LAB MANUAL

Semester-VII

Department of Computer Engineering

Institute Vision, Mission & Quality Policy

Vision

To foster and permeate higher and quality education with value added engineering, technology programs, providing all facilities in terms of technology and platforms for all round development with societal awareness and nurture the youth with international competencies and exemplary level of employability even under highly competitive environment so that they are innovative adaptable and capable of handling problems faced by our country and world at large.

Mission

The Institution is committed to mobilize the resources and equip itself with men and materials of excellence thereby ensuring that the Institution becomes pivotal center of service to Industry, academia, and society with the latest technology. RAIT engages different platforms such as technology enhancing Student Technical Societies, Cultural platforms, Sports excellence centers, Entrepreneurial Development Center and Societal Interaction Cell. To develop the college to become an autonomous Institution & deemed university at the earliest with facilities for advanced research and development programs on par with international standards. To invite international and reputed national Institutions and Universities to collaborate with our institution on the issues of common interest of teaching and learning sophistication.

Quality Policy

ज्ञानधीनं जगत् सर्वम।

Knowledge is supreme.

Our Quality Policy

It is our earnest endeavour to produce high quality engineering professionals who are innovative and inspiring, thought and action leaders, competent to solve problems faced by society, nation and world at large by striving towards very high standards in learning, teaching and training methodologies.

Our Motto: If it is not of quality, it is NOT RAIT!

Dr. Vijay D. Patil
President, RAES

Department Vision & Mission

Vision

To impart higher and quality education in computer science with value added engineering and technology programs to prepare technically sound, ethically strong engineers with social awareness. To extend the facilities, to meet the fast changing requirements and nuture the youths with international competencies and exemplary level of employability and research under highly competitive environments.

Mission

- To mobilize the resources and equip the institution with men and materials of excellence to provide knowledge and develop technologies in the thrust areas of computer science and Engineering.
- To provide the diverse platforms of sports, technical, cocurricular and extracurricular activities for the overall development of student with ethical attitude.
- To prepare the students to sustain the impact of computer education for social needs encompassing industry, educational institutions and public service.
- To collaborate with IITs, reputed universities and industries for the technical and overall uplifment of students for continuing learning and entrepreneurship.

Index

List of Experiments

Sr. No.	Experiments Name
1	Implementation of Fuzzy Operations.
2	Implementation of Fuzzy Relations (Max-min Composition)
3	Implementation of Fuzzy Controller (Washing Machine)
4	Implementation of Simple Neural Network (McCulloh-Pitts model)
5	Implementation of Perceptron Learning Algorithm
6	Implementation of Unsupervised Learning Algorithm
7	Implementation of Simple Genetic Application
8	Study of ANFIS Architecture
9	Study of Derivative-free Optimization
10	Study of research paper on Soft Computing.

Course Outcome & Experiment Plan

Course Outcomes:

CO1	Understand components of Soft Computing and differentiate between hard and soft computing.
CO2	Understand the difference between learning and programming and explore practical applications of Neural Networks (NN).
CO3	To analyse and appreciate the applications which can use fuzzy logic.
CO4	Understand the efficiency of a hybrid system and how Neural Network and fuzzy logic can be hybridized to form a Neuro-fuzzy network and its various applications and they will be able to design inference systems.
CO5	Appreciate the importance of optimizations and its use in computer engineering fields and other domains.
CO6	Understand the basics of genetic algorithm, use of GA operators and its applications.

Experiment Plan:

Module No.	Week No.	Experiments Name	Course Outcome
1	W1	Implementation of Fuzzy Operations.	CO3
2	W2	Implementation of Fuzzy Relations (Max-min Composition)	CO3
3	W3	Implementation of Fuzzy Controller (Washing Machine)	CO3
4	W4	Implementation of Simple Neural Network (McCulloh-Pitts model)	CO2
5	W5,W6	Implementation of Supervised Learning Algorithm	CO2
6	W7,W8	Implementation of Unsupervised Learning Algorithm	CO2

7	W9	Implementation of Simple Genetic Application	CO6
8	W10	Study of ANFIS Architecture	CO4
9	W11	Study of Derivative based and Derivative-free Optimization	CO5
10	W12	Study of research paper on Soft Computing.	CO1

Study and Evaluation Scheme

Course Code	Course Name	Teaching Scheme			Credits Assigned			
		Theory	Practical	Tutorial	Theory	Practical	Tutorial	Total
CPE 7025	Soft Computing	04	02	--	04	01	--	05

Course Code	Course Name	Examination Scheme		
		Term Work	Oral	Total
CPE7025	Soft Computing	25	25	50

Term Work:

1. Term work assessment must be based on the overall performance of the student with every experiment graded from time to time. The grades should be converted into marks as per the Credit and Grading System manual and should be added and averaged.
2. The final certification and acceptance of term work ensures satisfactory performance of laboratory work and minimum passing marks in term work.

Practical & Oral:

1. Oral exam will be based on the entire syllabus of Soft Computing..

Soft Computing

Experiment No. : 1

Implementation of Fuzzy Operations

Experiment No. 1

1. **Aim:** Implementation of Fuzzy Operations.

2. **Objectives:**

   - Provide an understanding of the basic mathematical elements of fuzzy sets.
   - Understand and analyse concepts of fuzzy set.
   - To use fuzzy set operations to implement current computing techniques used in fuzzy computing.

3. **Outcomes:** The students will be able to

   - Learn mathematical basis as well as the general principles of various soft computing techniques.
   - To analyze the applications using fuzzy set which uses current techniques, skills, and tools necessary for computing.
   - To identity and solves the engineering problems using the fuzzy set theory and identify the differences and similarities between fuzzy sets and classical sets theories.

4. **Hardware/Software Required :**JAVA/ C/C++/MATLAB

5. **Theory:**

Fuzzy Logic:

Fuzzy logic is an organized method for dealing with imprecise data. It is a multivalued logic that allows intermediate values to be defined between conventional solutions.
In classical set theory, the membership of elements in a set is assessed in binary terms according to a bivalent condition — an element either belongs or does not belong to the set. By contrast, fuzzy set theory permits the gradual assessment of the membership of elements in a set; this is described with the aid of a membership function valued in the real unit interval [0, 1].
Bivalent Set Theory can be somewhat limiting if we wish to describe a 'humanistic' problem mathematically. For example, Fig 1 below illustrates bivalent sets to characterize the temperature

of a room.The most obvious limiting feature of bivalent sets that can be seen clearly from the diagram is that they are mutually exclusive - it is not possible to have membership of more than one set. Clearly, it is not accurate to define a transition from a quantity such as 'warm' to 'hot' by the application of one degree Fahrenheit of heat. In the real world a smooth (unnoticeable) drift from warm to hot would occur.

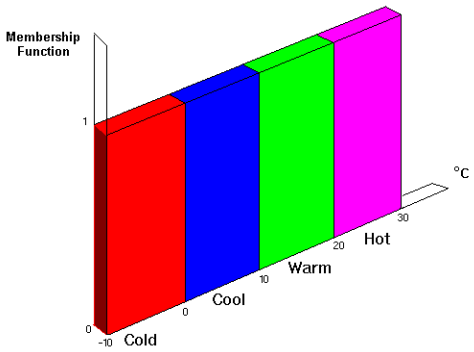


Fig. 1 : Bivalent Sets to Characterize the Temp. of a room.

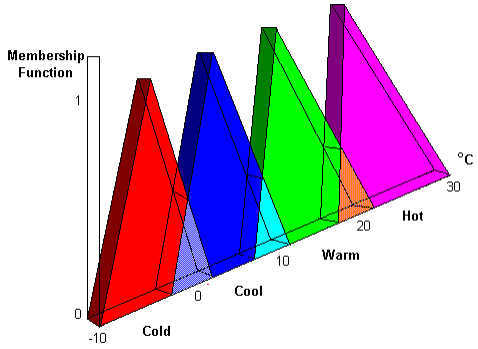


Fig. 2 - Fuzzy Sets to characterize the Temp. of a room.

Fuzzy Sets:

A fuzzy set is a pair (U, m) where U is a set and $m: U \to [0, 1]$.

For each $x \in U$, the value $m(x)$ is called the degree of membership of x in (U, m). For a finite set $U = \{x_1, \ldots, x_n\}$, the fuzzy set (U, m) is often denoted by $\{m(x_1)/x_1, \ldots, m(x_n)/x_n\}$.

Let $x \in U$. Then,

- x is called not included in the fuzzy set (U, m) if $m(x) = 0$
- x is called fully included if $m(x) = 1$
- x is called a fuzzy member if $0 < m(x) < 1$.

