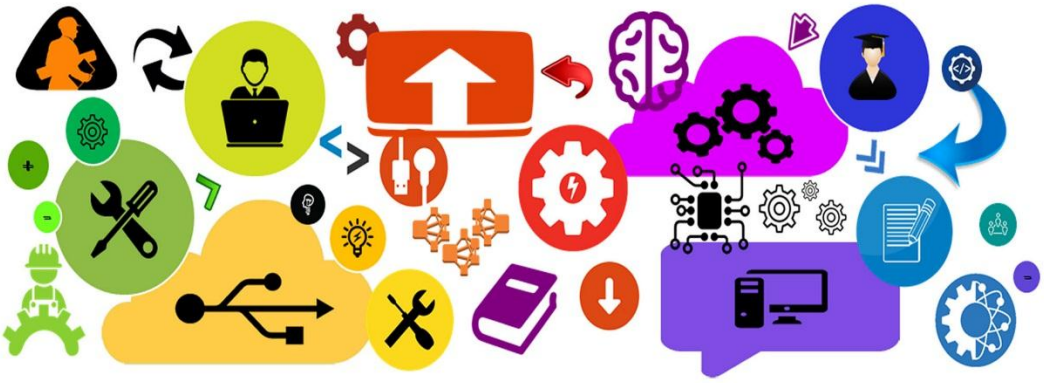
www.ToppersSolutions.com Support@ToppersSolutions.com

Topper's Solutions

....In Search of Another Topper



Soft Computing

Sem-7 (Computer)

(As Per Revised Syllabus 2012)

Topper's Solutions

…..In search of Another Topper

There are many existing paper solution available in market, but Topper's Solution is the one which students will always prefer if they refer… ;) Topper's Solution is not just paper solution, it includes many other important questions which are important from examination point of view. Topper's Solutions are the solution written by the Toppers for the students to be the upcoming Topper of the Semester.

It has been said that "**Action Speaks Louder than Words**" So Topper's Solution works on same principle. Diagrammatic representation of answer is considered to be easy & quicker to understand. So our major focus is on diagrams & representation how answers should be answered in examinations.

Why Topper's Solution:

- Point wise answers which are easy to understand & remember.
- Diagrammatic representation for better understanding.
- Additional important questions from university exams point of view.
- Covers almost every important question.
- In search of another topper.

****** Education is Free…. But its Technology used & Efforts utilized which charges ******

It takes lot of efforts for searching out each & every question and transforming it into Short & Simple Language. Entire Topper's Solutions Team is working out for betterment of students, do help us.

"Say No to Xerox…."

With Regards,

Topper's Solutions Team.

CHAPTER – 1: INTRODUCTION TO SOFT COMPUTING.

Q1] DEFINE SOFT COMPUTING? DISTINGUISH BETWEEN SOFT COMPUTING AND HARD COMPUTING?

ANS: **[5M – DEC15 & MAY16]**

SOFT COMPUTING:

1. Soft Computing is sometimes referred as **Computational Intelligence**.
2. The main goal of soft computing is to develop **intelligence machines** to provide solutions to **real world problems**, which are not modeled or too difficult to model mathematically.
3. It is an emerging approach to the computing.
4. The Role Model for Soft Computing is **Human Mind**.

Definition:

Soft Computing is an Computational Intelligence which uses the remarkable ability of human mind to **reason & learn** in an environment of uncertainty & imprecision.

Soft Computing Paradigms:

- **Fuzzy Set:** For Knowledge Representation via Fuzzy IF – Then Rule.
- **Neural Networks:** For learning & adaptation.
- **Genetic Algorithms:** For Evolutionary Computation.

DISTINGUISH BETWEEN SOFT COMPUTING & HARD COMPUTING:

Soft Computing	Hard Computing
It is also known as Computational Intelligence.	It is also known as Conventional Computing.
Soft Computing can evolve its own programs.	Hard Computing requires programs to be written.
It uses multivalued or fuzzy logic.	It uses two-valued logic.
It can deal with ambiguous and noisy data.	It requires exact input data.
It allows parallel computations	It is strictly sequential.
Note: For above question write 5 points. Some other points are given below if asked for 10 marks.	

It gives approximate answers.	It gives precise answer.
It is deterministic.	It is stochastic.
It uses soft constraints.	It uses real – time constraints.
It is useful for routine tasks that are not critical.	It is useful in critical systems.
Need of robustness rather than accuracy.	Need of accuracy & precision in calculations and outcomes.

Q2] APPLICATIONS OF SOFT COMPUTING.

ANS:

1. Hand writing recognition.
2. Image processing & Data compression.
3. Power systems.
4. Fuzzy Logic Control.
5. Speech & Vision Recognition Systems.
6. Machine Learning Applications

CHAPTER – 2: NEUTRAL NETWORKS.

Q1] WHAT IS LEARNING IN NEUTRAL NETWORKS? DIFFERENTIATE BETWEEN SUPERVISED & UNSUPERVISED LEARNING.

ANS: **[5M – DEC15] & [10M - MAY16]**

NEUTRAL NETWORKS:

1. It is also called as **Artificial Neutral Network** (ANN) in Soft Computing.
2. Neutral Network is a highly interconnected network of a large number of processing elements called neurons in an architecture inspired by the brain.

LEARNING:

1. Learning is sometimes also called as **training in Neutral Networks**.
2. Learning implies that a processing unit is capable of changing its input/output behavior as a result of changes in the environment.
3. Learning in a network is a process of forcing a network to yield particular response to a specific input.
4. It results in the outcome of desired response.

Figure 2.1: Learning In Neutral Network.

TYPES OF LEARNING:

***** Note: Use 1 – 2 figures in ever answer in exam. It makes the answer look logic & Clear. Use your know creativity for the same. *****

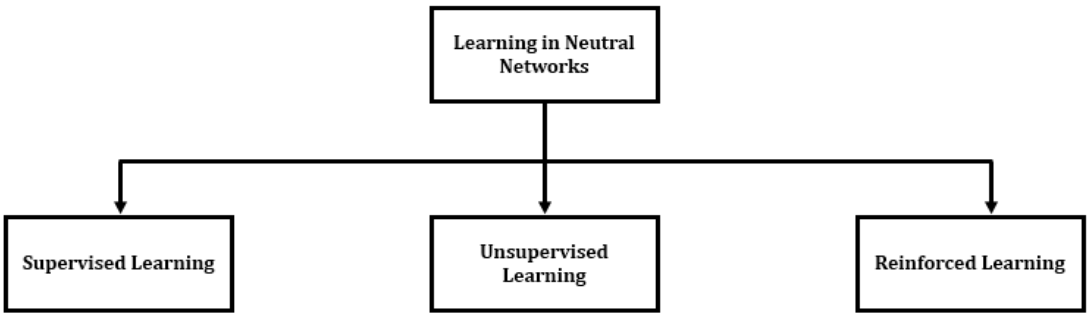
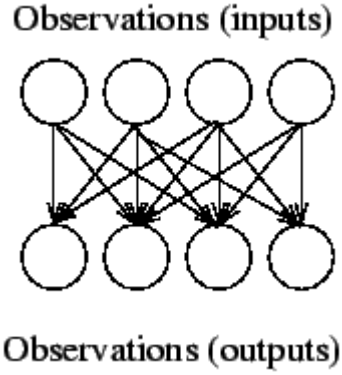
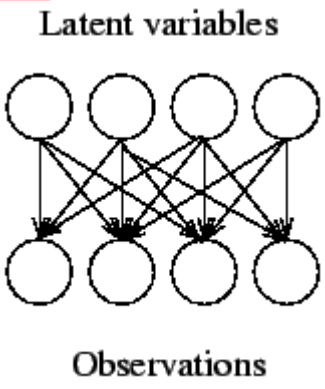


Figure 2.2: Types of Learning.

