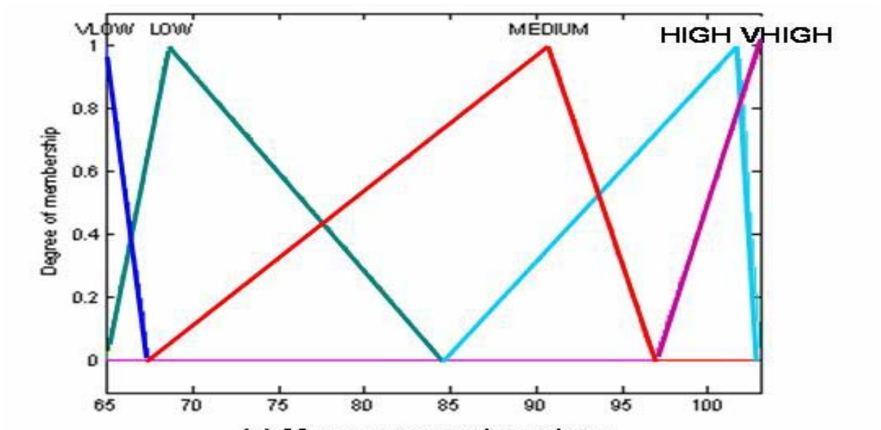
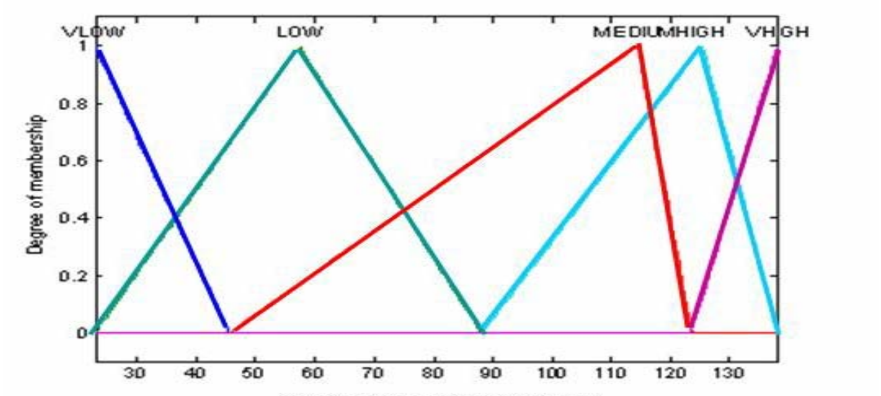


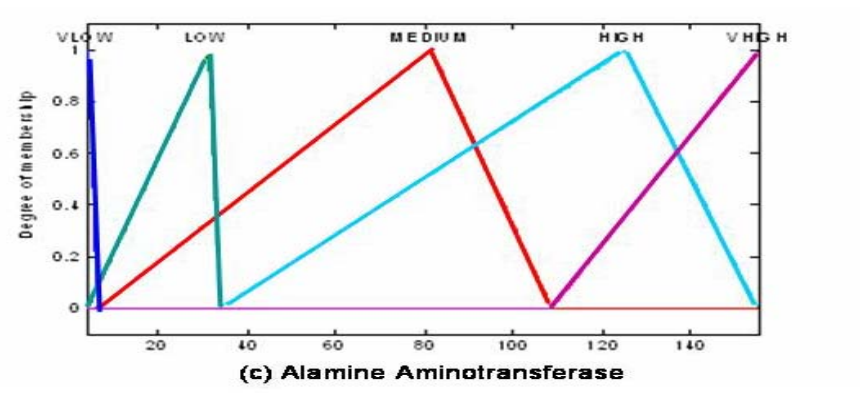
Figure 7.8 Fitness Vs generation plot

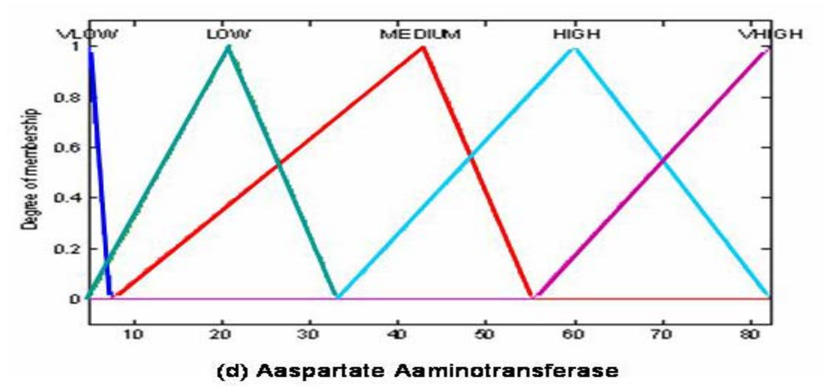


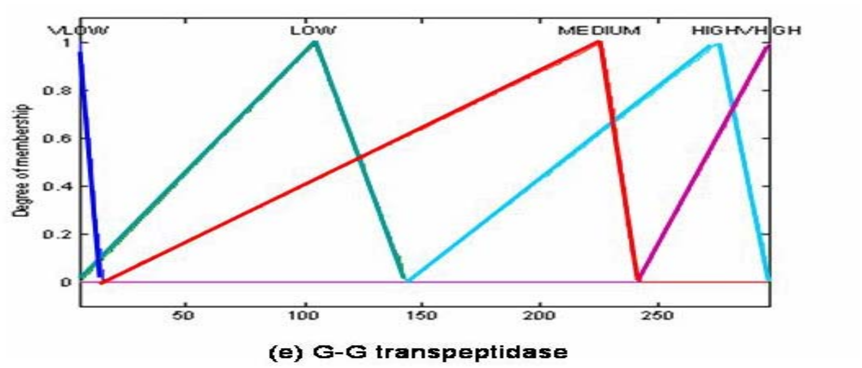
(a) Mean corpuscular volume



(b) Alkaline Phosphotase







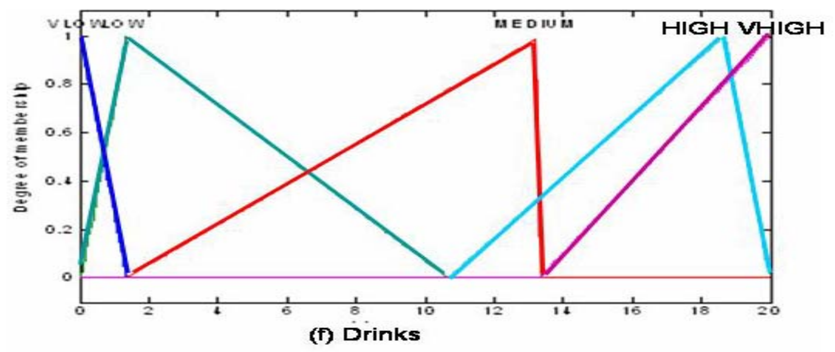


Figure 7.9 Optimized input membership function plots for Liver disorder -5 triangular MF and 3 rule base

The optimized 3 rule base

1. If *(Mean Corpuscular Volume is HIGH)* and *(Alkaline phosphotase is VLOW)* and *(G-G transpeptidase is VLOW)* then *(Diagnosis is LIVER DISORDER)*

2. If *(Mean Corpuscular Volume is VHIGH)* and *(Aspartate aminotransferase is MEDIUM)* and *(G-G transpeptidase is VHIGH)* then *(Diagnosis is LIVER DISORDER)*

3. If *(Alamine aminotransferase is VLOW)* and *(Aspartate aminotransferase is VHIGH)* and *(Drinks is VHIGH)* then *(Diagnosis is LIVER DISORDER)*

4. else *(Diagnosis is LIVER HEALTHY)*

7.2.3 Evolved fuzzy inference system for five triangular membership functions and five rule base.

Table 7.8 Results for Liver disorder: five triangular MF and 5 rule base

Number of Rules	Fitness	% of correctly classified data	Average number of variables per rule	Computation time (sec)
5	0.624277	66.6667	4	166
5	**0.630058**	**67.2515**	**3.2**	**171**
5	0.641618	63.7427	3.6	156

The Fitness Vs generation plot and the six input membership function plots of the best evolved five rule fuzzy system are shown in figure 7.10 and figure 7.11 respectively.

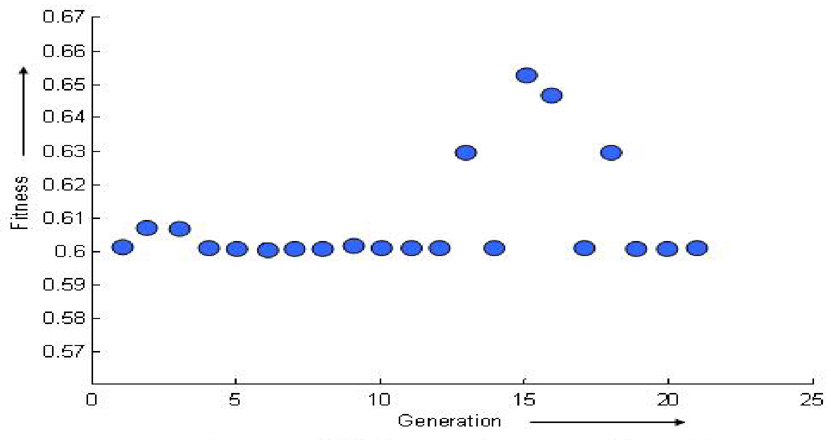
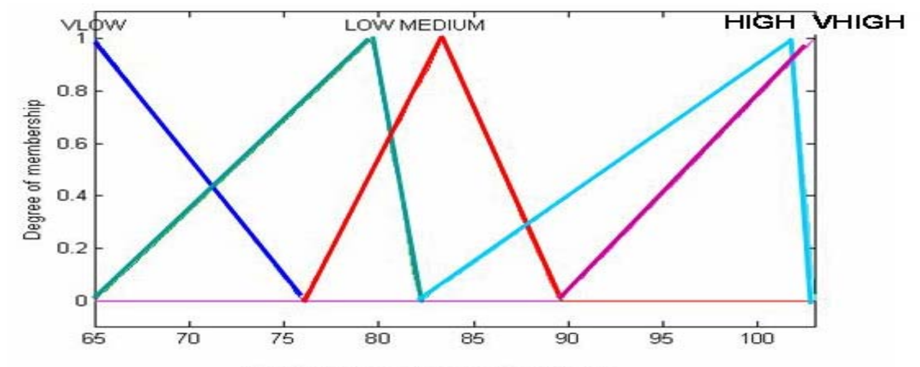
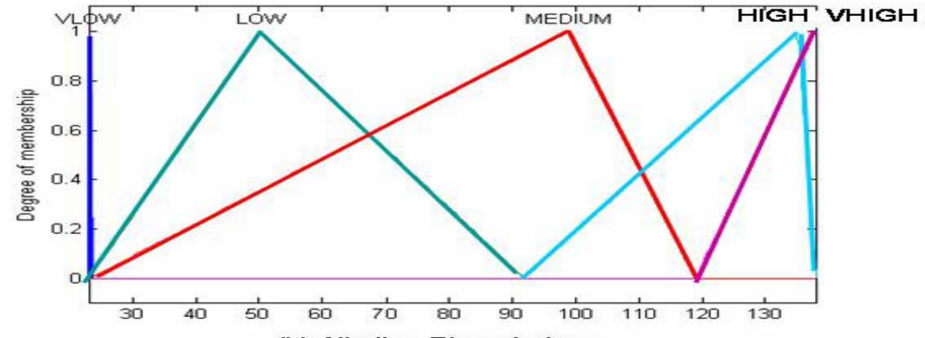