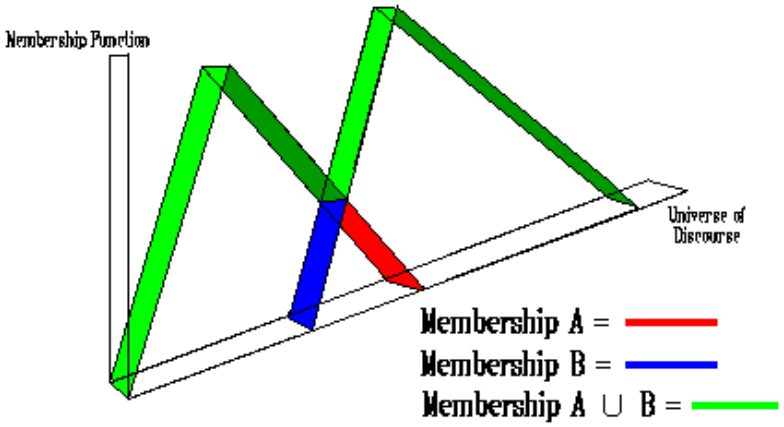
Example: $\tilde{A} = \{(x_1, 0.1), (x_2, 0.7), (x_3, 1), (x_4, 0)\}$

$\tilde{B} = \{(x_1, 0.4), (x_2, 0.3), (x_3, 1), (x_4, 0.2)\}$

1. **Union:**

Union of two fuzzy sets $\tilde{A}$ and $\tilde{B}$ is denoted as $\tilde{A} \cup \tilde{B}$ and is defined as,

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

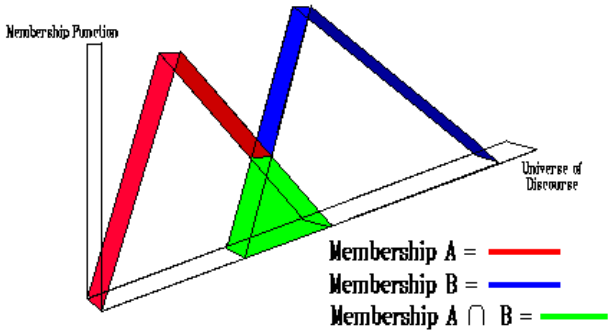


Example: $\tilde{A} \cup \tilde{B} = \{(x_1, 0.4), (x_2, 0.7), (x_3, 1), (x_4, 0.2)\}$

2. Intersection

Union of two fuzzy sets $\tilde{A}$ and B is denoted by $\tilde{A} \cap \tilde{B}$ and is defined as,

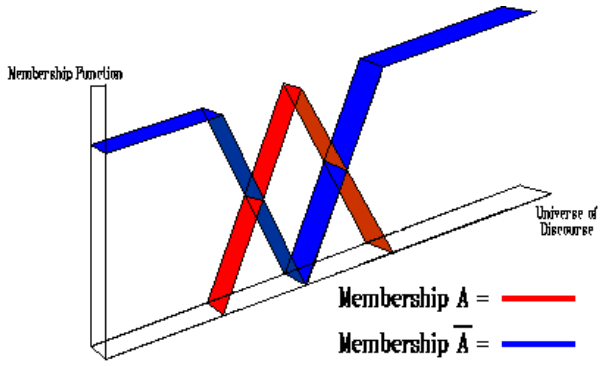
$$\mu_{A \cap B}(x) = \min\left(\mu_A(x), \mu_B(x)\right)$$



Example: $\tilde{A} \cap \tilde{B} = \{(x_1, 0.1), (x_2, 0.3), (x_3, 1), (x_4, 0)\}$

3. Complement

Complement of a fuzzy set $\tilde{A}$ denoted by $\widetilde{A^c}$ and is defined as,



Example: $\widetilde{A^c} = \{(x_1, 0.9), (x_2, 0.3), (x_3, 0), (x_4, 1)\}$

4. Algebraic Sum

Algebraic sum of two fuzzy sets $\tilde{A}$ and B is denoted by $\tilde{A} \dot{+} \tilde{B}$ and is defined as,

$\mu_{A \dot{+} B}(x) = \mu_A(x) + \mu_B(x) - \mu_A(x).\mu_B(x)$

Example: $\tilde{A} \dot{+} \tilde{B} = \{(x_1, 0.46), (x_2, 0.79), (x_3, 1), (x_4, 0.2)\}$

5. Algebraic Product

Algebraic product of two fuzzy sets $\widetilde{A}$ and B denoted by $\widetilde{A}.\widetilde{B}$ and is defined as,

$\mu_{A.B}(x) = \mu_A(x).\mu_B(x)$

Example: $\widetilde{A}.\widetilde{B} = \{(x_1, 0.04), (x_2, 0.21), (x_3, 1), (x_4, 0)\}$

6. Conclusion

The concepts of union, intersection and complement are implemented using fuzzy sets which helped to understand the differences and similarities between fuzzy set and classical set theories. It provides the basic mathematical foundations to use the fuzzy set operations.

7. Viva Questions:

- What are the properties of Fuzzy set?
- What is De Morgan's Law in Crisp Set?

- What is the difference between the crisp set and fuzzy set?

8. References:

1. S.N.Sivanandam, S.N.Deepa "Principles of Soft Computing" Second Edition, WileyPublication.
2. S.Rajasekaran and G.A.Vijayalakshmi Pai "Neural Networks, Fuzzy Logic and GeneticAlgorithms" PHI Learning.
3. Hagan, Demuth, Beale,"Neural Network Design" CENGAGE Learning, India Edition.
4. Satish Kumar, “Neural Networks –A classroom approach”, Second Edition,TMH

Soft Computing

Experiment No. : 2

Implementation of Fuzzy Relations

Experiment No. 2

1. **Aim:** Implementation of fuzzy relations (Max-Min Composition)

2. **Objectives:**

   - Provide an understanding of the basic mathematical elements of the theory of fuzzy sets.
   - To introduce the ideas of fuzzy sets, fuzzy logic.
   - To implement applications using the fuzzy set operations.

3 **Outcomes:** The will be able to
   - Understand the fuzzy arithmetic concepts.
   - Become familiar with fuzzy relations and the properties of these relations.
   - Identify, formulate and solve engineering problem.

4 **Software Required:** JAVA/ C/C++/MATLAB

5 **Theory:**

Max-Min Composition of fuzzy Relations

Fuzzy relation in different product space can be combined with each other by the operation called "Composition". There are many composition methods in use , e.g. max-product method, max-average method and max-min method. But max-min composition method is best known in fuzzy logic applications.

Definition:
Composition of Fuzzy Relations

- Consider two fuzzy relation; R (X ×Y) and S (Y ×Z), then a relation T (X × Z), can be expressed asmax-min composition

$$\mathbf{T = R \circ S}$$

$$\mu T(x, z) = \max\text{-}\min [\mu R(x, y), \mu S(y, z)]$$

= V [μR (x, y) ^ μS (y, z)]

- If algebraic product is adopted, then max-product compositionis adopted:

T = R o S

μT (x, z) = max [μR (x, y) . μS (y, z)]

= V [μR (x, y) . μS (y, z)]

- The max-min composition can be interpreted as indicating the strength of the existence of relation between the elements of X and Z.
- Calculations of (R o S) are almost similar to matrix multiplication.

Crisp relation:

Crisp relation is defined on the Cartesian product of two sets. Consider,

$$X \times Y = \{(x,y) | x \in X, y \in Y\}$$

The relation on this Cartesian product will be,

$$\mu_R = \begin{cases} 1, (x,y) \in R \\ 0, (x,y) \notin R \end{cases}$$

Example: Let X={1,4,5} and Y={3,6,7} then for relation R=x<y,

$$R = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix}$$

Fuzzy relation:

Let $X, Y \subseteq R$ be universal sets then,

$$R = \{((x,y), \mu_R(x,y)) \mid (x,y) \in X \times Y\}$$

Is called a fuzzy relation in $X \times Y \subseteq R$

Example: Let X={1,2,3} and Y={1,2}

If $\mu_R(x,y) = e^{-(x-y)^2}$, then

$$R = \left\{ \frac{e^{-(1-1)^2}}{(1,1)}, \frac{e^{-(1-2)^2}}{(1,2)}, \frac{e^{-(2-1)^2}}{(2,1)}, \frac{e^{-(2-2)^2}}{(2,2)}, \frac{e^{-(3-1)^2}}{(3,1)}, \frac{e^{-(3-2)^2}}{(3,2)} \right\}$$

$$R = \begin{bmatrix} 1 & 0.37 \\ 0.37 & 1 \\ 0.02 & 0.37 \end{bmatrix}$$

Max-Min Composition:

Let X,Y and Z be universal sets and let R and Q be relations that relate them as,

$R = \{ (x,y) | x \in X, y \in Y, R \subset X \times Y \}$

$Q = \{ (y,z) | y \in Y, z \in Z, Q \subset Y \times Z \}$

Then S will be a relation that relates elements of X with elements of Z as,

$S = R \circ Q$

$S = \{ (x,z) | x \in X, z \in Z, S \subset X \times Z$

Max min composition is then defined as,

$$\mu_S(x,z) = \max\left(\min\left(\mu_R(x,y), \mu_Q(y,z)\right)\right)$$

Example: $R = \begin{bmatrix} 0.6 & 0.5 & 0.4 \\ 0.2 & 0.1 & 0.2 \end{bmatrix}$ and $Q = \begin{bmatrix} 0.2 & 0.6 \\ 0.1 & 0.3 \\ 0.7 & 0.5 \end{bmatrix}$

$$S = \begin{bmatrix} 0.4 & 0.6 \\ 0.2 & 0.2 \end{bmatrix}$$

6. Conclusion

With the use of fuzzy logic principles max min composition of fuzzy set is calculated which describes the relationship between two or more fuzzy sets.