DIFFERENTIATE BETWEEN SUPERVISED & UNSUPERVISED LEARNING:

Supervised Learning	Unsupervised Learning
Supervised Learning can corresponds to learning in class room in the presence of teacher. Thus questions can be answered.	Unsupervised Learning corresponds to learning from a video tape lecture. Thus questions cannot be answered.
Supervised Learning usually provides the classification and recognition of the patterns.	Unsupervised Learning usually preforms the clustering of the training patterns.
In Supervised Learning, Desired output is known.	In Unsupervised Learning, desired output is unknown.
Error information can be used.	Error information cannot be used.
A training set is required.	A training set is not required.
A reward/punishment scheme is used.	It uses observation for learning.
Neutral Network Model for Supervised Learning is Perception & Feed – Forward Neutral Network.	Neutral Network Model for Unsupervised Learning is Self-Organizing Maps.
In supervised Learning, one set of observations, called inputs, is assumed to be the cause of another set of observations, called outputs.	In Unsupervised Learning all observations are assumed to be caused by a set of latent variables.
Diagram: 	**Diagram:** 
Example: Perceptron Learning Rule.	**Example:** Hebbian Learning Rule.

Q2] WINNER TAKE ALL LEARNING RULE IN NEURAL NETWORK.

ANS: [10M - MAY16]

1. Winner-Take-All is a strategy used in **Competitive Learning**.
2. It is used for **Unsupervised Network Training**.
3. Winner Takes All Learning Rule is originated from **Outstar Learning Rule.**
4. This learning rule differs substantially from all other learning rules.
5. It is used for learning statistical properties of inputs.
6. The learning is based on the premise that one of the neurons in the layer, say N^{th} neuron has maximum response due to input X as shown in figure 2.3.
7. This neuron is declared as the Winner Neuron with a weight.

$$W_m = [W_1 \quad W_2 \quad W_3 \quad \ldots\ldots W_{mn}]$$

8. The Increment is computed as follows:

$$\Delta W_m = \alpha (X - W_m)$$

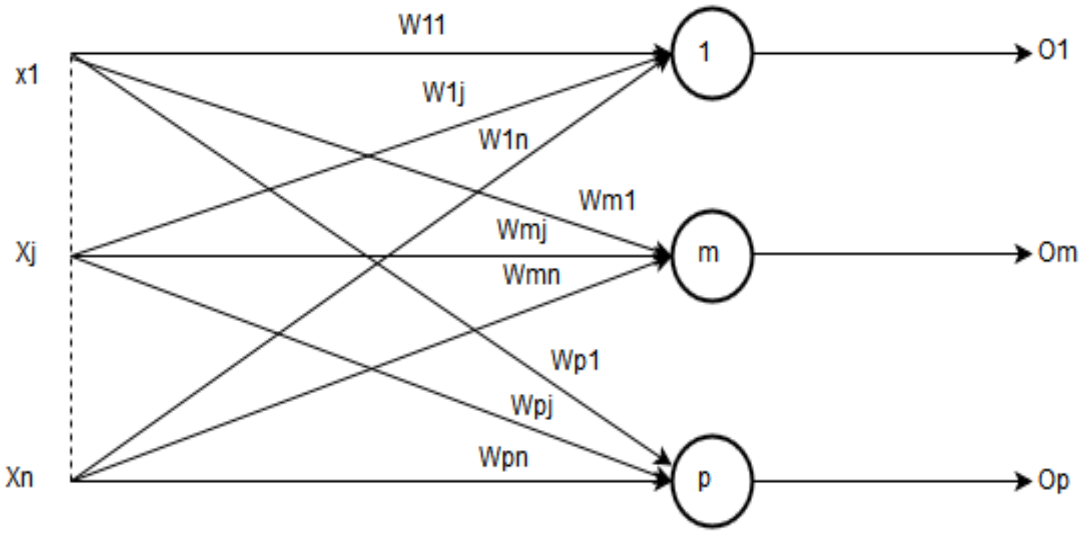


Figure 2.3: Winner Takes All.

9. In figure 2.3, 'm' is winner neuron.
10. The winner selection is based on the following criterion of maximum activating among all 'p' neurons participating in competitive environment.

$$W_m{}^t X = \text{Max} (W_i{}^t X) \ldots\ldots\ldots\ldots\ldots\ldots \text{(Where i = 1, 2, ... p)}$$

Q3] LEARNING VECTOR QUANTIZATION

ANS: [10M - MAY16]

LEARNING VECTOR QUANTIZATION:

1. Learning Vector Quantization was developed by **Kohonen**.
2. It is one of the most frequently used **unsupervised clustering algorithms**.
3. It is a supervised version of **vector quantization**.
4. Learning Vector Quantization (LVQ), is a prototype-based supervised classification algorithm.
5. It is used for pattern classification.
6. In LVQ each output unit represents a particular class or category.
7. In LVQ, Classes are predefined and we have a set of labelled data.
8. The goal is to determine a set of prototypes that best represent each class.
9. An LVQ system is represented by prototype W=(w(i),...,w(n))} which are defined in the feature space of observed data.
10. LVQ can be a source of great help in classifying text documents.

ARCHITECTURE:

The architecture of an LVQ neural net, is essentially the same as that of a Kohonen self-organizing map (without a topological structure being assumed for the output units.)

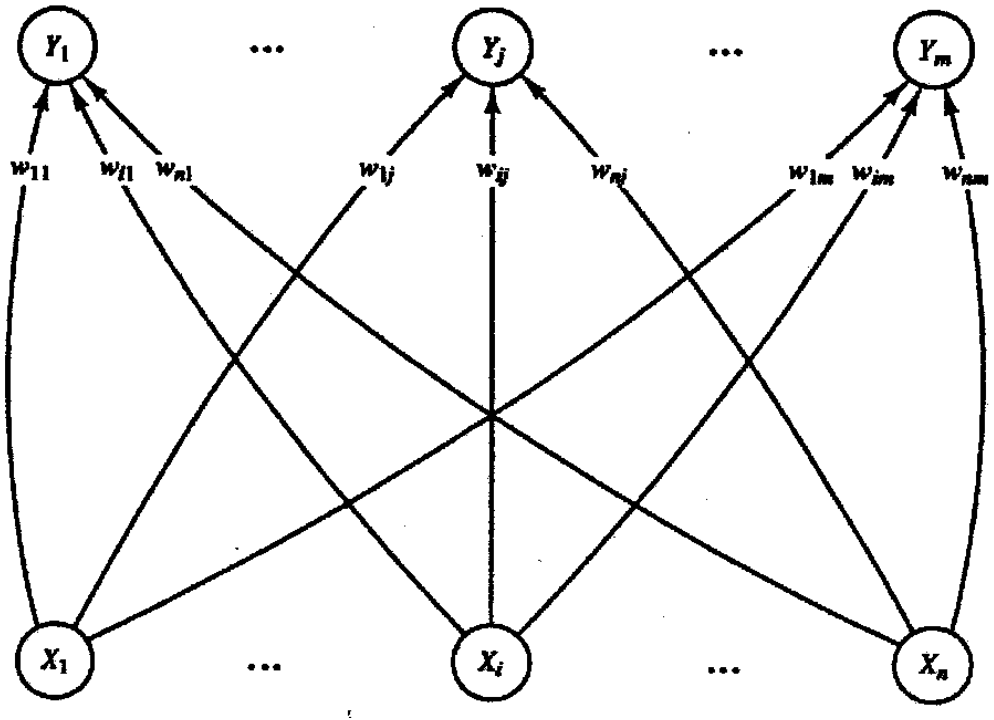


Figure 2.4: LOV Architecture.

ADVANTAGES:

- LVQ creates prototypes that are easy to interpret for experts.
- It is used in a variety of practical applications.

DISADVANTAGE:

- A key issue in LVQ is the choice of an appropriate measure of distance or similarity for training and classification.

Q4] EXPLAIN LINEAR SEPARABLE AND NON-LINEARLY SEPARABLE PATTERN WITH EXAMPLE.