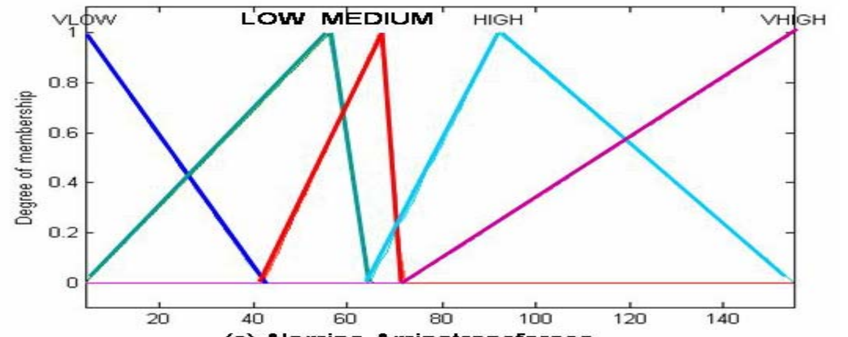
Figure 7.10 Fitness Vs generation plot



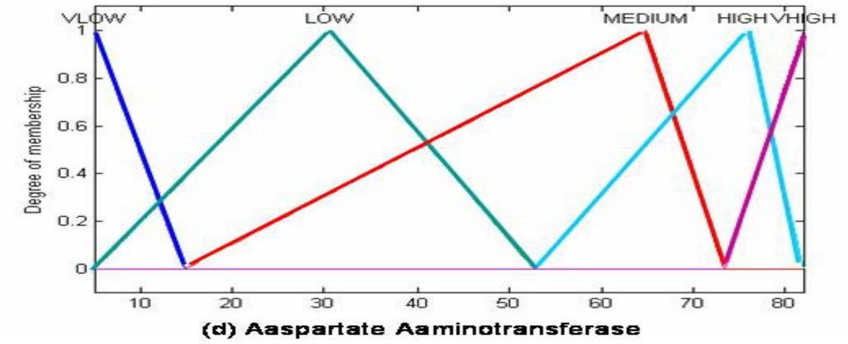
(a) Mean corpuscular volume



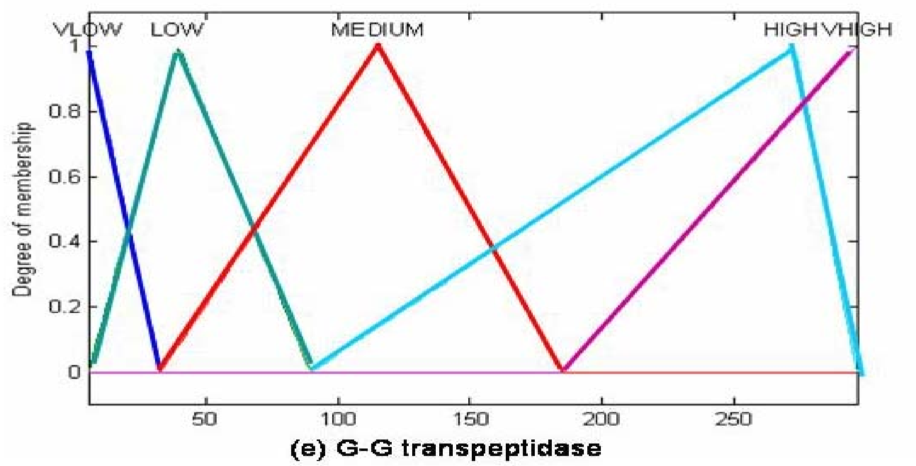
(b) Alkaline Phosphotase



(c) Alamine Aminotransferase



(d) Aaspartate Aaminotransferase



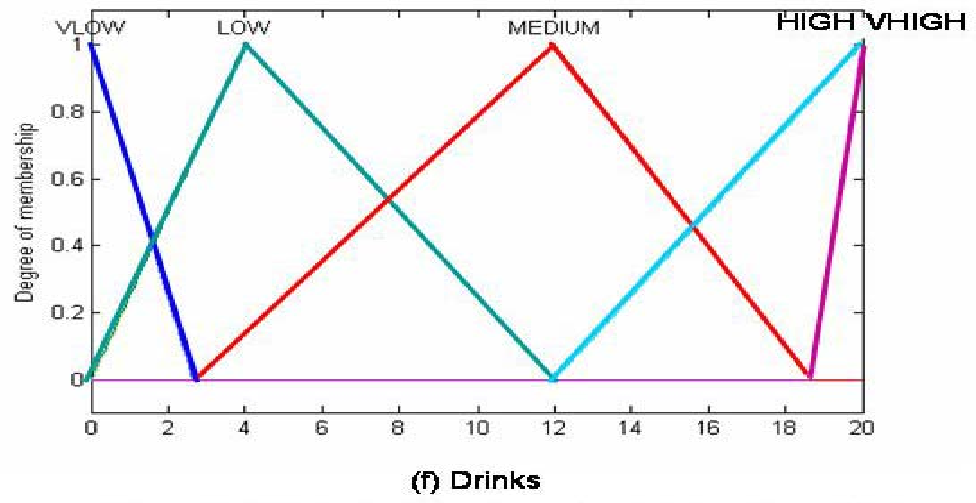


Figure 7.11 Optimized input membership function plots for Liver disorder 5 triangular MF and 5 rule base

Optimized five rule inference

1. If *(Mean Corpuscular Volume is VHIGH)* and *(Alkaline phosphotase is LOW)* and *(Alamine aminotransferase is VLOW)* then *(Diagnosis is LIVER DISORDER)*

2. If *(Mean Corpuscular Volume is HIGH)* and *(Alkaline phosphotase is VLOW)* and *(Aspartate aminotransferase is HIGH)* then *(Diagnosis is LIVER DISORDER)*

3. If *(Alamine aminotransferase is LOW)* and *(G-G transpeptidase is LOW)* and *(Drinks is VLOW)* then *(Diagnosis is LIVER DISORDER)*

4. If *(Mean Corpuscular Volume is VHIGH)* and *(Alkaline phosphotase is HIGH)* and *(Aspartate aminotransferase is VHIGH)* and *(G-G transpeptidase is LOW)* then *(Diagnosis is LIVER DISORDER)*

5. If *(Alkaline phosphotase is MEDIUM)* and *(G-G transpeptidase is VLOW)* and *(Drinks is MEDIUM)* then *(Diagnosis is LIVER DISORDER)*

6. else *(Diagnosis is LIVER HEALTHY).*

7.2.4 Testing the evolved system-diagnostic confidence

Suppose the following input data of patient no #15 from the UCI Liver dataset is given to the evolved fuzzy system of topic 7.2.3.

v_1	v_2	v_3	v_4	v_5	v_6
96	67	29	20	11	0.5

The membership value of each variable is then computed in accordance with the evolved database of 7.1.2.

Table 7.9 Fuzzification using the evolved database for liver disorder

	υ_1	υ_2	υ_3	υ_4	υ_5	υ_6
μ_{VLow}	0	0	0.35	0	0.55	0.8
μ_{Low}	0	0.6	0.5	0.56	0.3	0.1
μ_{Medium}	0	0.55	0	0.1	0	0
μ_{High}	0.7	0	0	0	0	0
μ_{VHigh}	0.45	0	0	0	0	0

Rule 1: $Min(0.45\ ,0.6\ ,0.35\ ) = 0.35$

Rule 2: $Min(0.7\ ,0\ ,0\ ) = 0$

Rule 3: $Min(0.5\ ,\ 0.3,\ 0.8\ ) = 0.3$

Rule 4: $Min(0.45, 0\ ,0,\ 0.3\ ) = 0$

Rule 5: $Min(0.55, 0.55\ ,\ 0\ ) = 0$

Each of the above rules has rule weight of 1 and the malignant rule has rule weight of 0.25. The output functions are singletons at 2 and 4 for benign and malignant respectively.

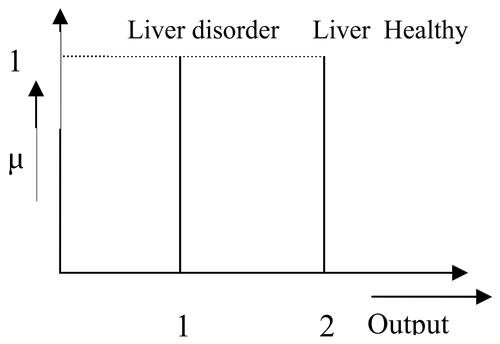


Figure 7.12 Output singletons membership functions

Weighted average for defuzzification:

$$\text{Defuzzified output} = \frac{(0.35 \times 1) + (0.3 \times 1) + (0.25 \times 2)}{0.35 + 0.3 + 0.25} = 1.28$$

Discussion:

- ❖ This appraisal (defuzzified) output from the evolved fuzzy system is then passed on to the threshold system having a threshold of 1.5. As 1.28 is less than 1.5, we have the diagnosis that the patient has liver disorder. *This is correct in relation to the output given in the output liver dataset.*
- ❖ The best diagnostic classification performance obtained is 67.2515.
- ❖ Five triangular input membership functions and five rule base fuzzy system work best for this problem.

7.3 Pima-Indian diabetes diagnosis

7.3.1 The best membership function

The results for trapezoidal and triangular membership function have been tabulated as in table 7.10 and table 7.11 respectively.