7. Viva Questions:

- What is the main difference between the probability and fuzzy logic?
- What are the types of fuzzy logic sets?
- Who is the founder of fuzzy logic?

8. References:

1. S.N.Sivanandam, S.N.Deepa "Principles of Soft Computing" Second Edition, WileyPublication.
2. S.Rajasekaran and G.A.Vijayalakshmi Pai "Neural Networks, Fuzzy Logic and GeneticAlgorithms" PHI Learning.
3. Hagan, Demuth, Beale,"Neural Network Design" CENGAGE Learning, India Edition.
4. Satish Kumar, “Neural Networks –A classroom approach”, Second Edition,TMH

Soft Computing

Experiment No. : 3

Implementation of Fuzzy Controller

Experiment No. 3

1. **Aim:** Implementation of fuzzy controller(Washing Machine)

2. **Objectives :**

   - Cover fuzzy logic inference with emphasis on their use in the design of intelligent or humanistic systems.
   - Prepare the students for developing intelligent systems.

3. **Outcomes:** The student will be able to

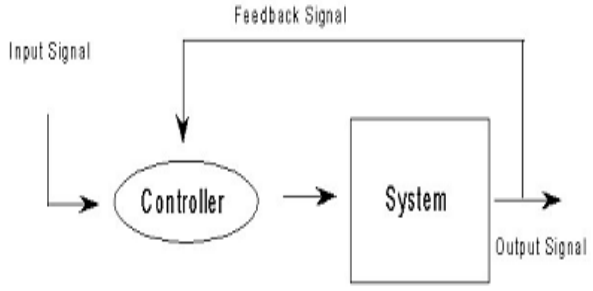
   - Become knowledgeable of conditional fuzzy propositions and fuzzy inference systems.
   - Become aware of the application of fuzzy inference in the area of control.
   - Identify, analyze, design and solve the problem, implement and validate the solution including both hardware and software.

4. **Software Required:** JAVA/ C/C++/MATLAB

5. **Theory:**

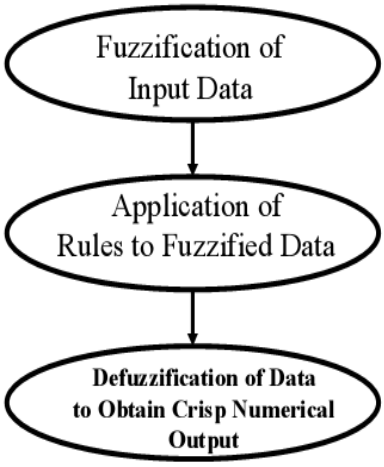
   Control System:

   Any system whose outputs are controlled by some inputs to the system is called control system.



Fuzzy Controller:

Fuzzy controllers are the most important applications of fuzzy theory. They work different than conventional controllers as:
Expert knowledge is used instead of differential equations to describe a system.
This expert knowledge can be expressed in very natural way using linguistic variables, which are described by fuzzy sets.
The fuzzy controllers are constructed in following three stages:
1. Create the membership values (fuzzify).
2. Specify the rule table.
3. Determine your procedure for defuzzifying the result.

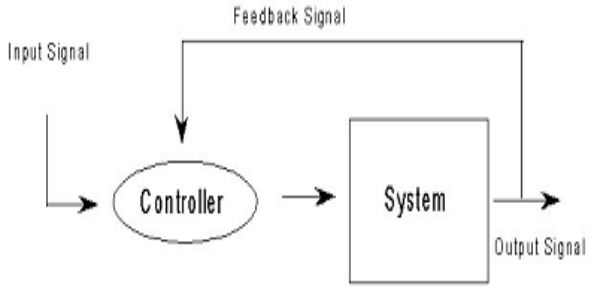


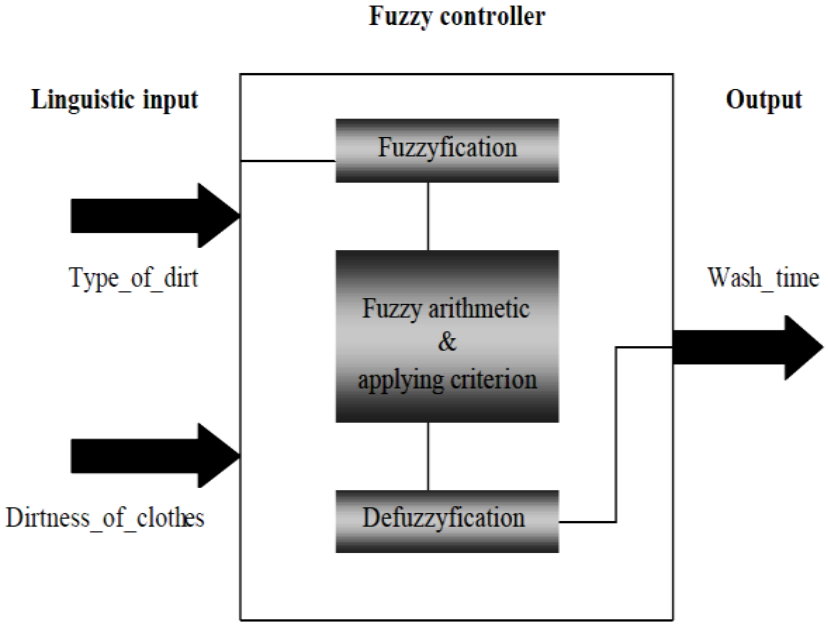
Washing Machine Controller:

To design a system using fuzzy logic, input & output is necessary part of the system. Main function of the washing machine is to clean cloth without damaging the cloth. In order to achieve it, the output parameters of fuzzy logic, which are the washing parameters, must be given more importance. The identified input & output parameters are:

Input: 1. Degree of dirt
2. Type of dirt

Output: Wash time

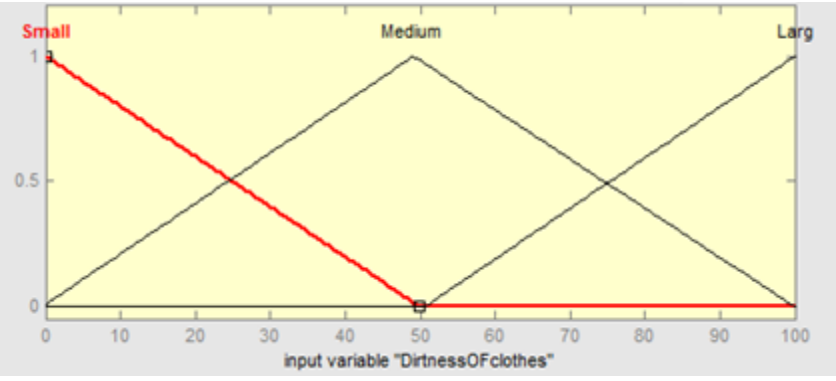




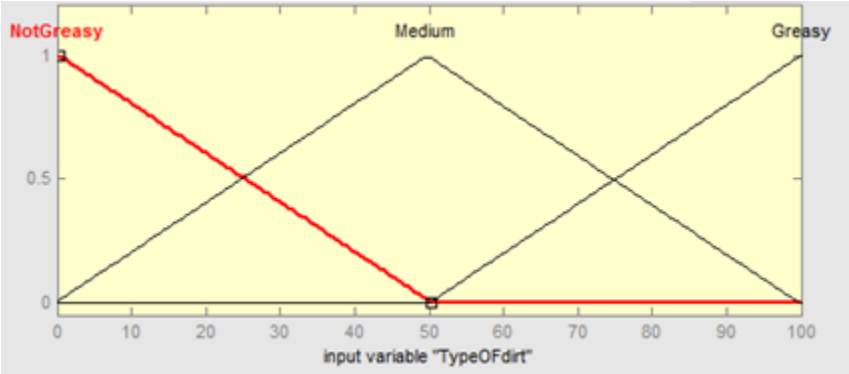
Fuzzy sets:

The fuzzy sets which characterize the inputs & output are given as follows:

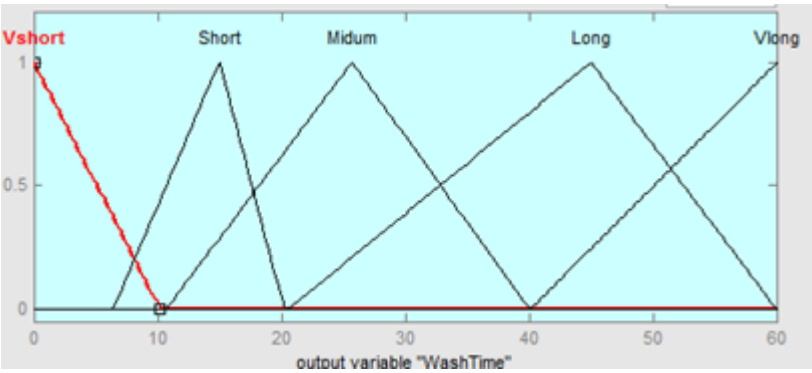
1. Dirtness of clothes



Type of dirt



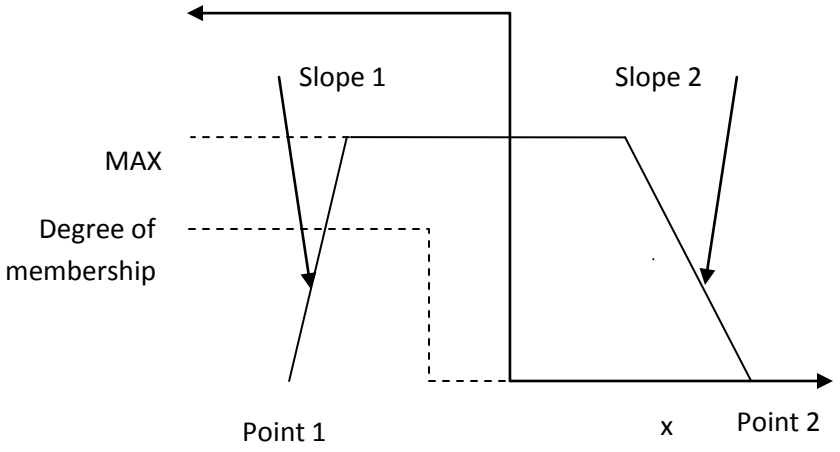
3. Wash time



Procedure:

Step1: Fuzzification of inputs

For the fuzzification of inputs, that is, to compute the membership for the antecedents, the formula used is,



$$Delta1 = x - point1$$
$$Delta2 = point2 - x$$
$$If(Delta1 \leq 0)\ or\ (Delta2 \leq 0)$$
$$then\ Degree\ of\ membership = 0$$
$$else$$
$$Degree\ of\ membership = min\begin{pmatrix} Delta1 * Slope1 \\ Delta2 * Slope2 \\ Max \end{pmatrix}$$

Step 2: Defining set of rules

	S	M	L
NG	VS	S	M
M	M	M	L
G	L	L	VL

1. If Dirtness of clothes is Large and Type of dirt is Greasy then Wash Time is Very Long;

2. If Dirtness of clothes is Medium and Type of dirt is Greasy then Wash Time is Long;

3. If Dirtness of clothes is Small and Type of dirt is Greasy then Wash Time is Long;

4. If Dirtness of clothes is Large and Type of dirt is Medium then Wash Time is Long;

5. If Dirtness of clothes is Medium and Type of dirt is Medium then Wash Time is Medium

6. If Dirtness of clothes is Small and Type of dirt is Medium then Wash Time is Medium;

7. If Dirtness of clothes is Large and Type of dirt is Not Greasy then Wash Time is Medium;

8. If Dirtness of clothes is Medium and Type of dirt is Not Greasy then Wash Time is Short;

9. If Dirtness of clothes is Small and Type of dirt is Not Greasy then Wash Time is Very Short;

Fuzzy Output

Fuzzy output of the system is the 'fuzzy OR' if all the fuzzy outputs of the rules with non zero rule strengths.

Step 3: Defuzzification

The Centre of Gravity method is applied go defuzzify the output,

$$x = \frac{\sum_{i=1}^{n} x_i . \mu(x_i)}{\sum_{i=1}^{n} \mu(x_i)}$$

6. Conclusion

Fuzzy controller for washing machine application is implemented using the fuzzy logic which also defines the rules for fuzzification and de-fuzzification. We have analyzed how outputs are controlled by some inputs of the system through this experiment.

7. Viva Questions:

- What is the reason that logic function has rapidly become one of the most successful technologies for developing sophisticated control systems?
- What is sequence of steps taken in designing a fuzzy logic machine?

8. References:

1. S.N.Sivanandam, S.N.Deepa "Principles of Soft Computing" Second Edition, Wiley Publication.
2. S.Rajasekaran and G.A.Vijayalakshmi Pai "Neural Networks, Fuzzy Logic and GeneticAlgorithms" PHI Learning.
3. Hagan, Demuth, Beale,"Neural Network Design" CENGAGE Learning, India Edition.
4. Satish Kumar, “Neural Networks –A classroom approach”, Second Edition,TMH

Soft Computing

Experiment No. : 4

To implement Mc-Culloch pits Model using XOR

Experiment No. 4

1. **Aim:** To implement Mc-Culloch pits Model using XOR

2. **Objectives:**

   - The student will be able to obtain the fundamentals and different architecture of neural networks.
   - The student will have a broad knowledge in developing the different algorithms for neural networks.

3. **Outcomes:** The students will be able to,

   - Describe the relation between real brains and simple artificial neural network models.
   - Understand the role of neural networks in engineering.
   - Apply the knowledge of computing and engineering concept to this discipline.

4. **Software Required:** C/C++/JAVA/ MATLAB

5. **Theory:**

   Neural network was inspired by the design and functioning of human brain and components.

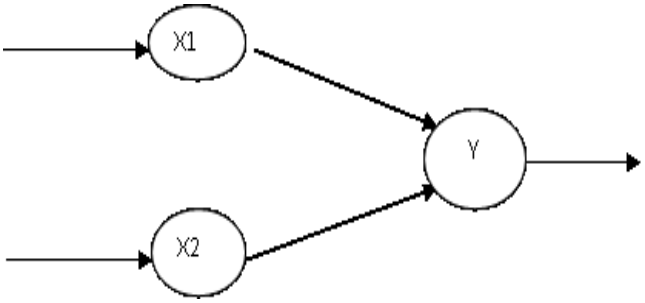
   Definition:

   "Information processing model that is inspired by the way biological nervous system (i.e the brain) process information, is called Neural Network."

   Neural Network has the ability to learn by examples. It is not designed to perform fix /specific task, rather task which need thinking (e.g. Predictions).

   ANN is composed of large number of highly interconnected processing elements(neurons) working in unison to solve problems. It mimic human brain. It is configured for special application such as pattern recognition and data classification through a learning process. ANN is 85-90% accurate.

Basic Operation of a Neural Network:



X1 and X2 – input neurons.

Y- output neuron

Weighted interconnection links- W1 and W2.

Net input calculation is :

Yin= x1w1+x2w2

Output is :

y=f(Yin)

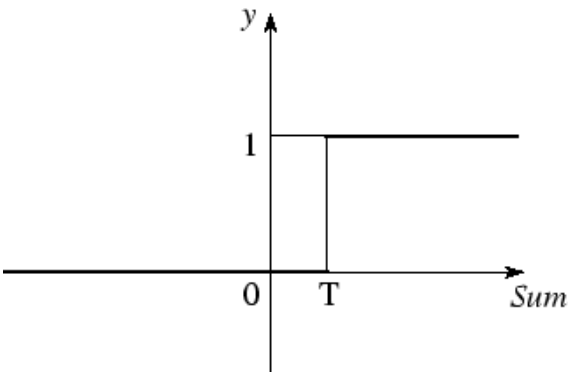
Output= function

The McCulloch-Pitts Model of Neuron:

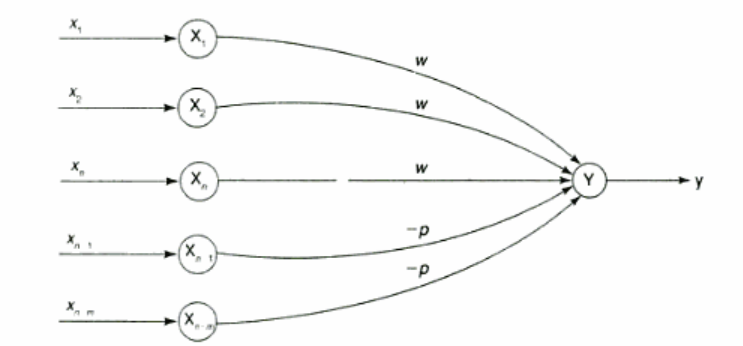
The early model of an artificial neuron is introduced by Warren McCulloch and Walter Pitts in 1943. The McCulloch-Pitts neural model is also known as linear threshold gate. It is a neuron of a set of inputs I1,I2,I3…Im and one output y . The linear threshold gate simply classifies the set of inputs into two different classes. Thus the output y is binary. Such a function can be described mathematically using these equations:

$$Sum = \sum_{i=1}^{N} I_i W_i,$$

$$y = f(Sum).$$



W1,W2…Wm are weight values normalized in the range of either (0,1) or (-1,1) and associated with each input line, Sum is the weighted sum, and T is a threshold constant. The function f is a linear step function at threshold T as shown in figure



A simple M-P neuron is shown in the figure.

It is excitatory with weight (w>0) / inhibitory with weight –p (p<0).

In the Fig., inputs from x1 to xn possess excitatory weighted connection and Xn+1 to xn+m has inhibitory weighted interconnections.

Since the firing of neuron is based on threshold, activation function is defined as

$$f(y_{in}) = \begin{cases} 1 & if\ y_{in} \geq \theta \\ 0 & if\ y_{in} < \theta \end{cases}$$

For inhibition to be absolute, the threshold with the activation function should satisfy the following condition:

$$\theta > nw - p$$

Output will fire if it receives "k" or more excitatory inputs but no inhibitory inputs where

$$kw \geq \theta > (k-1)\ w$$

- The M-P neuron has no particular training algorithm.