ANS: **[Q1 | 10M – DEC15] & [Q2 | 5M – MAY16]**

1. If a subset of the input pattern belongs to one class (Say X_1) and the remaining subset of the input pattern to another class (Say X_2) then the objective in a pattern classification problem is to determine a set of weights $W_1, W_2 \ldots W_m$
2. Such that if the weighted sum

$$\sum_{i=1}^{M} w_i x_i > \theta, \text{ then } x = (x_1, x_2, \ldots . x_M)^T$$

   Belongs to Class X_1
3. And if

$$\sum_{i=1}^{M} w_i x_i < \theta, \text{ then } x = (x_1, x_2, \ldots . x_M)^T$$

   Belongs to Class X_2
4. The dividing surface between the two classes is given by

$$\sum_{i=1}^{M} w_i x_i = \theta$$

5. If $W^T X > 0$, Then X belongs to Class X_1.
6. If $W^T X < 0$, Then X belongs to Class X_2.
7. The equation for dividing linear hyper line is $W^T X = 0$
8. If such a weight vector exists such that $W^T X > 0$ for each $X \in X_1$ and $W^T X < 0$ for each $X \in X_2$ then the patterns sets X_1 and X_2 are **Linearly Separable**.
9. If no weight vector exists such that $W^T X > 0$ for each $X \in X_1$ and $W^T X < 0$ for each $X \in X_2$ then the patterns sets X_1 and X_2 are **Non-Linearly Separable**.

10. Figure 2.5 shows the example of Linearly separable pattern.

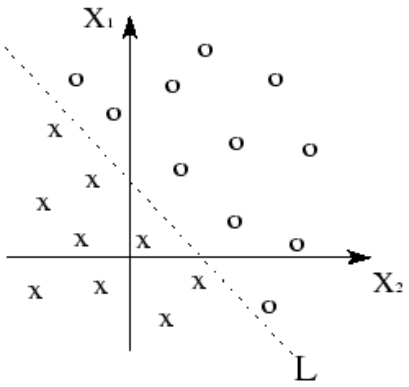


Figure 2.5: Linearly Separable pattern.

Example of Linearly Separable Pattern:

AND

X	Y	Class
0	0	0
0	1	0
1	0	0
1	1	1

OR

X	Y	Class
0	0	0
0	1	1
1	0	1
1	1	1

Example of Non-Linearly Separable Pattern:

XOR

X	Y	Class
0	0	0
0	1	1
1	0	1
1	1	0

Q5] CHARACTER RECOGNITION USING NEURAL NETWORK.

ANS: [10M - MAY16]

CHARACTER RECOGNITION:

1. Character Recognition is an application of the **Error Back Propagation Algorithm**.
2. Character Recognition techniques are used for both printed & hand written characters.
3. Figure 2.5 shows the block diagram of character recognition.

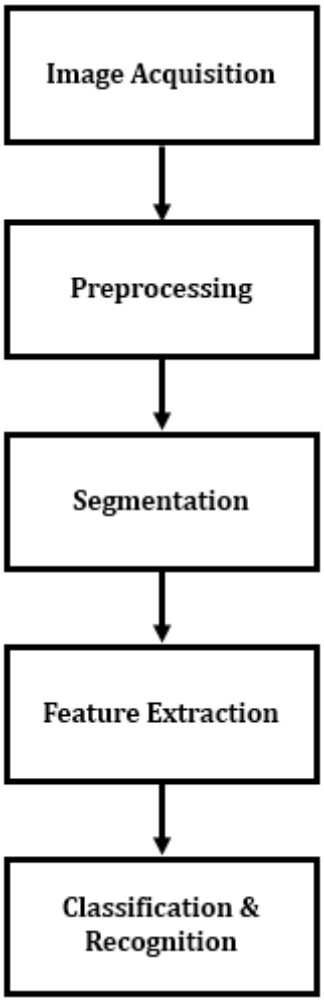


Figure 2.5: Character Recognition Technique.

I) Image Acquisition:

- In this Step, the original image is obtained & scanned.

II) Pre-Processing:

- The pre-processing is a series of operations performed on scanned input image.
- The role of pre-processing is to segment the interesting pattern from the background.
- Generally, noise filtering, smoothing and normalization should be done in this step.

III) Segmentation:

- After scanning the document, the document image is subjected to pre-processing for background noise elimination, and Binarization to generate the bit map image of the text.
- The pre-processed image is divided into lines, words and characters.
- Segmentation Includes:
  - **Line segmentation:** To separate the text lines, line segmentation is used.
  - **Word segmentation:** Word segmentation is provide the space between words.
  - **Character Recognition:** It is providing the Spacing between the characters. so it is called segmentation.

IV) Feature Extraction:

- This approach is useful when image sizes are large.
- A reduced feature representation is required to quickly complete tasks such as image matching and retrieval.

V) Classification & Recognition:

- Classification is used to take segmented input samples and classify them into group.
- Finally the segmented image is classified & recognition using various image recognition techniques.

Q6] WHAT IS MCCULLOCH PITTS NEURON MODEL WITH THE HELP OF EXAMPLE.

ANS: **[5M - MAY16]**

1. McCulloch-Pitts Neuron Model was introduced by Warren **McCulloch** and Walter **Pitts** in 1943.
2. It was the early model of an **Artificial Neuron**.
3. The McCulloch-Pitts Neural Model is also known as **Linear Threshold Gate**.
4. It is a neuron of a set of inputs I_1, I_2 … I_m and one output Y.
5. The linear threshold gate simply classifies the set of inputs into two different classes.
6. Thus the output Y is binary.
7. Such a function can be described mathematically using these equations:

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

$$y = f(Sum).$$

8. W_1, W_2 ….. W_m are weight values normalized in the range of either (0, 1) or (-1, 1).
9. Sum is the weighted sum, and ***T*** is a threshold constant.
10. The function 'F' is a linear step function at threshold ***T***.
11. The symbolic representation of the linear threshold gate is shown in figure 2.6

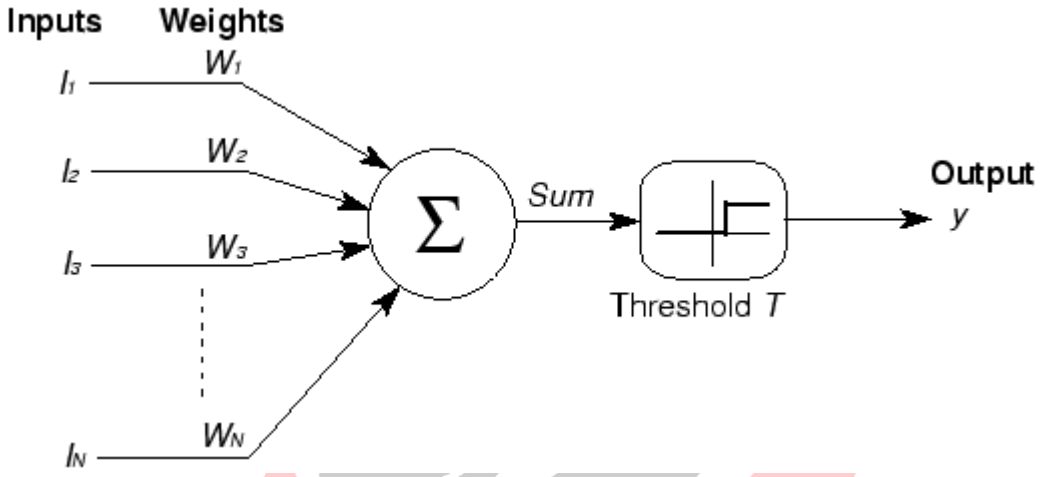


Figure 2.6: Symbolic Illustration of Linear Threshold Gate.

Q7] EXPLAIN ERROR BACK PROPORTIONAL ALGORITHM WITH HELP OF FLOWCHART.

ANS: [10M - MAY16]

ERROR BACK PROPORTIONAL ALGORITHM:

1. Error Back Proportional Algorithm is a **Supervised Learning Algorithm**.
2. It is common method of training **Artificial Neural Networks** used in conjunction with an optimization method such as **Gradient Descent**.
3. Back propagation requires a known, desired output for each input value in order to calculate the loss function gradient.
4. It is therefore usually considered to be a supervised learning method, although it is also used in some unsupervised networks such as **auto encoders**.
5. The algorithm allows experiential acquisition of input mapping knowledge within multi-layer networks.
6. Back propagation requires that the activation function used by the artificial neuron be differentiable.