Table 7.10 Results for diabetes: trapezoidal membership function

Number of Rules	Fitness	% of correctly classified data	Average number of variables per

			rule
3	0.721354	76.7624	4.66
3	0.708333	74.1567	4
3	0.700521	74.9347	4.66

Table 7.11 Results for diabetes: triangular membership function

Number of Rules	Fitness	% of correctly classified data	Average number of variables per rule	Computation time
3	**0.723958**	**78.0679**	**4.33**	**284**
3	0.713542	77.0235	4	285
3	0.721354	75.4569	4.33	287
2	0.723958	77.0235	4.5	275
2	0.714783	77.0235	6	278
2	0.71875	77.5457	3.5	281

Result: As seen from the table 7.8 and table 7.9 for pima Indian diabetes problem the triangular membership function works better.

Hence the best evolved three rule fuzzy inference system with classification performance of 78.0679% is as below. The Fitness Vs generation plot and the eight input membership function plots are shown in figure 7.13 and figure 7.14 respectively.

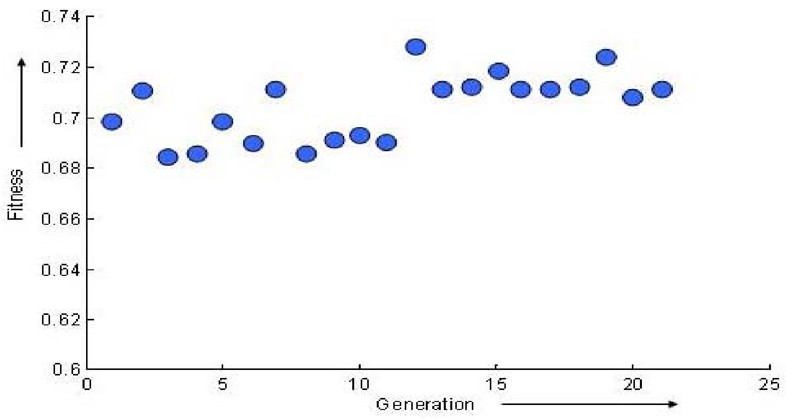
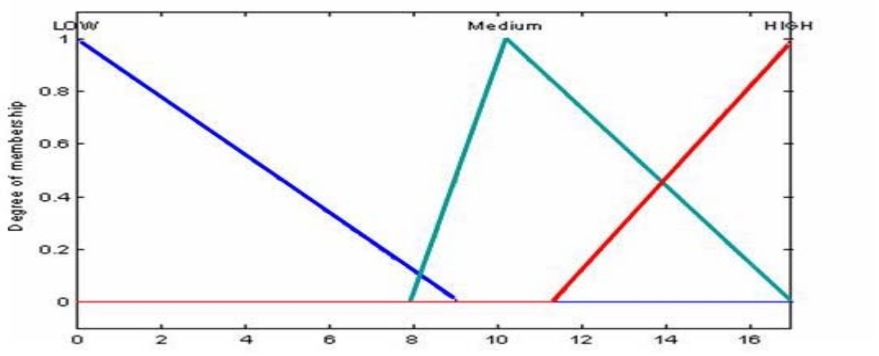
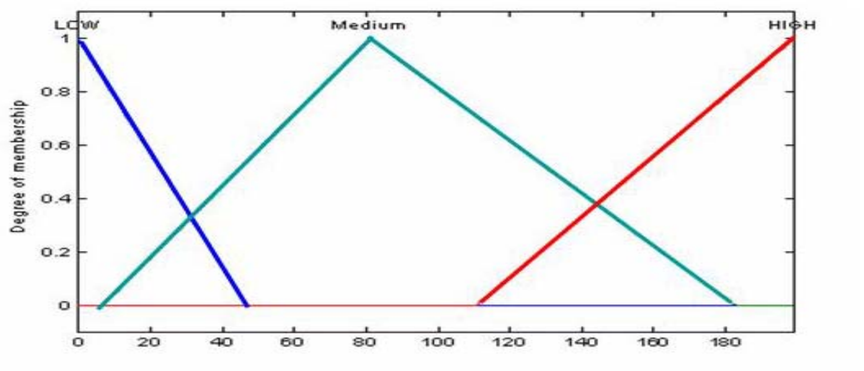


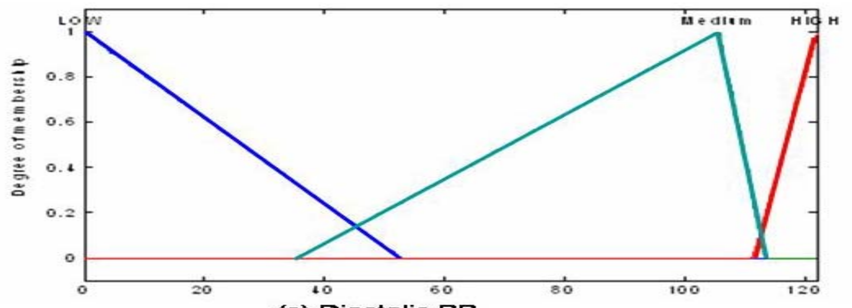
Figure 7.13 Fitness Vs Generation plot



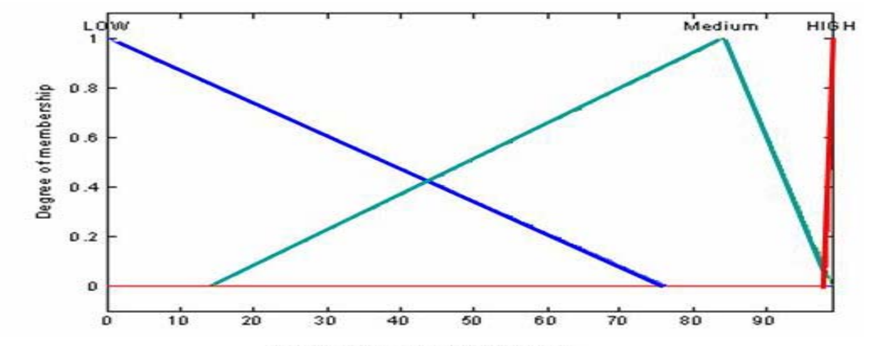
(a) Number of times pregnant



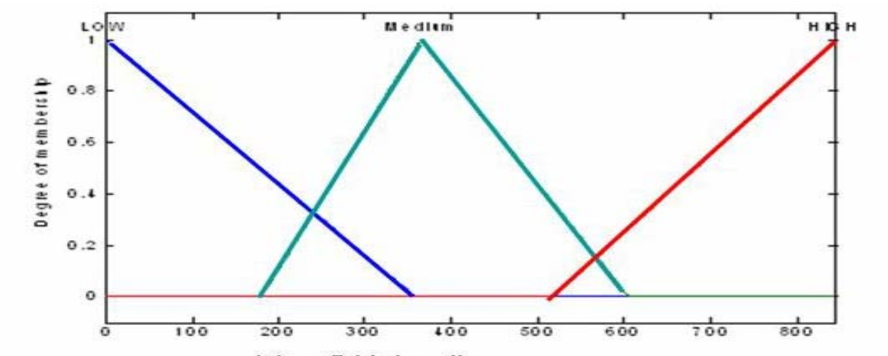
(b) Plasma glucose concentration



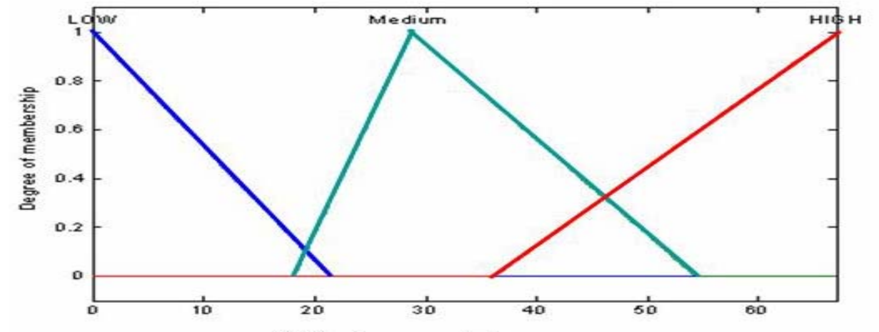
(c) Diastolic BP



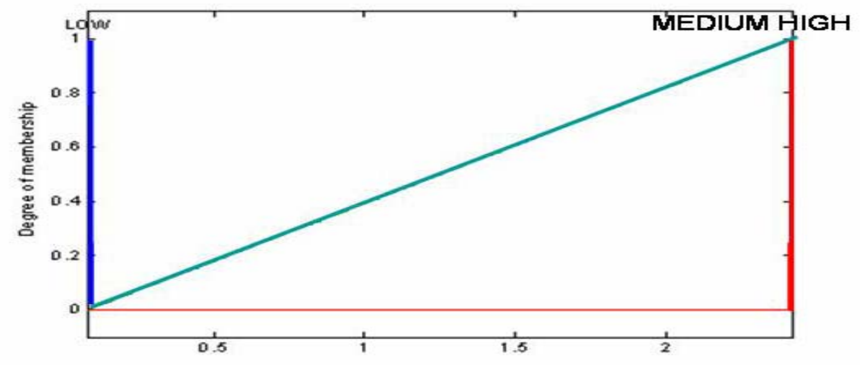
(d) Skin fold thickness



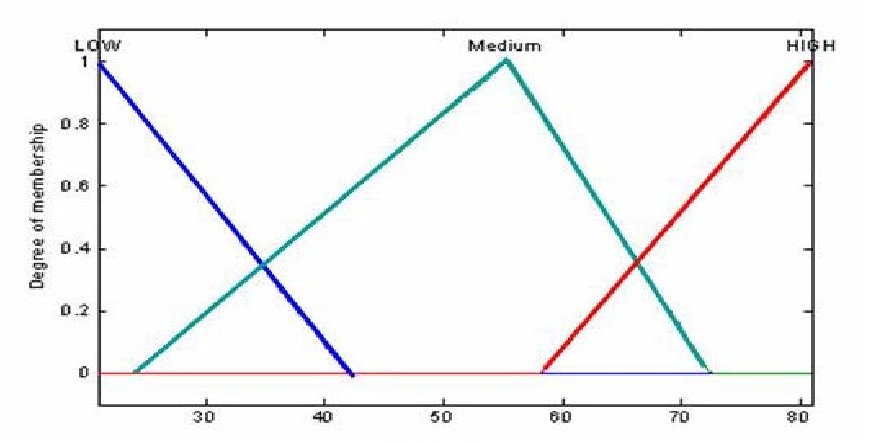
(e) 2 Hr insulin



(f) Body mass index



(g) Diabetes Pedigree function



(h) Age

Figure 7.14 Optimized input membership function plots for diabetes -3 triangular MF and 3 rule base

Optimized three rule base

1. If *(Plasma glucose concentration is LOW)* and *(Diabetes pedigree fun is Medium)* then *(Diagnosis is DIABETES POSITIVE)*

2. If *(Plasma glucose concentration is Medium)* and *(Diastolic BP is Medium)* and *(2Hr serum insulin is LOW)* and *(Body Mass Index is Medium)* and *(Diabetes pedigree fun is Medium)* and *(Age is Medium)* then *(Diagnosis is DIABETES POSITIVE)*

3. If *(Number of times pregnant is LOW)* and *(Skin fold thickness is Medium)* and *(2Hr serum insulin is Medium)* and *(Body Mass Index is LOW)* and *(Age is Medium)* then *(Diagnosis is DIABETES POSITIVE)*

4. else *(Diagnosis is DIABETES NEGATIVE)*

7.3.2 Evolved fuzzy system for five triangular membership functions and three rule base.

Table : 7.12 Results for diabetes :five triangular MF and 3 rule base

Number of Rules	Fitness	% of correctly classified data	Average number of variables per rule	Computation time
3	0.721354	78.3290	4.66	452
3	0.710938	77.8068	4.66	450

3	**0.716146**	**79.3734**	**4.33**	**500**

Thus the best evolved three rule fuzzy inference system with classification performance of 79.3734% is as below. The Fitness Vs generation plot and the eight input membership function plots are shown in figure 7.15 and figure 7.16 respectively.