- An analysis is performed to determine the weights and the threshold.

- It is used as a building block where any function or phenomenon is modelled based on a logic function.

Problem Statement: Implement XOR function using MP model

Truth table for XOR function is:

X1	X2	Y
0	0	0
0	1	1
1	0	1
1	1	0

Activation function Yin is as follows:

$Yin=x_1w_1+x_2w_2$

As we know,

X_1 XOR X_2 == (X_1 AND NOT X_2) OR (X_2 AND NOT X_1)

Let Z_1=(X_1 AND NOT X_2) and Z_2=(X_2 AND NOT X_1)

X1	X2	Z1
0	0	0
0	1	0
1	0	1
1	1	0

For Z1,

W_{11}=1 and W_{12}=-1

Θ=1

X1	X2	Z2
0	0	0
0	1	1
1	0	0
1	1	0

For Z2,

W_{11}=-1 and W_{12}=1

Θ=1

Y=Z1+Z2

Z1	Z2	Y
0	0	0
0	1	1
1	0	1
1	1	0

For Y,

W_{11}=1 and W_{12}=1

Θ=1

6. Conclusion:

Mc-Culloch pits Model is implemented for XOR function by using the thresholding activation function. Activation of M-P neurons is binary (i.e) at any time step the neuron may fire or may not fire. Threshold plays major role here.

7. Viva Questions:

- What are Neural Networks? What are the types of neural networks?

- How are Artificial Neural Networks different from Normal Computers?
- What is simple Artificial Neuron?
- What is meant by training of artificial neural networks?

8. References:

1. S.N.Sivanandam, S.N.Deepa "Principles of Soft Computing" Second Edition, Wiley Publication.
2. S.Rajasekaran and G.A.Vijayalakshmi Pai "Neural Networks, Fuzzy Logic and GeneticAlgorithms" PHI Learning.
3. Hagan, Demuth, Beale,"Neural Network Design" CENGAGE Learning, India Edition.
4. Satish Kumar, “Neural Networks –A classroom approach”, Second Edition,TMH

Soft Computing

Experiment No. : 5

Implementation of Single Layer Perceptron Learning Algorithm

Experiment No. 5

1. **Aim:** Implementation of Single layer Perceptron Learning Algorithm.

2. **Objectives:**
   - To become familiar with neural networks learning algorithms from available examples.
   - Provide knowledge of learning algorithm in neural networks.

3. **Outcomes:** The student will be able to,
   - Have an understanding of the concepts and techniques of neural networks through the study of the most important neural network models**.**
   - Discuss the main factors involved in achieving good learning and generalization performance in neural network systems.
   - Use the current techniques and tools required for computing practice.

4. **Software Required:** JAVA / MATLAB

5. **Theory:**

Neural networks are a branch of “Artificial Intelligence". Artificial Neural Network is a system loosely modelled based on the human brain. Neural networks are a powerful technique to solve many real world problems. They have the ability to learn from experience in order to improve their performance and to adapt themselves to changes in the environment. In addition to that they are able to deal with incomplete information or noisy data and can be very effective especially in situations where it is not possible to define the rules or steps that lead to the solution of a problem. In a nutshell a Neural network can be considered as a black box that is able to predict an output pattern when it recognizes a given input pattern. Once trained, the neural network is able to recognize similarities when presented with a new input pattern, resulting in a predicted output pattern.

In late 1950s, Frank Rosenblatt introduced a network composed of the units that were

enhanced version of McCulloch-Pitts Threshold Logic Unit (TLU) model. Rosenblatt's model of neuron, a perceptron, was the result of merger between two concepts from the 1940s, McCulloch-Pitts model of an artificial neuron and Hebbian learning rule of adjusting weights. In addition to the variable weight values, the perceptron model added an extra input that represents bias. Thus, the modified equation is now as follows:

$$Sum = \sum_{i=1}^{N} I_i W_i + b,$$

where b represents the bias value.

6. Algorithm:

Perceptron Learning Algorithm:

The perceptron learning rule was originally developed by Frank Rosenblatt in the late 1950s. Training patterns are presented to the network's inputs; the output is computed. Then the connection weights$w_j$are modified by an amount that is proportional to the product of

- the difference between the actual output, y, and the desired output, d, and
- the input pattern, x.

The algorithm is as follows:

1. Initialize the weights and threshold to small random numbers.
2. Present a vector x to the neuron inputs and calculate the output.
3. Update the weights according to:

$$w_j(t+1) = w_j(t)+\eta(d-y)x$$

where

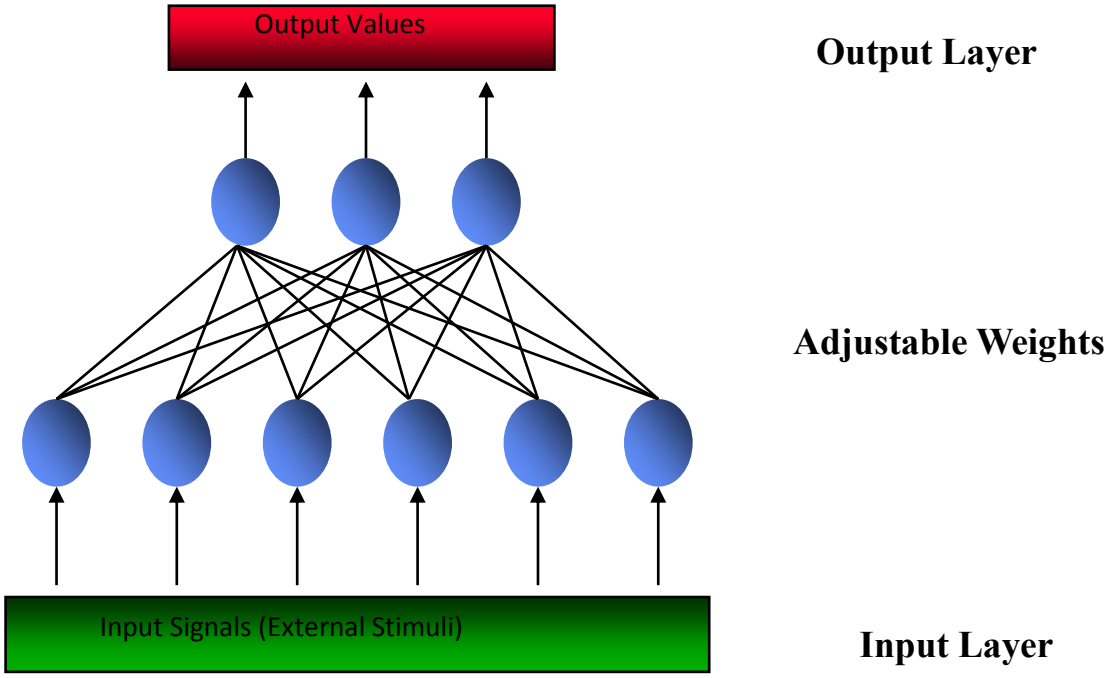
- d is the desired output,
- t is the iteration number, and
- eta is the gain or step size, where 0.0 < n < 1.0

4. Repeat steps 2 and 3 until:

1. the iteration error is less than a user-specified error threshold or
2. a predetermined number of iterations have been completed.

Learning only occurs when an error is made; otherwise the weights are left unchanged.

Multilayer Perceptron



Problem Statement: Implement AND function using perceptron model

Truth table for AND function is:

X1	X2	Y
0	0	0
0	1	0
1	0	0
1	1	1

7. Conclusion:

Single layer perceptron learning algorithm is implemented for AND function. It is used for train the iterations of neural network. Neural network mimics the human brain and perceptron learning algorithm trains the neural network according to the input given.

8. Viva Questions:

- What is feed forward network?
- Write the logistic sigmoid function?
- Why use Artificial Neural Networks? What are its advantages?
- List some commercial practical applications of Artificial Neural Networks.
- What are the disadvantages of Artificial Neural Networks?

9. References:

1. S.N.Sivanandam, S.N.Deepa "Principles of Soft Computing" Second Edition, Wiley Publication.
2. S.Rajasekaran and G.A.Vijayalakshmi Pai "Neural Networks, Fuzzy Logic and GeneticAlgorithms" PHI Learning.
3. Hagan, Demuth, Beale,"Neural Network Design" CENGAGE Learning, India Edition.
4. Satish Kumar, “Neural Networks –A classroom approach”, Second Edition,TMH

Soft Computing

Experiment No. : 6

Implementation of Unsupervised learning algorithm

Experiment No. 6

1. **Aim:** Implementation of unsupervised learning algorithm – Hebbian Learning

2. **Objectives:**

   - To become familiar with neural networks learning algorithms from available examples.
   - To give design methodologies for artificial neural networks.
   - Provide knowledge of un-supervised learning in neural networks.

3. **Outcomes:** The student will be able to,

   - Explain the differences between networks for supervised and unsupervised learning.
   - Understand how ANNs can be designed and trained.
   - Define the terms: unit, weight, activation function, threshold and architecture as they relate to ANNs.
   - Apply the knowledge of computing to engineering discipline to solve the problems of the neural network domains.

4. **Software Required:** C/ C++/JAVA/ MATLAB

5. **Theory:**

Unsupervised Learning Algorithm:

These types of model are not provided with the correct results during the training. It can be used to cluster the input data in classes on the basis of their statistical properties only.