STEPS:

Given are P training sets $\{(Z_1, D_1), (Z_2, D_2)\ldots.. (Z_p, D_p)\}$

Where Z = i x 1, D = K x i, Y = j x 1 & O = K x 1

i^{th} Component of Z = -1 & j^{th} Component of Y = -1

Step 1: η > 0 and some E_{max} is choose W & V some small random values

W = K x j

V = j x i

q = 1, p = 1 & E = 0

Step 2: Training Cycle starts here

Input: Z = Z x p & D = D x p

Output: Yj = F(Vj x Z)

Ok = F (Wk x Y)

Step 3: Compute error

$$E = \frac{1}{2}(d_k - O_k)^2 + E$$

Step 4: Compute Error Signal

$$\delta O_k = \frac{1}{2}(d_k - O_k)(1 - O_k^2)$$
$$\delta y_j = \frac{1}{2}(1 - y_j^2)\sum_{k=1}^{K} \delta O_k . W_{kj}$$

Step 5: Output layer weights are adjusted

$$W_{kj} = W_{kj} + n\delta O_k Y_j$$

Step 6: Hidden layer weights are adjusted

$$V_j i = V_j i + \eta \delta y_i z_i$$

Step 7:

If p < P

P = P + 1

q = q + 1

& go to step 2

Else step 8

Step 8:

If E < E_{max} Terminate

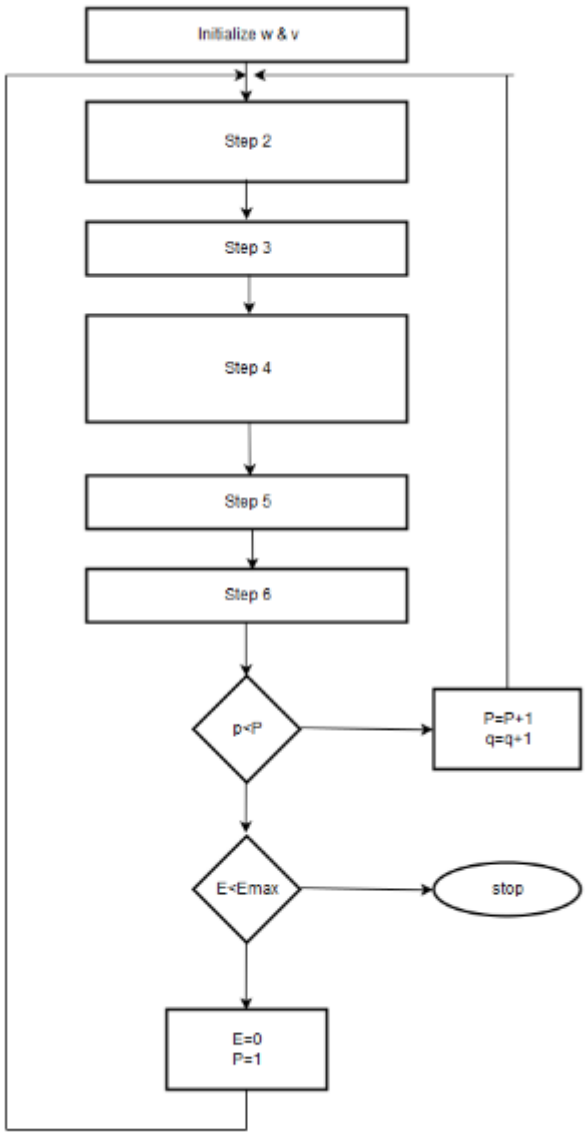
Else If E > E_{max}

E = 0

p = 1

& go to step 2

FLOWCHART:



Q8] WHAT IS SELF-ORGANIZING MAP? DRAW & EXPLAIN ARCHITECTURE OF KOHONEN SELF ORGANIZATION FEATURE MAP KSOFM.

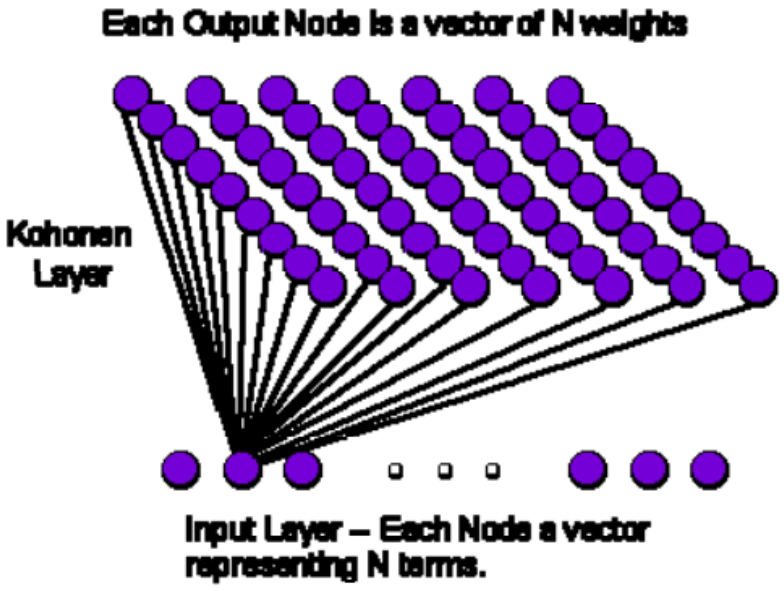
ANS: [10M – DEC15]

SELF-ORGANIZING MAP:

1. Self-Organizing Map is also called as **Self-Organizing Feature Map (SOFM)**.
2. It is a type of **Artificial Neutral Network**.
3. SOFM is **Unsupervised Learning Technique**.
4. SOFMs are **neural networks** that employ unsupervised learning methods.
5. It maps their weights to conform to the given input data with a goal of representing multidimensional data.
6. The output data is easier and understandable form for the human eye in SOFMs.

KOHONEN SELF ORGANIZATION FEATURE MAP:

1. Kohonen's SOFM is called a **Topology-Preserving Map.**
2. Kohonen's SOFMs are a type of **Unsupervised Learning**.
3. The goal is to discover some underlying structure of the data.
4. KSOFM consists of a two dimensional array of output units connected to all input nodes.
5. It works based on the property of Topology Preservation.
6. Nearby input patterns should activate nearby output units on the map.
7. KSOFM can be recognized as a **special competitive learning.**
8. Only the weight vectors of winner and its neighbor units are updated



CHAPTER – 3: FUZZY SET THEORY

Q1] EXPLAIN MAMDANI TYPE OF FUZZY INFERENCE SYSTEMS IN DETAILS

ANS: [10M - MAY16]

MAMDANI FIS:

1. The Mamdani Fuzzy Inference System was proposed by Ebrahim Mamdani in 1975.
2. Mamdani FIS is used to control a stream engine and boiler combination by synthesizing a set of fuzzy rules.

STEPS TO OBTAIN AN OUTPUT IN MAMDANI FIS:

1. Determine a set of Fuzzy Rules.
2. Fuzzify the inputs using the input membership functions.
3. Combine the fuzzified inputs according to the fuzzy rules to establish a rule strength.
4. Determine the consequence of the rule by combining the rule strength and the output membership function.
5. Combining the consequences to get an output distribution.
6. Defuzzify the output distribution.