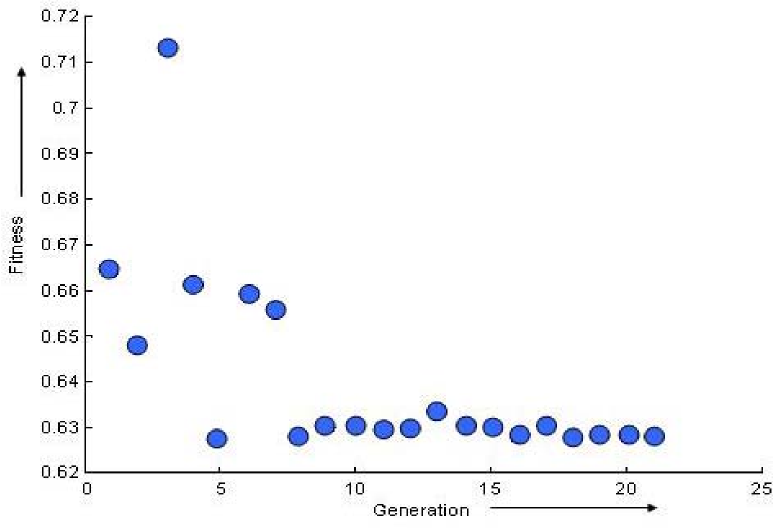
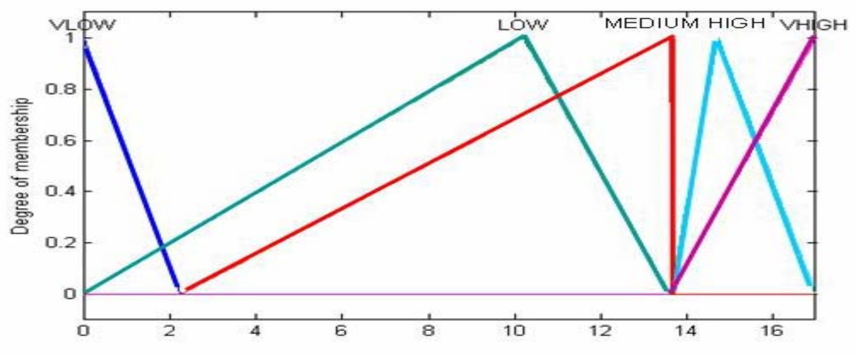
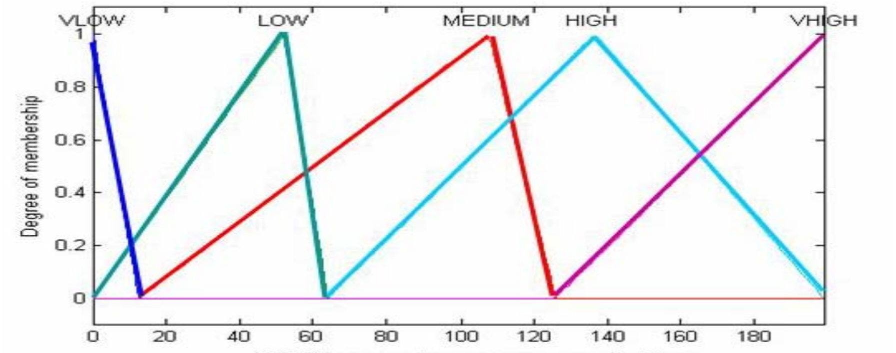


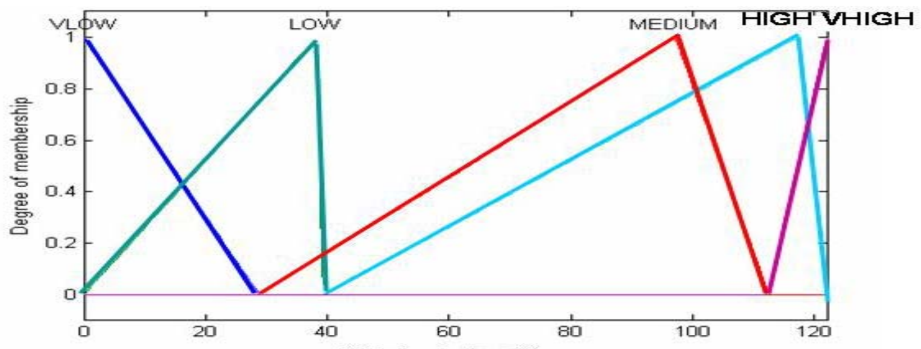
Figure 7.15 Fitness Vs generation plot



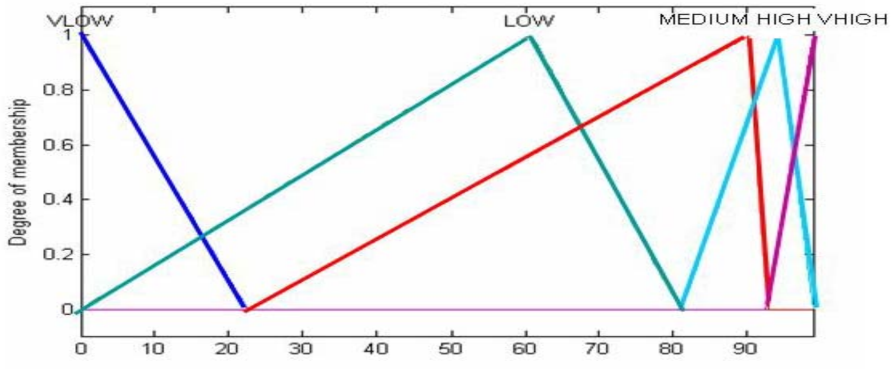
(a) Number of times pregnant



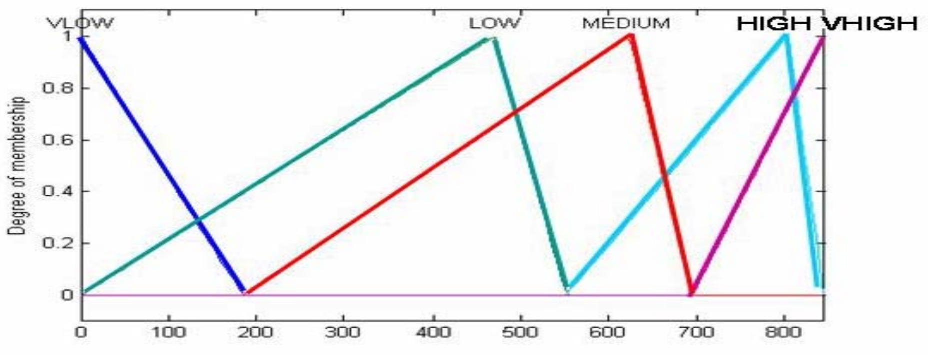
(b) Plasma glucose concentration



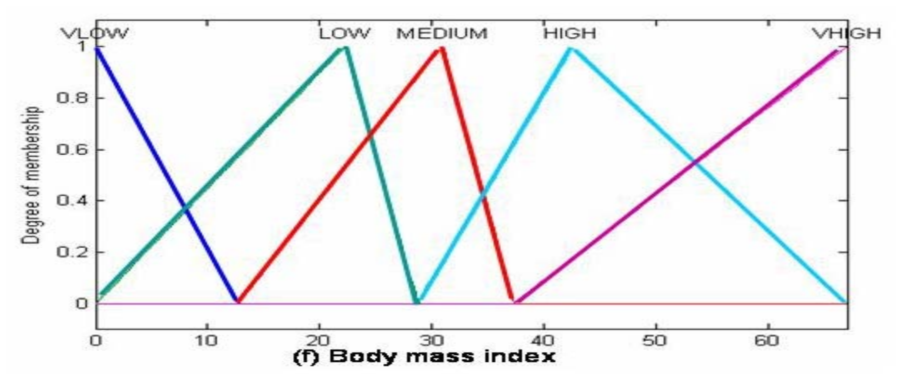
(c) Diastolic BP

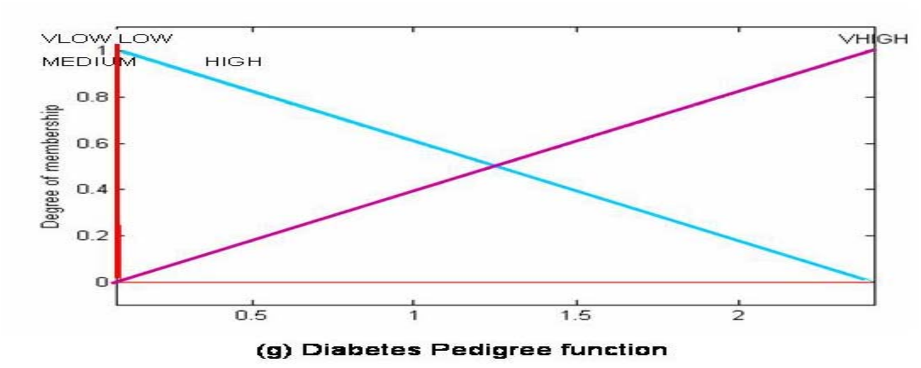


(d) Skin fold thickness



(e) 2 Hr insulin





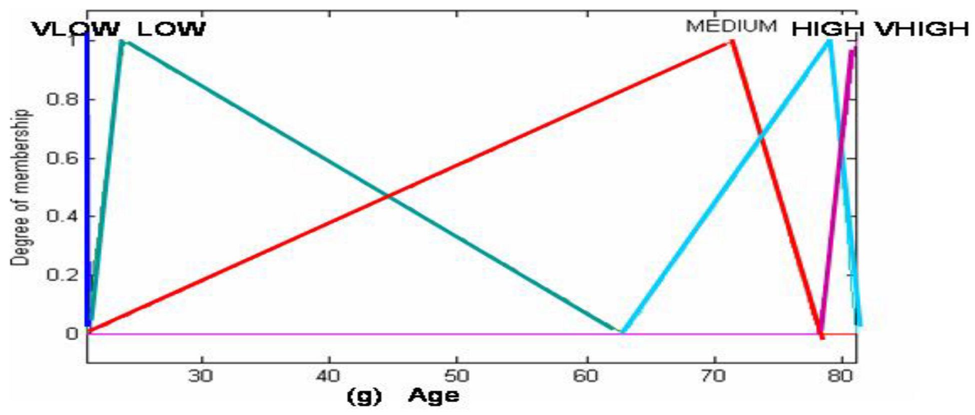


Figure 7.16 Optimized input membership function plots for diabetes- 5 triangular MF and 3 rule base