The labelling can be carried out even if the labels are only available for a small number of objects represented of the desired classes. All similar input patters are grouped together as clusters. If matching pattern is not found, a new cluster is formed.

In contrast to supervised learning, unsupervised or self-organized learning does not require an external teacher. During the training session, the neural network receives a

number of different patterns & learns how to classify input data into appropriate categories. Unsupervised learning tends to follow the neuro-biological organization of brain. It aims to learn rapidly & can be used in real-time.

Hebbian Learning:

In 1949, Donald Hebb proposed one of the key ideas in biological learning, commonly known as Hebb's Law. Hebb's Law states that if neuron i is near enough is excite enough to excite neuron j & repeatedly participates in its activation, the synaptic connection between these two neurons is strengthened & neuron j becomes more sensitive to stimuli from neuron i.

Hebb's Law can be represented in the form of two rules:

1. If two neurons on either side of a connection are activated synchronously, then the weight of that connection is increased.

2. If two neurons on either side of a connection are activated asynchronously, then the weight of that connection is decreased.

Hebb's law provide basis for learning without a teacher. Learning here is a local phenomenon occurring without feedback from the environment.

- Using Hebb's Law we can express the adjustment applied to weight w_{ij} at iteration p in the following form:

$$\Delta w_{ij}(p) = F[y_i(p), x_i(p)]$$

- As a special case, we can represent Hebb's Law as follows:

$$\Delta w_{ij}(p) = \alpha y_i(p) x_i(p)$$

Where α is the learning rate parameter.

- Hebbian learning implies that weights can only increase. To resolve this problem, we might impose a limit on the growth of synaptic weights. It can be done by introducing non-linear forgetting factor into Hebb's Law:

$$\Delta w_{ij}(p) = \alpha y_i(p) x_i(p) - \varphi y_i(p), w_{ij}(p)$$

Where φ is the forgetting factor.

6. Hebbian learning algorithm

Step 1: Initialization

Set initial synaptic weights and thresholds to small random values, say in an interval [0,1].

Step 2: Activation

Compute the neuron output at iteration p

$$y_j(p) = \sum_{i=1}^{n} x_i(p)\, w_{ij}(p) - \theta_j$$

Where n is number of neuron inputs, &θ_j is the threshold value of neuron j.

Step 3: Learning

Update the weights in the network

$$w_{ij}(p+1) = w_{ij}(p) + \Delta\, w_{ij}(p)$$

Where $\Delta w_{ij}(p)$ is the weight correction at iteration p.

$$\Delta w_{ij}(p) = \varphi\, y_j(p)[\lambda\, x_i(p) - w_{ij}(p)]$$

Step 4: Iteration

Increase iteration p by one, go back to step 2.

7. Conclusion:

Unsupervised Hebbian learning algorithm is implemented which does not require supervisor. It update the weights the accordingly if error comes and train the network.

8. Viva Questions:

- What is unsupervised training?
- How Artificial Neurons learns?
- What is the difference between neural network and fuzzy logic?

9. References:

1. S.N.Sivanandam, S.N.Deepa "Principles of Soft Computing" Second Edition, Wiley Publication.
2. S.Rajasekaran and G.A.Vijayalakshmi Pai "Neural Networks, Fuzzy Logic and GeneticAlgorithms" PHI Learning.
3. Hagan, Demuth, Beale,"Neural Network Design" CENGAGE Learning, India Edition.
4. Satish Kumar, “Neural Networks –A classroom approach”, Second Edition,TMH

Soft Computing

Experiment No. : 7

Implementation of Simple Genetic Application

Experiment No. 7

1. **Aim:** Implementation Genetic Application – Match Word Finding**.**

2. **Objectives:**

   - To familiarize with Mathematical foundations for Genetic algorithm, operator.
   - To study the Applications of Genetic Algorithms.

3. **Outcomes:** The student will be able to,

   - Creating an understanding about the way the GA is used and the domain of application.
   - To appreciate the use of various GA operators in solving different types of GA problems.
   - Match the industry requirements in the domains of Programming and Networking with the required management skills.

4. **Software Required:** C/C++/JAVA/MATLAB

5. **Theory:**
   Genetic algorithm:

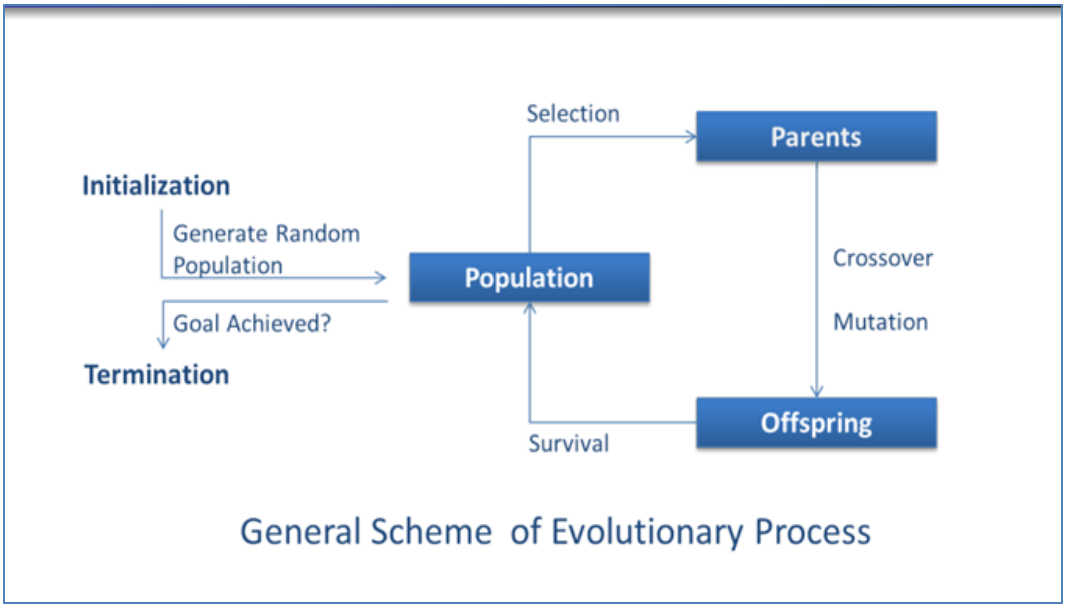
   - Genetic algorithm is a search technique used in computing to find true or approximate solutions to approximate solutions to optimization & search problems.
   - Genetic algorithms are inspired by Darwin's theory about evolution. Solution to a problem solved by genetic algorithms is evolved.
   - Algorithm is started with a set of solutions (represented by chromosomes) called population. Solutions from one population are taken and used to form a new population. This is motivated by a hope, that the new population will be better than the old one. Solutions which are selected to form new solutions (offspring) are selected according to their fitness - the more suitable they are the more

chances they have to reproduce.

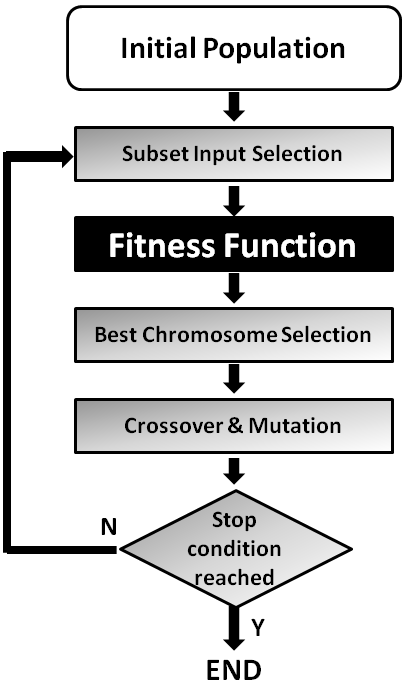
- This is repeated until some condition (for example number of populations or improvement of the best solution) is satisfied.

Outline of the Basic Genetic Algorithm

1. **[Start]** Generate random population of *n* chromosomes (suitable solutions for the problem)
2. **[Fitness]** Evaluate the fitness *f(x)* of each chromosome *x* in the population
3. **[New population]** Create a new population by repeating following steps until the new population is complete
    1. **[Selection]** Select two parent chromosomes from a population according to their fitness (the better fitness, the bigger chance to be selected)
    2. **[Crossover]** With a crossover probability cross over the parents to form a new offspring (children). If no crossover was performed, offspring is an exact copy of parents.
    3. **[Mutation]** With a mutation probability mutate new offspring at each locus (position in chromosome).
    4. [**Accepting**] Place new offspring in a new population
    5. [**Replace**] Use new generated population for a further run of algorithm
    6. [**Test**] If the end condition is satisfied, stop, and return the best solution in current population
    7. [**Loop**] Go to step 2



Flowchart



Problem Statement: Match Word Finding

Here we try to guess a word from the given population of word.

6. Algorithm:

Match Word Finding Algorithm:

Step 1: Select the word to be guessed

This value is taken through user input.

Step 2: Initialize the population

User inputs the population.

Step 3: Evaluate the population

Fitness is assigned based on number of correct letters in correct place.