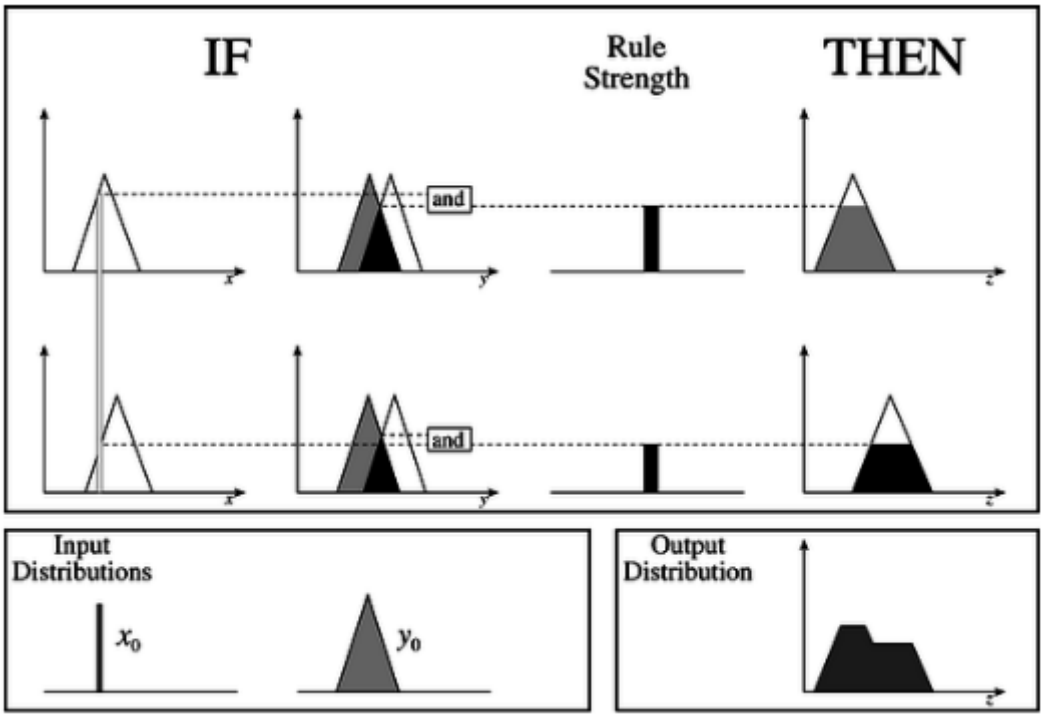


Figure 3.1: A Two Input, Two Rule Mamdani FIS with a Fuzzy Input.

- In Figure 3.1, Consider a two input Mamdani FIS with two rules.

- It Fuzzifies the two inputs by finding the intersection of the two crisp input values with the input membership function.
- It uses the minimum operator to compute the Fuzzy Input "and" for combining the two fuzzified inputs to obtain a rule strength.
- The output membership function is clipped at the rule strength.
- Finally, the maximum operator is used to compute the Fuzzy Output "or" for combining the outputs of the two rules.

Q2] EXPLAIN ANY FOUR DEFUZZIFICATION WITH SUITABLE EXAMPLE.

ANS: [10M - MAY16]

DEFUZZIFICATION:

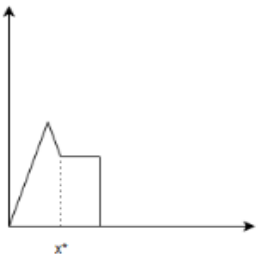
- Defuzzification refers to the way a crisp value is extracted from a fuzzy set as a representative value.
- There are different defuzzification methods depending upon continue or discrete value, convex or non-convex value, symmetric or non-symmetric wave from the various methods are:

***** Note: we have listed all Defuzzification Method. Write any four for above question. *****

I) Centroid Method:

- It is used for Convex and Non-convex Fuzzy Set.
- It is only used for Continuous Fuzzy Set.

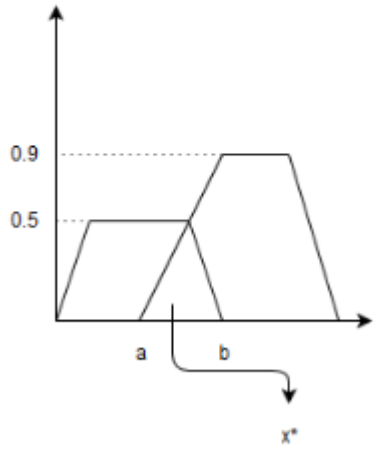
$$x^* = \frac{\int x.\mu(x)dx}{\int \mu(x)dx}$$



II) Weighted Average Method:

- It is only used for discrete values.
- It is also used only for symmetrical waveform.
- The symmetrical waveform will be in overlapped manner, in which the shaded parts comes only once.

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



III) Center of Sums:

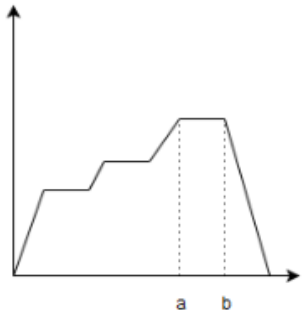
- In this method shaded part appears many times. Thus, the waveform is asymmetrical in this technique we calculate the weighted average for each shaded part and consider the output close to the center one.

$$x^* = \frac{\sum_{i=1}^{n} x_i . \sum_{k=1}^{N} \mu_{AK}(x_i) dx}{\sum_{i=1}^{n} \sum_{k=1}^{N} \mu_{AK}(x_i) dx}$$

IV) Mean of Max:

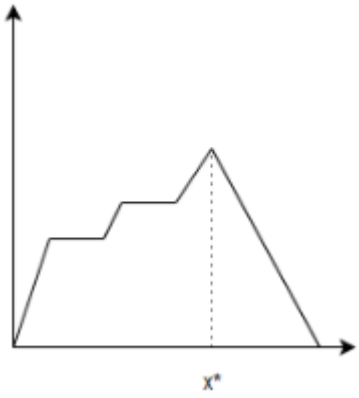
- If the defuzzified values are having more than one peak (center) we go for mean of max i.e. we consider average value.

$$x^* = \frac{a+b}{2}$$



- If the maximum value is same as that of height then there is no need of mean.

$$x^* = height$$



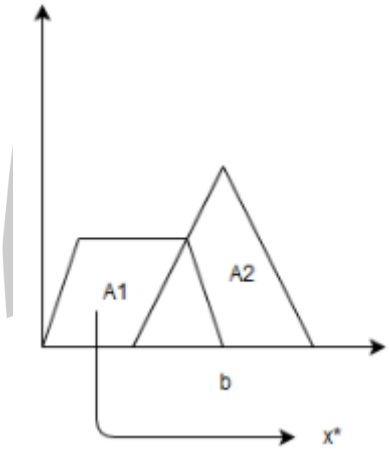
V) Max Membership Principle:

- If considers only the height of each area for defuzzification in this case the values will be discrete and non-symmetrical also the shaded area will appear more than once.

$$x^* = \sum_{i=1}^{n} x_i \cdot \sum_{k=1}^{N} \mu_{AK}(x_i)$$

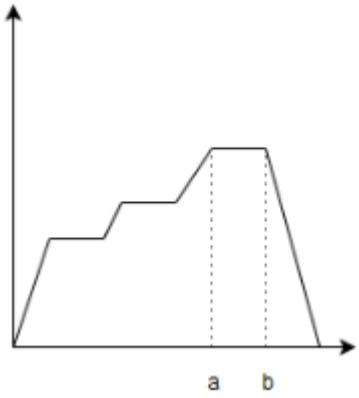
VI) Center of Largest:

- It is used only for non-convex Fuzzy set.
- If multiple waveforms are there then calculate the area of individual waveform and take the center of largest area as the final answer.



VII) First/Last of Maxima:

- It is very similar to mean of max only that first and last of maximize is consider
- When there is only one height X^* = Height it is same as mean of max.



Q3] PROVE THAT FOR BIPOLAR CONTINUOUS ACTIVATION FUNCTION F1 (NET) = 0 (1-0²)/2;

ANS: [5M – MAY16]

Delta Training rules for bipolar continuous activation function:

The activation function in the case of bipolar continuous activation function is given by

$$f(net) = \frac{2}{1 + exp(-net)} - 1$$

We obtain

$$f'(net) = \frac{2exp(-net)}{[1 + exp(-net)]^2}$$

An useful identify can be applied here

$$\frac{2exp(-net)}{[1 + exp(-net)]^2} = \frac{1}{2}(1 - 0^2)$$

Verification of identity

Letting o = f (net)

$$\frac{1}{2}(1 - 0^2) = \frac{1}{2}\left[1 - \left(\frac{1 - exp(-net)}{1 + exp(-net)^2}\right)\right]$$

$$\frac{1}{2}\left[1 - \left(\frac{1 - exp(-net)}{1 + exp(-net)^2}\right)\right] = \frac{2exp(-net)}{[1 + exp(-net)]^2}$$

LHS = RHS

The delta value for a bipolar continuous activation function is given by

$$\delta_o k = \frac{1}{2}(d_k - o_k)(1 - o_k^2)$$

Which uses the following identity for f' (net)

$$f'(net) = \frac{1}{2}(1 - o^2)$$

Hence, for bipolar continuous activation function

$$f'(net) = ½(1 - o^2)$$

Q4] PROVE THAT FOR UNIPOLAR CONTINUOUS ACTIVATION FUNCTION F1 (NET) = O (1-O);

ANS: **[5M – DEC15, MAY16]**

Delta Training rules for unipolar continuous activation function:

$$f'(net) = \frac{exp(-net)}{[1 + exp(-net)]^2}$$

This can be rewritten as

$$f'(net) = \frac{1}{1 + exp(-net)} \cdot \frac{1 + exp(-net) - 1}{1 + exp(-net)}$$

Or

$$f'(net) = 0(1 - 0)$$

Hence, for unipolar continuous activation function f1 (net) = 0 (1-0).