The optimized three rule inference

1. If *(Number of times pregnant is MEDIUM)* and *(Skin fold thickness is MEDIUM)* and *(Body Mass Index is VLOW)* and *(Diabetes pedigree fun is LOW)* and *(Age is LOW)* then *(Diagnosis is DIABETES POSITIVE)*

2. If *(Plasma glucose concentration is VHIGH)* and *(Diastolic BP is VHIGH)* and *(Skin fold thickness is LOW)* and *(Body Mass Index is VHIGH)* and *(Diabetes pedigree fun is LOW)* then *(Diagnosis is DIABETES POSITIVE)*

3. If *(Plasma glucose concentration is VHIGH)* and *(Diastolic BP is MEDIUM)* and *(Body Mass Index is HIGH)* then *(Diagnosis is DIABETES POSITIVE)*

4. else (Diagnosis is DIABETES NEGATIVE)

7.3.3 Testing the evolved system-diagnostic confidence

Suppose the following input data of patient no #12 from the pima-Indian diabetes dataset is given to the evolved fuzzy system of topic 7.3.2

v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8
10	168	74	0	0	38	0.537	34

The membership value of each variable is then computed in accordance with the evolved database of 7.3.2

Table 7.13 Fuzzification using the evolved database for Pima-Indian diabetes

	υ_1	υ_2	υ_3	υ_4	υ_5	υ_6	υ_7	υ_8
μ_{VLow}	0	0	0	1	1	0	0	0
μ_{Low}	1	0	0	0	0	0	0	0.75
μ_{Medium}	0.7	0	0.65	0	0	0	0	0.3
μ_{High}	0	0.5	0.45	0	0	0.65	0.8	0
μ_{VHigh}	0	0.58	0	0	0	0.02	0.2	0

Rule 1: $Min(0.7\ ,0\ ,0\ ,0\ ,0.75\ ) = 0$

Rule 2: $Min(0.58\ ,0\ ,0\ ,0.02\ ,0\ ) = 0$

Rule 3: $Min(0.58\ ,\ 0.65, 0.65\ ) = 0.58$

Each of the above rules has rule weight of 1 and the else rule has rule weight of 0.25. The output functions are singletons at 1 and 2 for diabetes positive and negative respectively.

Weighted average for defuzzification: $\dfrac{(0.58 \times 1) + (0.25 \times 2)}{0.58 + 0.25} = 1.3$

Discussion:

This appraisal (defuzzified) output from the evolved fuzzy system is then passed on to the threshold system having a threshold of 1.5. As 1.3 is less than 1.5, we have the diagnosis that the patient suffers from diabetes. *This is correct in relation to the output given in the diabetes output dataset*.

7.3.4 Evolved Fuzzy system for five triangular membership functions and five rule base.

Table: 7.14 Results for diabetes: five triangular MF and 5 rule base

Number of Rules	Fitness	% of correctly classified data	Average number of variables per rule	Computation time (sec)
5	0.710938	77.0235	4.2	484
5	0.721354	77.5457	4.2	509
5	**0.721354**	**77.8068**	**5**	**481**

Thus the best evolved five rules fuzzy inference system with classification performance of 77.8068% is as below. The Fitness Vs generation plot and the nine input membership function plots are shown in figure 7.17 and figure 7.18 respectively.

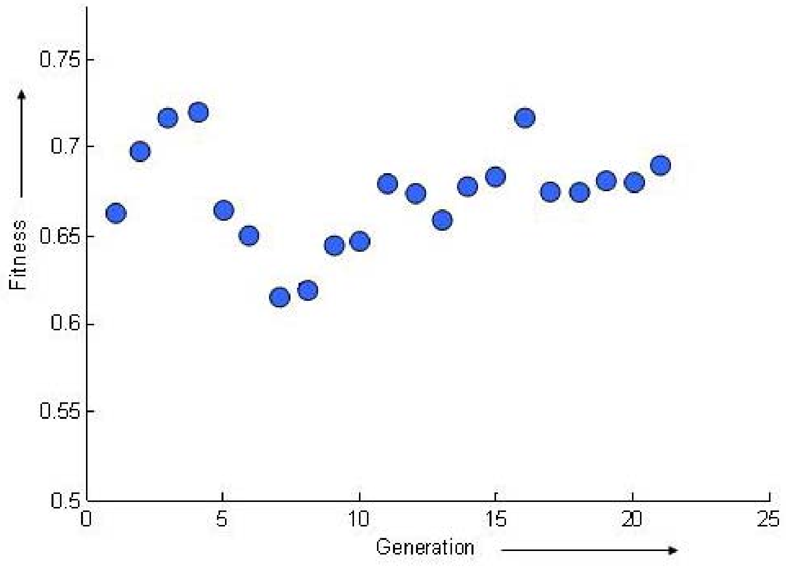
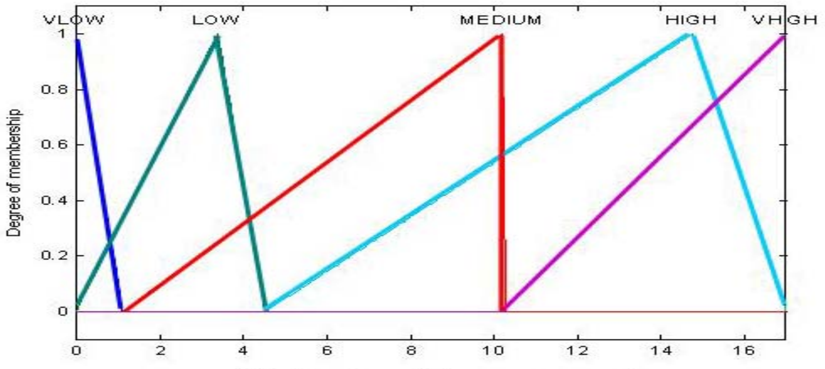
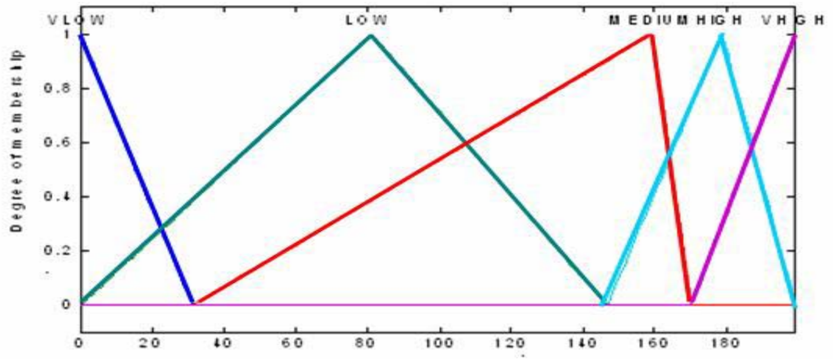