Step 4: Select breeding population

Selection is done on the basis of fitness.

Step 5: Create new population

Population is created by using uniform crossover between breeding populations.

Step 6: Check for stopping condition

Here maximum fitness value in population is checked. If it is 60%.

Step 7: If stopping condition is not true, goto Step 3;

Else return the offspring with highest fitness value.

7. Conclusion:

The match word finding algorithm is implemented using the genetic algorithms which include all the genetic algorithm operators. Genetic algorithm includes the selection, crossover, mutation operators along with fitness function.

Viva Questions:

- Name some of the existing search methods.

- What are the operators involved in a simple genetic algorithm?
- What is reproduction?
- What is crossover?

8. References:

1. S.N.Sivanandam, S.N.Deepa "Principles of Soft Computing" Second Edition, Wiley Publication.
2. S.Rajasekaran and G.A.Vijayalakshmi Pai "Neural Networks, Fuzzy Logic and GeneticAlgorithms" PHI Learning.
3. Hagan, Demuth, Beale,"Neural Network Design" CENGAGE Learning, India Edition.
4. Satish Kumar, “Neural Networks –A classroom approach”, Second Edition,TMH

Soft Computing

Experiment No. : 8

Study of ANFIS Architecture

Experiment No. 8

1. **Aim:** Study of ANFIS Architecture.

2. **Objectives:**

   - Study of hybrid systems.
   - Prepare the students for developing intelligent system.

3. **Outcomes:** The learner will be able to,

   - Aware of the use of neuro fuzzy inference systems in the design of intelligent or humanistic systems.
   - To become knowledgeable about neuro fuzzy inference systems.
   - Identify, analyze the problem and validate the solution using software and hardware.

4. **Theory:**

The adaptive network-based fuzzy inference systems (ANFIS) is used to solve problems related to parameter identification. This parameter identification is done through a hybrid learning rule combining the back-propagation gradient descent and a least-squares method.

ANFIS is basically a *graphical* network representation of Sugeno-type fuzzy systems endowed with the neural learning capabilities. The network is comprised of nodes with specific functions collected in layers. ANFIS is able to construct a network realization of IF / THEN rules.

Consider a Sugeno type of fuzzy system having the rule base

1. If x is $A1$and y is $B1$, then $f_1 = c_{11}x+c_{12}y+c_{10}$
2. If x is $A2$and y is $B2$, then $f_2 = c_{21}x+c_{22}y+c_{20}$

Let the membership functions of fuzzy sets Ai, Bi, i=1,2, be , $\mu_{Ai}\mu_{Bi}$.

In evaluating the rules, choose *product* for T-norm (logical *and*).

1. Evaluating the rule premises results in

$$w_i = \mu_{A_i}(x)\mu_{B_i}(y),\ i = 1,2.$$

2. Evaluating the implication and the rule consequences gives

$$f(x,y) = \frac{w_1(x,y)f_1(x,y) + w_2(x,y)f_2(x,y)}{w_1(x,y) + w_2(x,y)}.$$

Or leaving the arguments out

$$f = \frac{w_1 f_1 + w_2 f_2}{w_1 + w_2}$$

This can be separated to phases by first defining

$$\overline{w}_i = \frac{w_i}{w_1 + w_2}$$

Then f can be written as

$$f = \overline{w}_1 f_1 + \overline{w}_2 f_2$$

All computations can be presented in a diagram form. ANFIS normally has 5 layers of neurons of which neurons in the same layer are of the same function family.

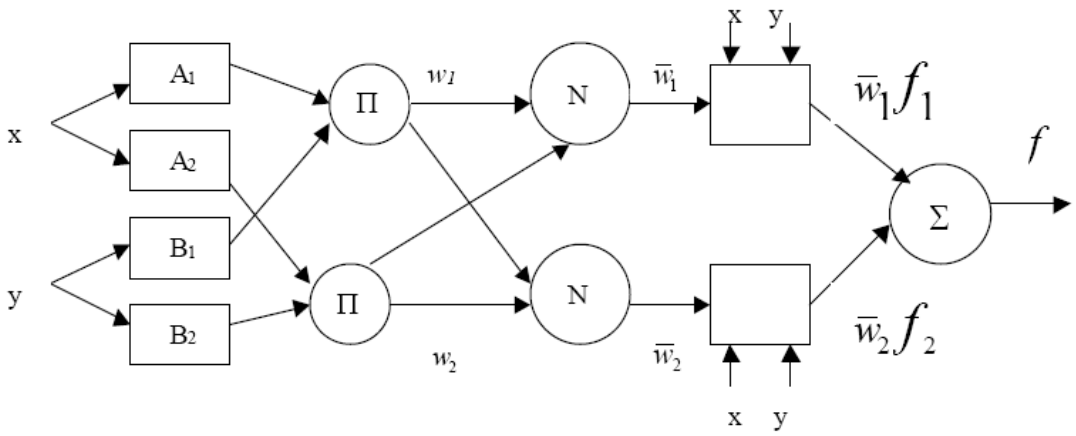


Figure: Structure of the ANFIS network.

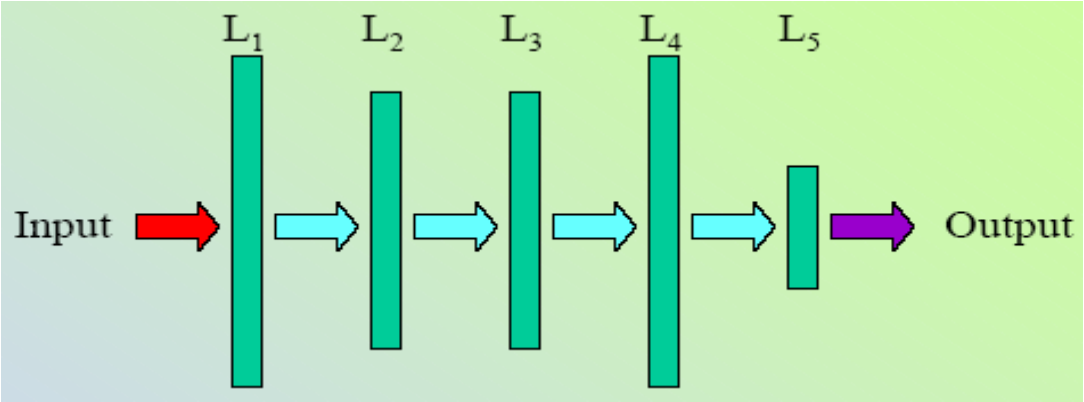


Figure : ANFIS Architecture

Layer 1 (L1): Each node generates the membership grades of a linguistic label.

An example of a membership function is the generalised bell function:

where {a, b, c} is the parameter set. As the values of the parameters change, the shape of the bell-shaped function varies. Parameters in that layer are called premise parameters.

Layer 2 (L2): Each node calculates the firing strength of each rule using the min or prod operator. In general, any other fuzzy AND operation can be used.

Layer 3 (L3): The nodes calculate the ratios of the rule are firing strength to the sum of all the rules firing strength. The result is a normalised firing strength.

Layer 4 (L4): The nodes compute a parameter function on the layer 3 output. Parameters in this layer are called consequent parameters.

Layer 5 (L5): Normally a single node that aggregates the overall output as the summation of all incoming signals.

5. Algorithm:

When the premise parameters are fixed, the overall output is a linear combination of the consequent parameters. In symbols, the output f can be written as

which is linear in the consequent parameters cij (i = 1,2, j = 0,1,2). A hybrid algorithm adjusts the consequent parameters cij in a forward pass and the premise parameters {ai, bi, ci} in a backward pass (Jang et al., 1997). In the forward pass the network inputs propagate forward until layer 4, where the consequent parameters are identified by the least-squares method. In the backward pass, the error signals propagate backwards and the premise parameters are updated by gradient descent.

Because the update rules for the premise and consequent parameters are decoupled in the hybrid learning rule, a computational speedup may be possible by using variants of the gradient method or other optimisation techniques on the premise parameters.

6. Conclusion:

This study experiments describe the architecture of neuro fuzzy systems. Fuzzy rule based system includes the model like sugenor type fuzzy which is having neural learning capabilities.

7. Viva Questions:

- What are the hybrid systems?
- What are fuzzy inference systems?
- How neuro fuzzy inference systems work?

8. References:

1. S.N.Sivanandam, S.N.Deepa "Principles of Soft Computing" Second Edition, Wiley Publication.
2. S.Rajasekaran and G.A.Vijayalakshmi Pai "Neural Networks, Fuzzy Logic and GeneticAlgorithms" PHI Learning.
3. Hagan, Demuth, Beale,"Neural Network Design" CENGAGE Learning, India Edition.
4. Satish Kumar, “Neural Networks –A classroom approach”, Second Edition,TMH

Soft Computing

Experiment No. : 9

Study of Derivative-free Optimization

Experiment No. 9

1. **Aim:** Study of Derivative-free Optimization.

2. **Objectives:** From this experiment, the student will be able to,

    - Study of hybrid systems.
    - Prepare the students for developing intelligent system.

3. **Outcomes:** The learner will be able to,

    - Aware of the use of neuro fuzzy inference systems in the design of intelligent or humanistic systems.
    - To become knowledgeable about neuro fuzzy inference systems.
    - An ability to apply knowledge of computing and use of current computing techniques appropriate to the discipline.