Q5] DETERMINE ALL α – LEVEL SETS AND STRONG α – LEVEL SETS FOR THE FOLLOWING FUZZY SET.

A = {((1, 0.2), (2, 0.5), (3, 0.8), (4, 1), (5, 0.7), (6, 0.3))}

ANS: **[5M – DEC15, MAY16]**

α – Level Sets:

The α level of a fuzzy set A is the crips set A that contains all the elements of the universal set X whose membership values in A are greater than or equal to the specified value of α.

$$A_{\alpha=0.2} = \{1, 2, 3, 4, 5, 6\}$$
$$A_{\alpha=0.3} = \{2, 3, 4, 5, 6\}$$
$$A_{\alpha=0.5} = \{2, 3, 4, 5\}$$

$$A_{\alpha=0.7} = \{3, 4, 5\}$$
$$A_{\alpha=0.8} = \{3, 4\}$$
$$A_{\alpha=1} = \{4\}$$

Strong α – Level Sets:

The Strong α level of a fuzzy set A is the crips set A that contains all the elements of the universal set X whose membership values in A are greater than or equal to the specified value of α.

$$A_{\alpha'=0.2} = \{2, 3, 4, 5, 6\}$$
$$A_{\alpha'=0.3} = \{2, 3, 4, 5\}$$
$$A_{\alpha'=0.5} = \{3, 4, 5\}$$
$$A_{\alpha'=0.7} = \{3, 4\}$$
$$A_{\alpha'=0.8} = \{4\}$$
$$A_{\alpha'=1} = \{\}$$

CHAPTER – 4: HYBRID SYSTEM.

Q1] EXPLAIN ANFIS ARCHITECTURE WITH NEAT DIAGRAM.

Q2] DRAW THE FIVE LAYER ARCHITECTURE OF ANFIS & EXPLAIN EACH LAYER IN BRIEF.

ANS: [Q1 | 10M - MAY16] & [Q2 | 5M - DEC15]

ANFIS:

1. ANFIS Stands for **Adaptive Neuro (Network Based) Fuzzy Inference System**.
2. ANFIS is a kind of Artificial Neutral Network (ANN) that is based on **Takagi - Sugeno Fuzzy Inference System**.
3. It is a hybrid system comprising of the **neutral network** and the **fuzzy logic**.
4. ANFIS is consider as **Universal Estimator**.

ANFIS ARCHITECTURE:

1. ANFIS Architecture uses two fuzzy If Then Rules based on a first order model.

   Rule 1: If x is A_1 and y is B_1 then $F_1 = P_1x + Q_1y + R_1$

   Rule 2: If x is A_2 and y is B_2 then $F_2 = P_2x + Q_2y + R_2$

Where, x and y are the inputs.

A_i and B_i are the fuzzy sets.

F_i are the outputs.

P_i, Q_i and R_i are the design parameters that are determined during the training process.

2. Here Type-3 Fuzzy Inference System Proposed by Takagi & Sugeno is used.
3. In this interference system the output of each rule is a linear combination of the input variables added by a constant term.
4. The final output is the weighted average of each rule's output.
5. The corresponding equivalent ANFIS Structure is shown in figure 4.1

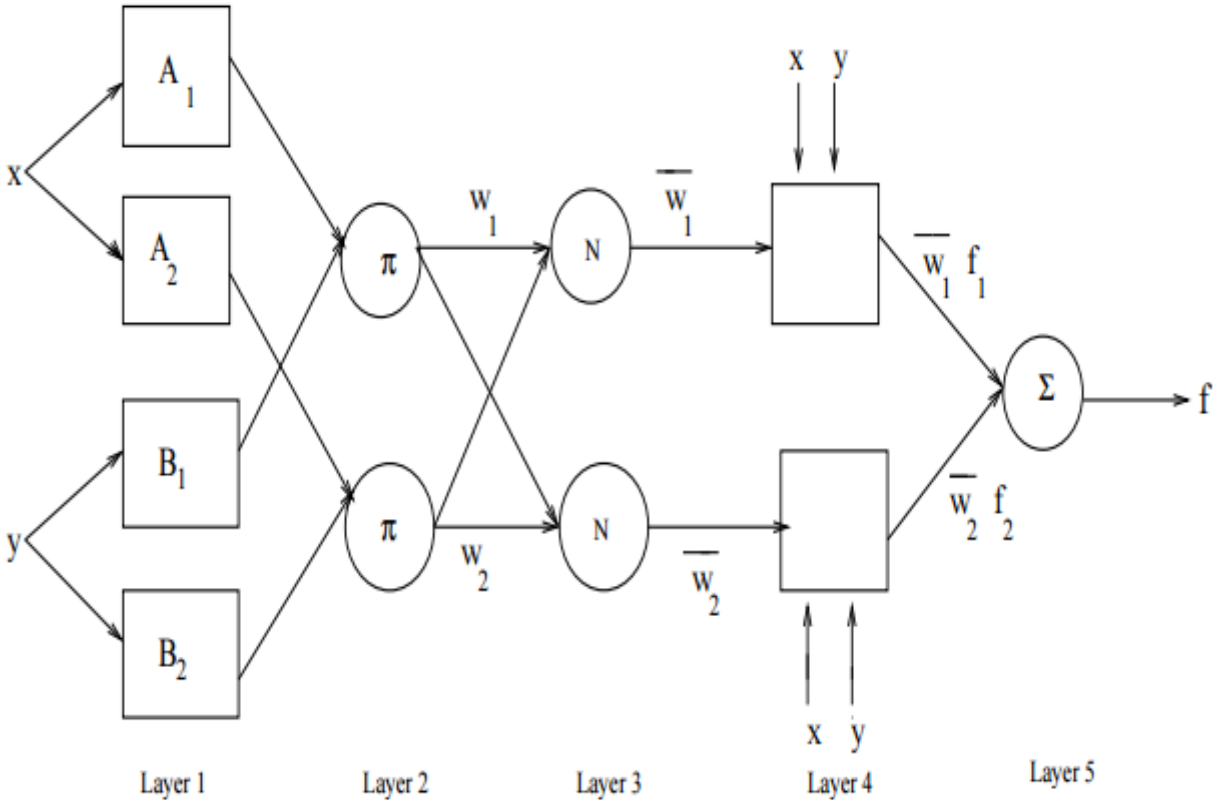


Figure 4.1: Type-3 ANFIS Structure.

The individual layers of this ANFIS structure are described below:

I) Layer 1:

- Every node 'I' in this layer is adaptive with a node function

$$O_i^1 = \mu_{A_i}(x)$$

- Where, x is the input to node i,
  A_i is the linguistic variable associated with this node function and
  μA_i is the membership function of A_i.
- Usually μAi (x) is chosen as:

$$\mu_{A_i}(x) = \frac{1}{1 + [(\frac{x - c_i}{a_i})^2]^{b_i}}$$

Where x is the input and $\{a_i, b_i, c_i\}$ is the premise parameter set.