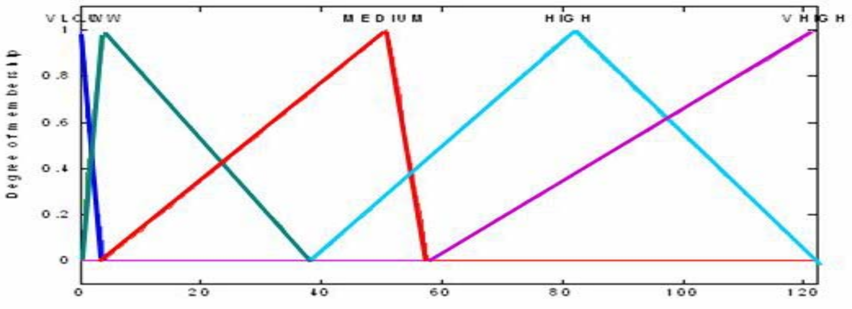
Figure 7.17 Fitness Vs generation plot



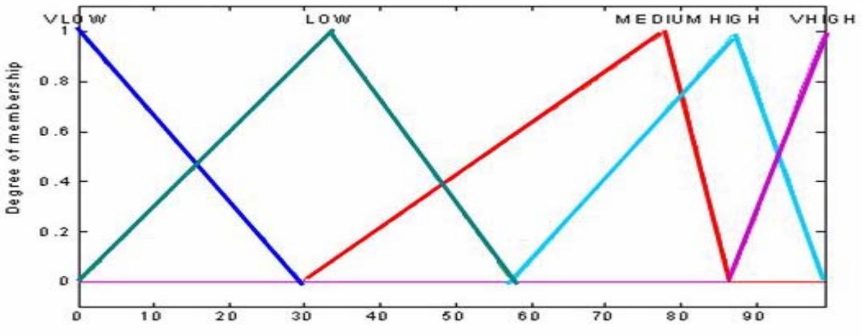
(a) Number of times pregnant



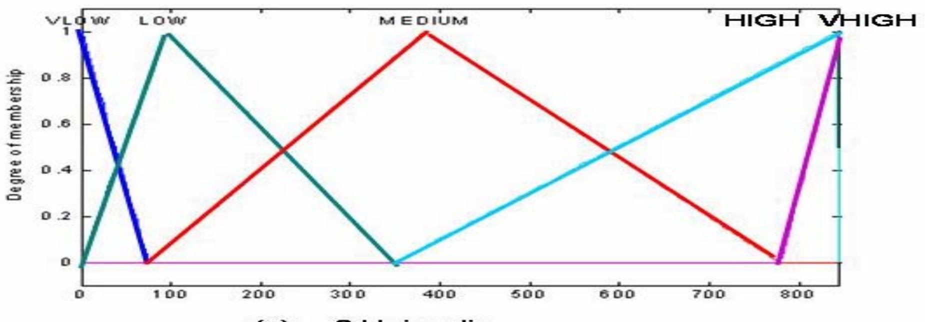
(b) Plasma glucose concentration



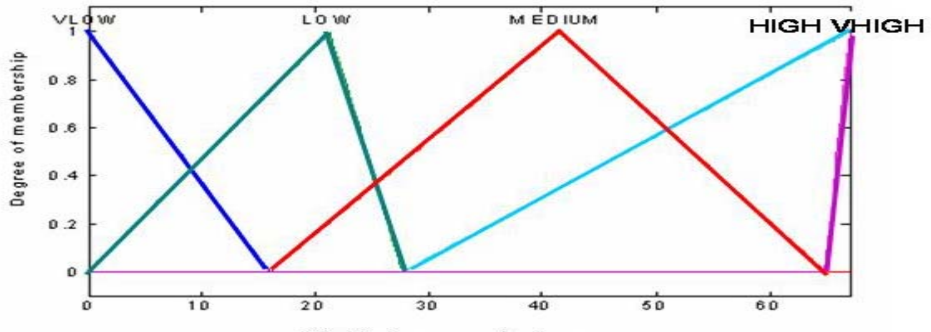
(c) Diastolic BP



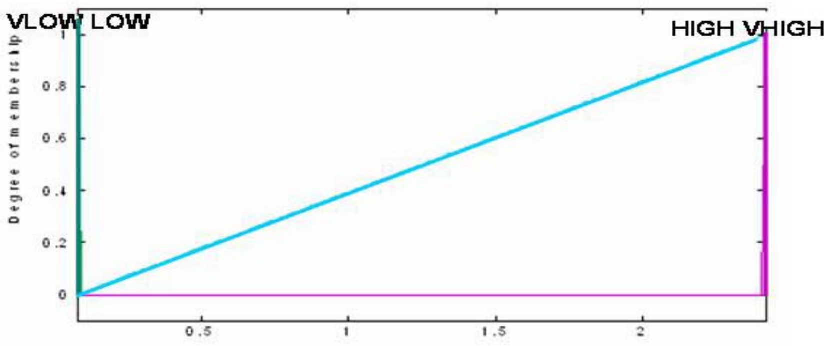
(d) Skin fold thickness



(e) 2 Hr insulin



(F) Body mass index



(g) Diabetes Pedigree function

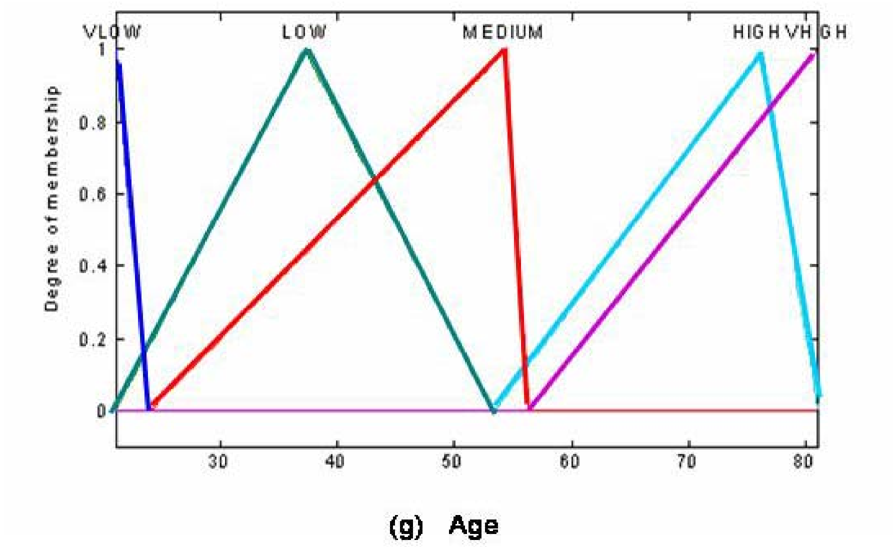


Figure 7.18 Optimized input membership function plots for diabetes 5 triangular MF and 5 rule base

Optimized five rule base:

1. If (Number of times pregnant is HIGH) and (Plasma glucose concentration is LOW) and (Diastolic BP is VHIGH) and (Skin fold thickness is VLOW) and (Age is VHIGH) then (Diagnosis is DIABETES POSITIVE)

2. If (Number of times pregnant is LOW) and (Plasma glucose concentration is MEDIUM) and (2Hr serum insulin is LOW) and (Body Mass Index is HIGH) and (Diabetes pedigree fun is VLOW) then (Diagnosis is DIABETES POSITIVE)

3. If (Number of times pregnant is HIGH) and (Skin fold thickness is LOW) and (Diabetes pedigree fun is LOW) then (Diagnosis is DIABETES POSITIVE)

4. If (Diastolic BP is HIGH) and (Diabetes pedigree fun is VLOW) and (Age is VHIGH) then (Diagnosis is DIABETES POSITIVE)

5. If (Number of times pregnant is LOW) and (Plasma glucose concentration is MEDIUM) and (Diastolic BP is HIGH) and (Skin fold thickness is VLOW) and (Diabetes pedigree fun is LOW) then (Diagnosis is DIABETES POSITIVE)

6. else (Diagnosis is DIABETES NEGATIVE)

Discussion:

- The best diagnostic classification performance obtained for this problem is 79.3734%.
- Five triangular input membership function and three rule base fuzzy system work best for this problem.

CHAPTER-VIII

CONCLUSION AND FUTURE SCOPE

CONCLUSION

The optimized fuzzy inference system, evolving both the membership functions and the rules has been obtained for breast cancer, liver disorder and diabetes diagnosis problems. The best evolved system has also been tested for correct diagnosis, giving excellent classification performance.

The resulted evolved systems exhibit both characteristics outlined in Chapter 1: they attain high classification performance with the possibility of attributing a confidence measure to the output diagnosis and the resulting systems involve a few simple rules, and are therefore interpretable.

Trapezoidal membership functions works best for the Wisconsin breast cancer problem, while triangular membership functions work best for liver disorder and diabetes problem.

The diagnostic classification performance for breast cancer diagnosis is better as compared to the work done by R. Setiono [20], Taha and Ghosh [23] .

The diagnostic classification performance for pima-indian diabetes diagnosis is better as compared to work done by W.Au et al [1].

The problem encountered is that as the length of the genome is large , the computation time is more. Moreover as the fitness function just takes into account the classification performance, and does not penalize systems with a large number of variables per rule, rules are more complicated , thus interpretability decreases.

FURTHER WORK

Further work on this project should concentrate on the following issues:

The experiments were constrained with the limited number of rules per system. It was kept fixed between 2 to 5. The system can be worked with more number of rules and its effect on the fitness function can be seen.

The active rules in the experiments diagnose benignity, liver disorder and diabetes tested positive with the default diagnosis being malignancy, no liver disorder and diabetes tested negative. It can be sought to find out what would happen if this were reversed, i.e. could better systems be evolved with benignity, liver disorder and diabetes tested positive as the default diagnosis?

The fitness function used in this project takes into account just the classification performance. It can be changed taking into consideration two criteria:

- ❖ The interpretability-penalizing systems with a large number of variables per rule.
- ❖ The quadratic error (difference between the appraisal value and the correct diagnosis given by the database). It adds selection pressure towards systems with low quadratic error.

What would be the effect of the above functions on the correct diagnosis? This would trade off performance for better interpretability.

One can investigate the fuzzy genetic approach presented here for other complex diagnosis problems and test how well it can diagnose the disease.

REFERENCES

1. Au W.-H and Chan K.C.C., "*Classification with Degree of Membership: A Fuzzy Approach,"* in Proc. of the 1st IEEE Int'l Conference on Data mining. 2001. pages : 35-42

2. Bennett K. P. and Mangasarian O. L.. *Neural network training via linear programming*. In P. M. Pardalos, editor, Advances in Optimization and Parallel Computing, pages 56–57. Elsevier Science, 1992.

3. Cordón O., Herrera F., Hoffmann F. and Magdalena L. (2001). *Genetic Fuzzy Systems: Evolutionary Tuning and Learning of Fuzzy Knowledge Bases.* Advances in Fuzzy Systems– Applications and Theory Vol. 19, World Scientific.

4. Fidelis M.V.; Lopes H.S.; Freitas, A.A.; *Discovering comprehensible classification rules with a genetic algorithm Evolutionary Computation* ,2000 Proceedings of the 2000 Congress Volume 1, 16-19 July 2000 Page(s):805 - 810 Vol.1

5. Fogel L.J., Owens A. J., and Walsh. M. J. *Artificial Intelligence through Simulated Evolution*. John Wiley & Sons, New York, 1966.

6. Goldberg. D.E. *"Genetic Algorithms in Search, Optimization and Machine Learning"* Addison-Wesley, 1989

7. Gopal M. *Digital control and State variable Methods*. 2nd edition. Tata Mcgraw Hill publishing Ltd.

8. Holland. J. H. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, MI.

9. Jain R., Mazumdar J., and Moran W.. *Application of fuzzy-classifier system to coronary artery disease and breast cancer.* Australasian Physical and Engineering Sciences in Medicine, 21(3):141–147, September 1998.

10. Jang J.S.R, Sun C.T, Mizutani E. *Neuro Fuzzy and soft computing.* Pearson Education ,Inc. 2004

11. Nagrath I.J and Gopal. M. *Control system engineering* 3rd edition , New age international publishing company Ltd 2003

12. Paolo Dadone *Design Optimization of Fuzzy Logic Systems*. Doctor of Philosophy Dissertation ,2001

13. Pena-Reyes C.A. and Sipper M., " *Evolving fuzzy rules for breast cancer diagnosis,* " in Proceedings of 1998 International Symposium on Nonlinear theory and Applications, Lausanne, 1998 , Vol 2, oo.369-372, Presses Polytechniques et Universitaires Romandes.