4. **Theory:**

Optimization problems defined by functions for which derivatives are unavailable or available at a prohibitive cost are appearing more and more frequently in computational science and engineering. Increasing complexity in mathematical modelling, higher sophistication of scientific computing, and abundance of legacy codes are some of the reasons why derivative-free optimization is currently an area of great demand. Difficulties and challenges arise from multiple sources: expensive function evaluation, black-box/legacy codes, noise and uncertainty, unknown a priori function domains, and hidden constraints.

- **Derivative free Optimization**

Derivative free optimization is a subject of mathematical optimization. It may refer to problems for which derivative information is unavailable, unreliable or impractical to obtain (derivative-free optimization problems), or methods that do not use derivatives (derivative-free optimization methods)

- **Derivative free Optimization algorithm**

- Genetic algorithms (GA)
- Simulated Annealing (SA)

Genetic algorithms (GA)

In the field of artificial intelligence, a genetic algorithm (GA) is a search heuristic that mimics the process of natural selection. This heuristic (also sometimes called a metaheuristic) is routinely used to generate useful solutions to optimization and search problems.[1] Genetic algorithms belong to the larger class of evolutionary algorithms (EA), which generate solutions to optimization problems using techniques inspired by natural evolution, such as inheritance, mutation, selection and crossover.

Optimization problems

In a genetic algorithm, a population of candidate solutions (called individuals, creatures, or phenotypes) to an optimization problem is evolved toward better solutions. Each candidate solution has a set of properties (its chromosomes or genotype) which can be mutated and altered; traditionally, solutions are represented in binary as strings of 0s and 1s, but other encodings are also possible.[2]

The evolution usually starts from a population of randomly generated individuals, and is an iterative process, with the population in each iteration called a generation. In each generation, the fitness of every individual in the population is evaluated; the fitness is usually the value of the objective function in the optimization problem being solved. The more fit individuals are stochastically selected from the current population, and each individual's genome is modified (recombined and possibly randomly mutated) to form a new generation. The new generation of candidate solutions is then used in the next iteration of the algorithm. Commonly, the algorithm terminates when either a maximum number of generations has been produced, or a satisfactory fitness level has been reached for the population.

A typical genetic algorithm requires:

a genetic representation of the solution domain,
a fitness function to evaluate the solution domain.
A standard representation of each candidate solution is as an array of bits.[2] Arrays of other types and structures can be used in essentially the same way. The main property that makes these genetic representations convenient is that their parts are easily aligned

due to their fixed size, which facilitates simple crossover operations. Variable length representations may also be used, but crossover implementation is more complex in this case. Tree-like representations are explored in genetic programming and graph-form representations are explored in evolutionary programming; a mix of both linear chromosomes and trees is explored in gene expression programming.

Once the genetic representation and the fitness function are defined, a GA proceeds to initialize a population of solutions and then to improve it through repetitive application of the mutation, crossover, inversion and selection operators.

Initialization

The population size depends on the nature of the problem, but typically contains several hundreds or thousands of possible solutions. Often, the initial population is generated randomly, allowing the entire range of possible solutions (the search space). Occasionally, the solutions may be "seeded" in areas where optimal solutions are likely to be found.

Selection

During each successive generation, a proportion of the existing population is selected to breed a new generation. Individual solutions are selected through a fitness-based process, where fitter solutions (as measured by a fitness function) are typically more likely to be selected. Certain selection methods rate the fitness of each solution and preferentially select the best solutions. Other methods rate only a random sample of the population, as the former process may be very time-consuming.

The fitness function is defined over the genetic representation and measures the quality of the represented solution. The fitness function is always problem dependent. For instance, in the knapsack problem one wants to maximize the total value of objects that can be put in a knapsack of some fixed capacity. A representation of a solution might be an array of bits, where each bit represents a different object, and the value of the bit (0 or 1) represents whether or not the object is in the knapsack. Not every such representation is valid, as the size of objects may exceed the capacity of the knapsack. The fitness of the solution is the sum of values of all objects in the knapsack if the representation is valid, or 0 otherwise.

In some problems, it is hard or even impossible to define the fitness expression; in these cases, a simulation may be used to determine the fitness function value of a phenotype (e.g. computational fluid dynamics is used to determine the air resistance of a vehicle

whose shape is encoded as the phenotype), or even interactive genetic algorithms are used.

Simulated annealing (SA)

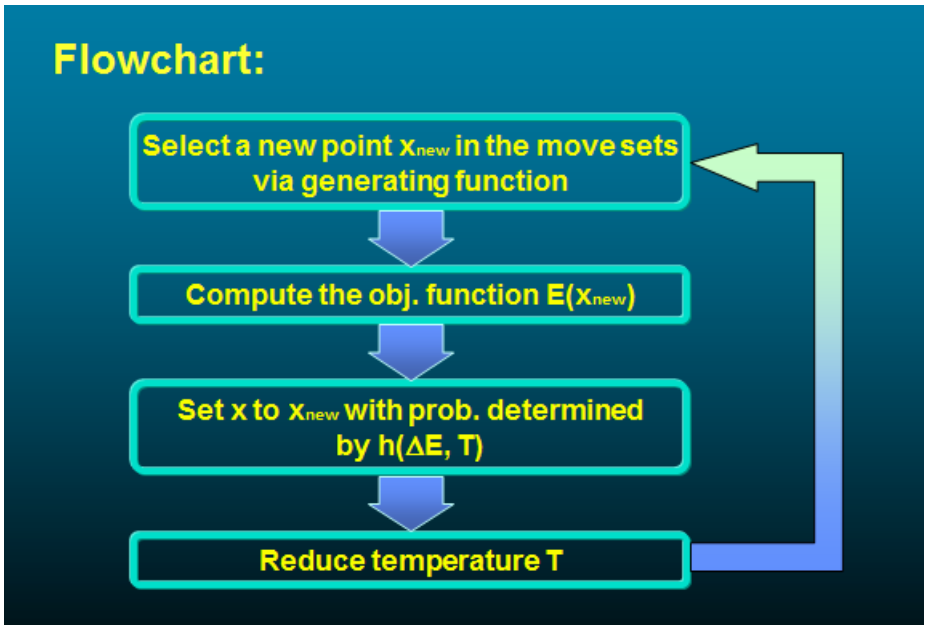
is a probabilistic technique for approximating the global optimum of a given function. Specifically, it is a metaheuristic to approximate global optimization in a large search space. It is often used when the search space is discrete (e.g., all tours that visit a given set of cities). For problems where finding the precise global optimum is less important than finding an acceptable local optimum in a fixed amount of time, simulated annealing may be preferable to alternatives such as brute-force search or gradient descent.

Simulated annealing interprets slow cooling as a slow decrease in the probability of accepting worse solutions as it explores the solution space. Accepting worse solutions is a fundamental property of metaheuristics because it allows for a more extensive search for the optimal solution.

Terminology:

1. Objective function E(x): function to be optiimized
2. Move set: set of next points to explore
3. Generating function: to select next point
4. Acceptance function h(DE, T): to determine if the selected point should be accept or not. Usually h(DE, T) = 1/(1+exp(DE/(cT)).
5. Annealing (cooling) schedule: schedule for reducing the temperature T

Flowchart



6. Conclusion:

This study experiments describe the various techniques used for derivative free optimization. It also describes how to use optimization techniques in soft computing domain.

7. Viva Questions:

- What is the meaning of optimization?
- What do you mean by simulated annealing?
- What are the algorithms used for derivative free optimization?
- What are the algorithms used for derivative based optimization?

8. References:

1. S.N.Sivanandam, S.N.Deepa "Principles of Soft Computing" Second Edition, Wiley Publication.

2. S.Rajasekaran and G.A.Vijayalakshmi Pai "Neural Networks, Fuzzy Logic and GeneticAlgorithms" PHI Learning.
3. Hagan, Demuth, Beale,"Neural Network Design" CENGAGE Learning, India Edition.
4. Satish Kumar, “Neural Networks –A classroom approach”, Second Edition,TMH

Soft Computing

Experiment No. : 10

Study of research paper on Soft Computing

Experiment No. 10

1. **Aim:** Study of research paper on Soft Computing.

2. **Objectives:** From this experiment, the student will be able to,

   - Understand the research trends in soft computing.
   - Crate awareness among the students towards the recent trends in soft computing.

3. **Outcomes:** The learner will be able to,

   - Understand the importance of research in soft computing.
   - To become knowledgeable for developing soft computing applications.
   - An ability to match the industry requirements in the domains of Programming with the required management skills.to analyze the local and global impact of computing on individuals, organizations, and society.

4. **Theory:**

   Students can find the research papers based on the artificial neural network, hybrid systems, genetic algorithm, fuzzy system, fuzzy logic, fuzzy inference system etc.
   Students need to search recent papers on any of the above mentioned topics, study it and prepare presentation on the same

5. **Conclusion:**

   Through this experiment, we have understood the recent advancements and applications of various subdomains of soft computing.

6. **References:**

1. www.ieeeexplore.com
2. www.Scicencedirect.com
3. Any open access journal