II) Layer 2:

- Each node in this layer is a fixed node which calculates the firing strength W_i of a rule.
- The output of each node is the product of all the incoming signals to it and is given by:

$$O_i^2 = w_i = \mu_{A_i}(x) \times \mu_{B_i}(y), \quad i = 1, 2$$

III) Layer 3:

- Every node in this layer is a fixed node.
- Each i^{th} node calculates the ratio of the i^{th} rule's firing strength to the sum of firing strengths of all the rules.
- The output from the i^{th} node is the normalized firing strength given by:

$$O_i^3 = \overline{w_i} = \frac{w_i}{w_1 + w_2}, \quad i = 1, 2$$

IV) Layer 4:

- Every node in this layer is an adaptive node with a node function given by:

$$O_i^4 = \overline{w_i} f_i = \overline{w_i}\,(p_i x + q_i y + r_i), \quad i = 1, 2$$

- Where W_i is the output of Layer 3 and {pi , qi , ri} is the consequent parameter set.

V) Layer 5:

- This layer comprises of only one fixed node that calculates the overall output as the summation of all incoming signals, i.e.

$$O_i^5 = overall\ output = \sum_i \overline{w_i} f_i = \frac{\sum_i w_i f_i}{\sum_i w_i}$$

CHAPTER – 5: INTRODUCTION TO OPTIMIZATION TECHNIQUES.

Q1] STEEPEST DESCENT ALGORITHM.

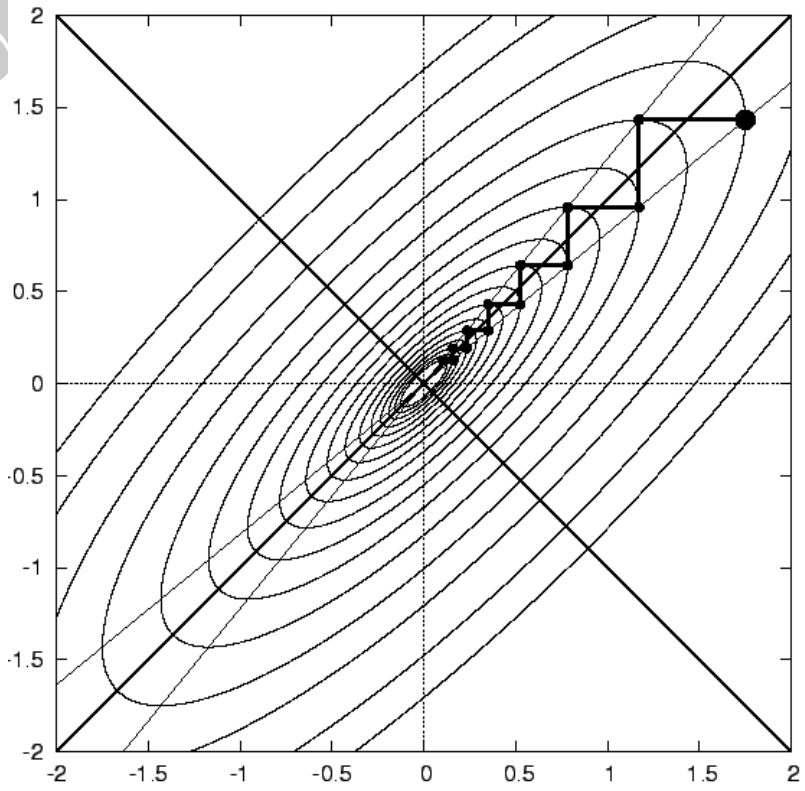
ANS: [10M – DEC15]

1. Steepest Descent Method is the **simplest gradient method**.
2. It is a well-known **iterative minimization method**.
3. It can be applied to any surface for which the gradient may be computed.
4. The method of Steepest Descent computes the gradient at its current location, then travels in the opposite direction of the gradient until it reaches a minimum in this direction.
5. Elementary calculus shows that at this minimum the new gradient is perpendicular to the previous gradient.
6. When applied to a quadratic form 'F' with initial guess X_0.
7. Steepest Descent may be summarized as:

$\mathbf{d}_i = -f'(\mathbf{x}_i) = \mathbf{b} - \mathbf{A}\mathbf{x}_i$ Choose search direction.

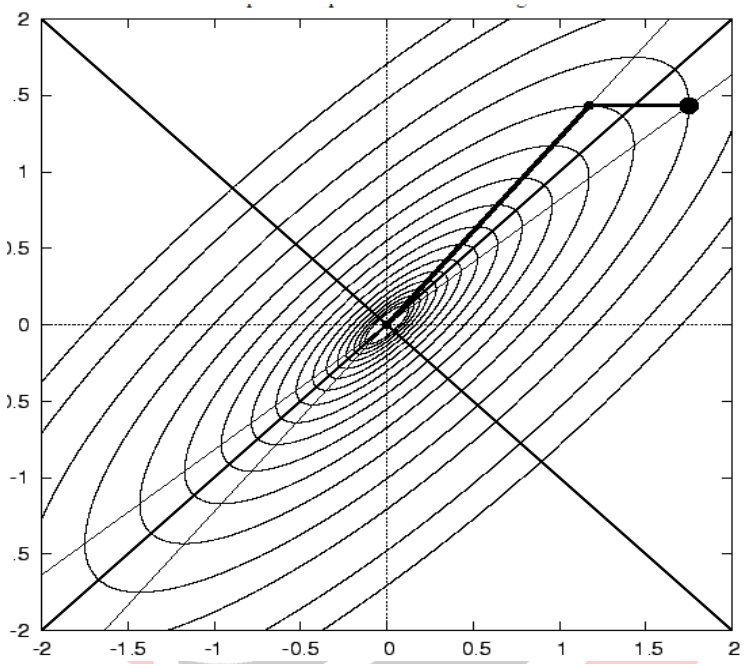
$\alpha = \frac{\mathbf{d}_i^T \mathbf{d}_i}{\mathbf{d}_i^T \mathbf{A}\mathbf{d}_i}$ Compute distance to minimum for line search.

$\mathbf{x}_{i+1} = \mathbf{x} + \alpha \mathbf{d}_i$ Compute next location.



8. The large dot is X_0, and the smaller dots are iterates X_i.
9. Steepest Descent follows $F'(X_0)$ until it finds a perpendicular gradient.

10. Then it repeats the process with the new gradient.



Q2] WHAT ARE THE DIFFERENCES BETWEEN DERIVATIVE FREE AND DERIVATIVE BASED OPTIMIZATION.

ANS: **[5M – DEC15]**

Derivative Free Optimization	Derivative Based Optimization
Derivative Free Optimization cannot be derived.	Derivative Based Optimization can be derived.
It makes use of evolutionary concepts.	It does not makes use of evolutionary concepts.
It is slower than Derivative Based Optimization.	It is faster than Derivative Free Optimization.
It makes use of random number generator to find the search directions.	It does not makes use of random number generator to find the search directions.
No analysis is done sue to randomness.	Analysis is performed at every step.
There is no need of only differentiable function.	There is need of differentiable function.
Some natural wisdom is used that is based on evolution & thermo dynamics.	No natural wisdom is used.
Technique: Simulated Annealing.	**Technique:** Descent Method & Newton's Method.

CHAPTER – 6: GENETIC ALGORITHMS & ITS APPLICATION.

Q1] EXPLAIN GENETIC ALGORITHM WITH THE HELP OF EXAMPLE.

ANS: [10M - MAY16]

GENETIC ALGORITHM:

1. Genetic Algorithms are **Adaptive Heuristic Search Algorithm.**
2. Genetix`c Algorithms are a part of evolutionary computing, which is a rapidly growing area of Artificial Intelligence & Soft Computing.
3. Genetic Algorithms are inspired by Darwin's theory about evolution.
4. It is an intelligent random search technique.
5. It is used to solve optimization problem.

GENETIC OPERATORS:

I) Reproduction:

- It is usually the first operator applied on population.
- Chromosomes are selected from the population of parents to cross over and produce offspring.
- It is based on Darwin's evolution theory of "Survival of the fittest".
- Therefore, this operator is also known as 'Selection Operator'.

II) Cross Over:

- After reproduction phase, population is enriched with better individuals.
- It makes clones of good strings but does not create new ones.
- Cross over operator is applied to the mating pool with a hope that it would create better strings.