14. Pena-Reyes C.A , Moshe Sipper .*A fuzzy-genetic approach to breast cancer diagnosis*. Artificial Intelligence in Medicine. Vol 17. 1999 131-155

15. Peña-Reyes C. A. and Sipper M. *Evolutionary computation in medicine: An overview*. Artificial Intelligence in Medicine, 19(1):1–23, May 2000.

16. Rajasekaran S. and Vijayalakshmi Pai G.A : *Neural Networks, Fuzzy Logic and Genetic algorithm* , PHI 2003

17. Rotshtein A. , Rakytyanska H., (2000), *Genetic Algorithm for Fuzzy Logical Equations Solving in Diagnostic Expert Systems*, Lecture Notes in Computer Science, Vol. 2070, pp.349

18. Seshiah V., Diabetes care and research institute, Chennai. *Diabetes prevention: problem, promise and reality*. THE HINDU May 18, 2006 Pg 17.

19. Setiono R. *Generating concise and accurate classification rules for breast cancer diagnosis*. Artificial Intelligence in Medicine, 18(3):205 – 219, 2000.

20. Setiono. R. *Extracting rules from pruned neural networks for breast cancer diagnosis. Artificial Intelligence in Medicine*, 8:37–51, 1996.

21. Setino R. and Liu H., "*Symbolic representation of neural networks*," IEEE Computer, Vol. 29, no.3 ,pp 71-77, March 1996

22. Smith S.F., *A learning system based on genetic adaptive algorithms*, Doctoral Dissertation, Department of Computer Science, University of Pittsburgh, 1980.

23. Taha I. and Ghosh J. , " *Evaluation and ordering of rules extracted from feed forward networks"*, in Proceedings of the IEEE International Conference on Neural Networks, 1997 pp.221-226

24. Voget S., Kolonko M., *Multidimensional optimization with a fuzzy genetic algorithm*, J. Heuristics 4 (3) (1998) 221–244.

25. Wang, Z.S.; Chen, J.D.Z., (2000), *Robust ECG R-R wave peak detection using Evolutionary-Programming-based Fuzzy Inference System (EPFIS) and its application to assessing brain-gut interaction*, 1st Int. Conf. on Adv. In Medical Signal and Inf. Processing, (IEE Conf. Publ. No. 476), pp. 265-274

26. Yuhui Shi, Russell Eberhart and Yaobin Chen : *Implementation of Evolutionary Fuzzy Systems.* IEEE Transactions on Fuzzy Systems, VOL. 7, NO. 2, APRIL 1999- 109

27. Yager R .R, Filev D .P. *Essentials of Fuzzy Modeling and Control*. Wiley, 1994.

28. Zadeh. L. A. *Outline of a new approach to the analysis of complex systems and decision processes*. IEEE Transactions on Systems, Man and Cybernetics, SMC-3(1):28–44, January 1973

Websites:

29. www.mathworks.com

30. http://www.ics.uci.edu/~mlearn/databases/breast-cancer-wisconsin/breast-cancer-wisconsin.names

31. http://www.ics.uci.edu/~mlearn/databases/liver-disorders/bupa.data

32. http://www.ics.uci.edu/~mlearn/databases/pima-indians-diabetes/pima-indians-diabetes.names

33. www.cs.Berkeley.edu/~zadeh

APPENDIX

1. MAIN PROGRAM FOR DIAGNOSIS

```
% n = number of inputs
% s = The input dataset matrix.
% p = number showing which diagnosis to be done
% y = Min max value matrix of each input
% total = total patient data in each dataset.
% List_inp = character array of the name of inputs in the dataset
% name = The function called to get List
% vlb vub = The lower and upper bounds of each input
% bits = Number of bits required to code each input
% e = The cpu time taken to run the genetic loop
% x = The optimized genome
% k = The rule base from the optimized genome
% f = The MF parameters from the optimized genome

clear all
clc
disp(sprintf('Choose which diagnosis'));
disp('*---------------------------------------------*');
disp(sprintf('1) Breast cancer'));
disp(sprintf('2) Liver Disorder'));
disp(sprintf('3) Diabetes'));
p=input('Type the desired :')

% Load the particular dataset, the number of inputs , total data and
   input names and   outputs.
switch p,

case 1,
   s=load('C:\Documents and Settings\My Documents\uci\cancer.data');
   n =9;
   total=342;
   List_inp=name(1);
   List_out=nameo(1);

case 2,
   s=load('C:\Documents and Settings\My Documents\uci\liver.data');
   n =6;
   total=173;

  List_inp=name(2);
  List_out=nameo(2);

case 3,
   s=load('C:\DocumentsandSettings\My Documents\uci\diabetes.data');
   n=8;
   total=384;
   List_inp=name(3);
   List_out=nameo(3);
end;

y=[minmax(transpose(s))];
y=transpose(y);
if(p==1) y(:,1)=[];
end
y(:,n+1)=[];
y=transpose(y);

disp('%%%%%%%%%%%%%%%%%%%%%%')
disp('     The Inputs:      ')
disp('%%%%%%%%%%%%%%%%%%%%%%')
disp(List_inp)
pause;
disp('%%%%%%%%%%%%%%%%%%%%%%%')
disp('  The Input ranges:'   )
disp('%%%%%%%%%%%%%%%%%%%%%%%')
disp(y)
pause;
disp('%%%%%%%%%%%%%%%%%%%%%')
disp('  The Output classes:' )
disp('%%%%%%%%%%%%%%%%%%%%%')
disp(List_out)
pause;

% Generate the upper and lower bounds, bits required to code each input
vlb1=[];vub1=[];b1=[];r=[];
i=1;

while(i<n+1)
            y1(i,1)=y(i,2)-y(i,1);
            b1=round(log2(y1(i,1)));
            lb1=cat(2,vlb1,repmat(y(i,1),1,6));
            vub1=cat(2,vub1,repmat(y(i,2),1,6));
            b1=cat(2,r,repmat(b1,1,6));
            r=b1;
            i=i+1;
 end

% Generate the upper and lower bounds, bits required to code each rule
vlb2=zeros(1,3*n);
vub2=7*ones(1,3*n);
b2=3*ones(1,3*n);
vlb=cat(2,vlb1,vlb2);
vub=cat(2,vub1,vub2);
bits=cat(2,b1,b2);

% Make a Fuzzy inference system with the input and output variables
a=newfis('fuzzy');
l=1;
Lis=char(List_inp);
Lis_o=char(List_out);
    for i=1:n
        a=addvar(a,'input',List_inp(i),[y(l,1) y(l,2)]);
        a.input(i).name=Lis(i,:);
        l=l+1;
    end;
a=addvar(a,'output','Diagnosis',[0 5]);

% Give the necessary genetic algorithm parameters
options = foptions([1 1e-3]);
options(13) = 0.01;                 % Mutation probability
options(14) =20;                    % Number of generations
options(11)=20;                     % Population size
options(12)=1;                      % Crossover probability

% Note the initial time before entering into the genetic loop
t=cputime;


disp('%%%%%%%%%%%%%%%%%%%%%%%%%%%%')
disp('    Genetic optimization    ')
disp('%%%%%%%%%%%%%%%%%%%%%%%%%%%%')

% Start the genetic algorithm by encoding  all the MF parameters of each
% input and all the 3 rules into the genome.
%  the  genetic  function  has  not  been  shown  in  this  thesis  .It  is
basically a
% genetic algorithm implementation

[x,stats,options,bf,fgen,igen]=genetic('fit',[],options,vlb,vub,bits,a,n,p,s,  y,total);


% Find the time taken by the genetic algorithm
disp('%%%%% The time taken by the genetic algorithm %%%%%');
e=cputime-t
```

```
% The optimized genome in decoded form
disp('%%%%% The optimized genome in decoded form %%%%%%%%')
x
pause;

% Decode the genome into the MF parameters(f) and the rules(t1)
q=length(x);
k=x(n*6+1:q);
t1=x(1:n*6);
h=1;j=1;
    while (j<n*6+1)
        f(h,:)=t1(j:j+5);
        j=j+6;
        h=h+1;
    end
f=sort(f,2);
l=1;
% Define the Name , Type and Parameters of each MF for each input, using
the optimized genome.
disp('%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%')
disp(' The optimized Membership function plots:   ')
disp('%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%')
pause;

% For trapezoidal MF use  'trapmf' and number of the parameters to four.
for i=1:n
    a=addmf(a,'input',i,'VLOW','trimf',[y(l,1) y(l,1) f(i,1)]);
    a=addmf(a,'input',i,'LOW','trimf',[y(l,1) f(i,2) f(i,3)]);
    a=addmf(a,'input',i,'MEDIUM','trimf',[f(i,1) f(i,4) f(i,5)]);
    a=addmf(a,'input',i,'HIGH','trimf',[f(i,3) f(i,6) y(l,2)]);
    a=addmf(a,'input',i,'VHIGH','trimf',[f(i,5) y(l,2) y(l,2)]);
l=l+1;
end

% Add the output MF to the FIS.
if(p==1)v1=2;v2=4;
else v1=1;v2=2;
end

a=addmf(a,'output',1,List_out(1),'trimf',[v1 v1 v1]);
a=addmf(a,'output',1,List_out(2),'trimf',[v2 v2 v2]);
plotmf(a,'output',1);
pause;
a.output(1).mf(1).name=Lis_o(1,:);
a.output(1).mf(2).name=Lis_o(2,:);

disp('%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%')
disp('     The optimized rules       ')
disp('%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%')
k
pause;

% Find the average number of variables per rule
avg=0;
for i=1:n*3
    if(k(1,i)>5) k(1,i)=0;
    end
    if (k(1,i)~=0)avg=avg+1;
    end
end
avg=avg/3

% Add the optimized rules into the FIS
switch p,
    case 1,
        rulelist=[k(1,1)  k(1,2)  k(1,3)  k(1,4)  k(1,5)  k(1,6)  k(1,7)
            k(1,8) k(1,9) 1 1 1; k(1,10) k(1,11) k(1,12) k(1,13) k(1,14)
            k(1,15)  k(1,16)  k(1,17)  k(1,18)  1  1  1  ;  k(1,19)  k(1,20)
            k(1,21) k(1,22) k(1,23) k(24) k(1,25) k(1,26) k(1,27) 1 1
            1];
    case 2,
        rulelist=[k(1,1)  k(1,2)  k(1,3)  k(1,4)  k(1,5)  k(1,6)  1  1  1;
            k(1,7) k(1,8) k(1,9) k(1,10) k(1,11) k(1,12) 1 1 1 ; k(1,13)
            k(1,14) k(1,15) k(1,16) k(1,17) k(1,18) 1 1 1];
    case 3,
        rulelist=[k(1,1)  k(1,2)  k(1,3)  k(1,4)  k(1,5)  k(1,6)  k(1,7)
            k(1,8) 1 1 1; k(1,9) k(1,10) k(1,11) k(1,12) k(1,13) k(1,14)
            k(1,15)  k(1,16)  1  1  1  ;k(1,17)  k(1,18)  k(1,19)  k(1,20)
            k(1,21) k(1,22) k(1,23) k(1,24) 1 1 1];
end;

a=addrule(a,rulelist);

% Testing the optimized FIS to obtain the percentage of  correctly
classified data for the 50% of the dataset.
if (p==1)      L1=343;L2=683;L3=340;
elseif(p==2)  L1=174;L2=345;L3=171;
else          L1=385;L2=768;L3=383;
end;

% Call the weighted average function.
out=wgav(L2,n,f,p,k,s,y,L1);

% Compare the weighted average output with the output set of the
dataset.
c2=comp(out,s,L1,L2,p);

disp('%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%')
disp('     The correctly classified data using the optimized FIS   ')
disp('%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%')
Percent_classification= (c2/L3)*100

% plot the optimized membership function plots
for d=1:n
plotmf(a,'input',d);
pause;
end

% show the optimized rules
    showrule(a,1)
    showrule(a,2)
    showrule(a,3)

% Give any input to the optimized FIS for diagnosis
disp('%%%%%%%%%%%%%%%%%%')
disp('      DIAGNOSIS  ' )
disp('%%%%%%%%%%%%%%%%%%')
disp('The input range is:');
disp(y)

[data]=input('Enter the input data whose diagnosis is to be found:')
wgt_outp=wgtav(1,n,f,p,k,data,y,1);

switch(p)
    case 1,
        if(wgt_outp>=3) sprintf('Malignant')
        else sprintf('Benign')
        end;
    case 2,
        if(wgt_outp>=1.5) sprintf('Liver disorder')
        else sprintf('No liver problem')
        end;
    case 3,
        if(wgt_outp>=1.5) sprintf('Diabetes +')
        else sprintf('Diabetes -')
        end;
end;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%
```

2. THE FITNESS FUNCTION

```
% x = The decoded genome containing the MF parameters and the rule
      values
% a = The fuzzy inference system
% n1 = The number of inputs
% p1 = Number showing which diagnosis to be done
% s1 = The input dataset matrix.
% y1 = Min max value matrix of each input
% tot = Total number of patient data in the dataset
% t = Matrix of the MF parameters.
% k = rules coded in decimal format for 3 rules
%       : 1 =LOW, 2=MEDIUM, 3=HIGH
% outp = The weighted average output for each patient data .
% comp = Function which compares the weighted average output and the
         output
%        set of the dataset.
% wgtav = The weighted average function to find the defuzzified output
% c = Correctly classified data which is to be maximized

function c = fitness(x,a,n1,p1,s1,y1,tot)

% Find the length of the decoded genome
e=length(x);

% Divide the genome into the MF parameter part(t) and the rule part(k)
t1=x(1:n1*6);
k=x(n1*6+1:e);
s=1;j=1;
while (j<n1*6+1)
    t(s,:)=t1(j:j+5);
    j=j+6;
    s=s+1;
end
t=sort(t,2);                                % t has the MF
parameters

% Add triangular membership functions to the FIS input and output
variables.
m=1;

for i=1:n1
    a=addmf(a,'input',i,'Vlow','trimf',[y1(m,1) y1(m,1) t(i,1)]);
    a=addmf(a,'input',i,'low','trimf',[y1(m,1) t(i,2) t(i,3)]);
    a=addmf(a,'input',i,'medium','trimf',[t(i,1) t(i,4) t(i,5)]);
    a=addmf(a,'input',i,'high','trimf',[t(i,3) t(i,6) y1(m,2)]);
    a=addmf(a,'input',i,'vhigh','trimf',[t(i,5) y1(m,2) y1(m,2)]);
    m=m+1;
end

if(p1==1)v1=2;v2=4;
else v1=1;v2=2;
end

    a=addmf(a,'output',1,'class1','trimf',[v1 v1 v1]);
    a=addmf(a,'output',1,'class2','trimf',[v2 v2 v2]);

% k has the rules coded in decimal format for 3 rules
%       : 1 =LOW, 2=MEDIUM, 3=HIGH
for i=1:n1*3
    if(k(1,i)>5) k(1,i)=0;
    end
end

switch p1,
    case 1,
        rulelist=[k(1,1)  k(1,2)  k(1,3)  k(1,4)  k(1,5)  k(1,6)  k(1,7)
            k(1,8) k(1,9) 1 1 1; k(1,10) k(1,11) k(1,12) k(1,13) k(1,14)
            k(1,15)  k(1,16)  k(1,17)  k(1,18)  1  1  1  ;  k(1,19)  k(1,20)
            k(1,21) k(1,22) k(1,23) k(24) k(1,25) k(1,26) k(1,27) 1 1
            1];
    case 2,
        rulelist=[k(1,1)  k(1,2)  k(1,3)  k(1,4)  k(1,5)  k(1,6)  1  1  1;
            k(1,7) k(1,8) k(1,9) k(1,10) k(1,11) k(1,12) 1 1 1 ; k(1,13)
            k(1,14) k(1,15) k(1,16) k(1,17) k(1,18) 1 1 1];
    case 3,
        rulelist=[k(1,1)  k(1,2)  k(1,3)  k(1,4)  k(1,5)  k(1,6)  k(1,7)
            k(1,8) 1 1 1; k(1,9) k(1,10) k(1,11) k(1,12) k(1,13) k(1,14)
            k(1,15)  k(1,16)  1  1  1;k(1,17)  k(1,18)  k(1,19)  k(1,20)
            k(1,21) k(1,22) k(1,23) k(1,24) 1 1 1];
end;

% Add the rules to the FIS
a=addrule(a,rulelist);

% Call the weighted average function
outp=wgav(tot,n1,t,p1,k,s1,y1,1);

% Compare the weighted average output with the output set of the
dataset.
c2=comp(outp,s1,1,tot,p1);

% c is the fitness function (Correctly classified data) which is to be
maximized
c=c2/tot;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%
```

3. WEIGHTED AVERAGE FUNCTION

```
% This function finds out the weighted average for all input of a
particular diagnosis dataset.
% tot = Total number of data for each dataset.
% r = Total inputs for each dataset
% t1 = The MF parameters
% p1 = The number specifying which diagnosis
% y = Min max matrix for each input
% k1 = Rule parameters
% input = Input dataset.
% wt = Matrix containing the defuzzified weighted average output

function  [wt] =wgav(tot,r,t1,p1,k1,input,y,z)

q=1;q1=r;
for l=1:3
  k(l,:)=k1(q:q1);
  q=q1+1;
  q1=q+r-1;
end;
k=transpose(k);

% For validation use 50% of the data for training.
for b=z:tot
        c=1;
        if(p1==1)j=2;
        else j=1;
        end

% Fuzzify the input using triangular MF
% For trapezoidal MF use 'trapmf' and number of the parameters to four.
        for q=1:r
        w(q,1)=evalmf(input(b,j),[y(c,1) y(c,1) t1(q,1)],'trimf');
        w(q,2)=evalmf(input(b,j),[y(c,1) t1(q,2) t1(q,3)],'trimf');
        w(q,3)=evalmf(input(b,j),[t1(q,1) t1(q,4) t1(q,5)],'trimf');
        w(q,4)=evalmf(input(b,j),[t1(q,3) t1(q,6) y(c,2)],'trimf');
        w(q,5)=evalmf(input(b,j),[t1(q,5) y(c,2) y(c,2)],'trimf');
        c=c+1;
        j=j+1;
        end


% Defuzzify using weighted average method as output is singleton
for j=1:3
rule=[];
g1=[];
[f,g,v] = find(k(j,:));
e=length(v);
for i=1:e
g1(1,i)=w(g(i),v(1,i));
end
rule=min(g1);
end

den=sum(rule)+0.25;
 if(p1==1)m=2;m1=4;
else m=1;m1=2;
end
wt(b)=(sum(rule)*m+0.25*m1)/den;
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%
```

4. COMPARE FUNCTION

```
% This function compares the weighted average output with the output set
of
% the dataset.
% c1 = Total number of patient data correctly diagnosed/classified
% x1 = The weighted average output for each patient data
% y = The dataset
% b = The patient number to start with
% t = The patient number to end with
% p = Number showing which diagnosis to be done
% k = The output column number in the dataset
% k1 = Class 2 value shown in the dataset
% k2 = Class 1 value shown in the dataset

function c1=comp(x1,y,b,t,p)
l=1;
m=1;

% Set the threshold value: j
if(p==2)j=1.5;k=7;k1=2;k2=1;
    elseif(p==1) j=3;k=11;k1=4;k2=2;
        else j=1.5;k=9;k1=0;k2=1;
end

for i=b:t
    % Compare the output with the threshold
    if(x1(i)>=j)
        % Compare the output with the output in the dataset
        if(y(i,k)==k1) l=l+1;
        end
    elseif(y(i,k)==k2) m=m+1;
    end
end
c1=l+m;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%
```

5. INPUT/OUTPUT NAMES FUNCTION

```
% This function sets the names of all the inputs of each dataset.
% n1 = The chosen diagnosis number.
% listn = The array containing the names of the inputs
```

```
function listn = name(n1)

if (n1==1)
listn={'Clump   thickness';   'Cell   size   Uniformity'   ;'Cell   shape
Uniformity'; 'Marginal adhesion'; 'Epithelial cell size'; 'Bare nuclei';
'Bland chromatin'; 'Normal nucleoli'; 'Mitosis'};

elseif (n1==2)
listn={'Mean   Corpuscular   Volume';'Alkaline   phosphotase';'Alamine
aminotransferase';  'Aspartate  aminotransferase';  'G-G  transpeptidase';
'Drinks'};

else listn={'Number of times pregnant'; 'Plasma glucose concentration';
'Diastolic  BP';  'Skin  fold  thickness';  '2Hr  serum  insulin  ';  'Body
MassIndex'; 'Diabetes pedigree fun'; 'Age'};
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% This function sets the names of all the outputs of each dataset.
% n1 = The chosen diagnosis number.
% listn = The array containing the names of the outputs

function listn=nameo(n1)
if (n1==1)
    listn={'BENIGN';'MALIGNANT'};
elseif (n1==2)
    listn={'LIVER DISORDER';'NO LIVER DISORDER'};
else listn={'DIABETES POSITIVE';'DIABETES NEGATIVE'};
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```