III) Mutation:

- After cross over, the strings are subjected to mutation.
- Mutation of a bit involves flipping it, changing 0 to 1 and vice-versa.

GENETIC ALGORITHM PSEUDO CODE:

Generate *an initial population of individuals*
Evaluate *the fitness of all individuals*
While *termination condition not met* ***do***
    Select *fitter individuals for reproduction.*
    Recombine *between individuals*
    Mutate *individuals*
    Evaluate *the fitness of the modified individuals*
    Generate *a new population*
End While

EXAMPLE:

Problem: Find the binary number 11010010

Solution:

I) Initialization:

We start with 5 Random Binary Numbers with 8 Digits each.

01001010

10011011

01100001

10100110

01010011

II) The Fitness Function:

We define the fitness of a number to be the sum of the distances of its digits from these in the target, with the sign minus.

The target: 11010010

Fitness (01001010) = - ((|1 - 0| + |1 - 1| + |0 - 0| + |1 - 0| + |0 - 1| + |0 - 0| + |1 - 1| + |0 - 0|)) = -3

Fitness (10011011) = -3

Fitness (01100001) = -5

Fitness (10100110) = -4

Fitness (01010011) = -2

III) Parent Selection:

In each generation, some constant number of parents, say 4 are chosen. Higher is the fitness, greater is the probability of choosing the individual.

{01001010, 10011011, 01100001, 10100110, 01010011}

{-3, -3, -5, -4, -2}

Assume we select "10011011", "10100110"

Fitness ("10011011") > Fitness ("10100110")

Therefore 10011011 is selected.

Now we get {01001010, 10011011, 10100110, 01010011}

IV) Recombination:

Two selected parents will be coupled with probability 0.5.

("01001010", "10011011"), ("01001010", "10100110"), ("10100110", "01010011")

3 is selected as a random number from 1 to 8

Then we do a Crossover:

From ("01001010", "10011011")

010**01010**

100**11011**

We get ("01011011, 10001010")

We repeat that for each selected couple.

V) Mutation:

For each digits in each of the off springs, we make the bit slip with probability 0.05.

For "10001010" we have "10011010"

Q2] What are the different types of encoding, selection, crossover, mutations of GA. Explain each type with suitable examples.

Ans: **[10M – DEC15]**

Encoding:

1. Encoding of chromosomes is the first question to ask when starting to solve a problem with GA.
2. Encoding depends on the problem heavily.

Types:

I) Binary Encoding:

- Binary Encoding is the most common method of encoding.
- In binary encoding, every chromosome is a string of bits - 0 or 1.

Example:

Chromosome A	**101100101100101011100101**
Chromosome B	**111111100000110000011111**

II) Permutation Encoding:

- Permutation Encoding can be used in ordering problems, such as travelling salesman problem or task ordering problem.
- In permutation encoding, every chromosome is a string of numbers that represent a position in a sequence.

Example:

Chromosome A	**1 5 3 2 6 4 7 9 8**
Chromosome B	**8 5 6 7 2 3 1 4 9**

III) Value Encoding:

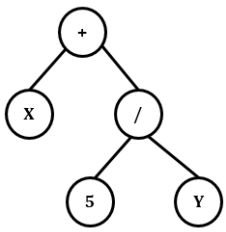
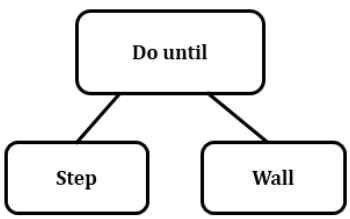
- In the value encoding, every chromosome is a sequence of some values.
- Values can be anything connected to the problem, such as (real) numbers, chars or any objects.

Example:

Chromosome A	**1.2324 5.3243 0.4556 2.3293 2.4545**
Chromosome B	**ABDJEIFJDHDIERJFDLDFLFEGT**
Chromosome C	**(back), (back), (right), (forward), (left)**

IV) Tree Encoding:

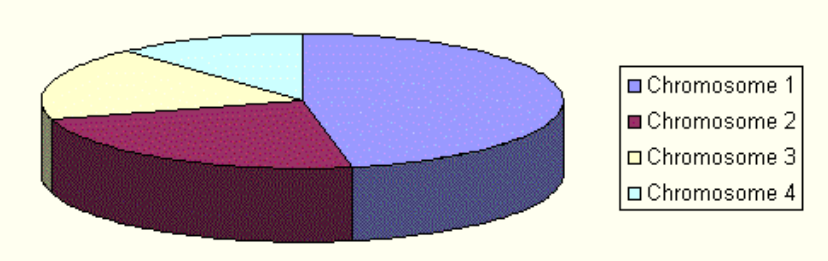
- Tree encoding is used mainly for evolving programs or expressions, i.e. for genetic programming.
- In the tree encoding every chromosome is a tree of some objects, such as functions or commands in programming language.

Chromosome A	Chromosome B
	
(+ x (/ 5 y))	**(do_until step wall)**

SELECTION:

I) Roulette Wheel Selection:

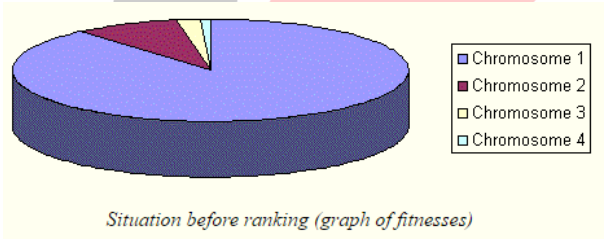
- Parents are selected according to their fitness.
- The better the chromosomes are, the more chances to be selected they have.
- Imagine a roulette wheel where are placed all chromosomes in the population as shown below.



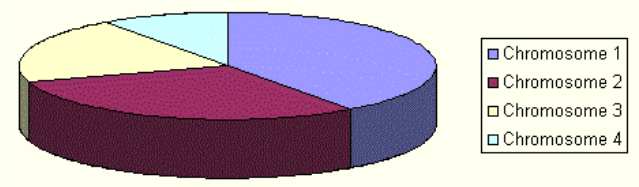
- Then a marble is thrown there and selects the chromosome. Chromosome with bigger fitness will be selected more times.

II) Rank Selection:

- The previous type of selection will have problems when they are big differences between the fitness values.
- For example, if the best chromosome fitness is 90% of the sum of all finesses then the other chromosomes will have very few chances to be selected.
- Rank selection ranks the population first and then every chromosome receives fitness value determined by this ranking.
- The worst will have the fitness 1, the second worst 2 etc. and the best will have fitness N (number of chromosomes in population).
- See in following diagram, how the situation changes after changing fitness to the numbers determined by the ranking.



Situation before ranking (graph of fitnesses)



Situation after ranking (graph of order numbers)

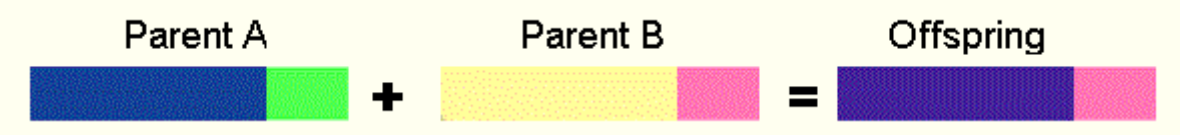
CROSSOVER & MUTATION:

Crossover is one of the most important part of genetic algorithm.

Types:

I) Single Point Crossover:

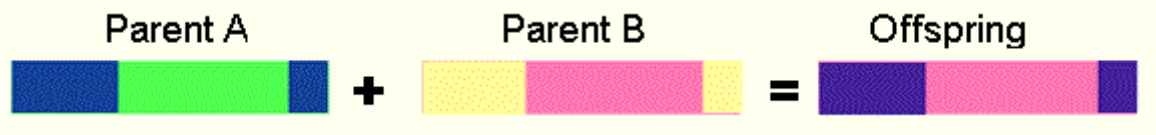
One crossover point is selected, binary string from beginning of chromosome to the crossover point is copied from one parent, and the rest is copied from the second parent.



Example: 11001011 + 11011111 = 11001111

II) Two Point Crossover:

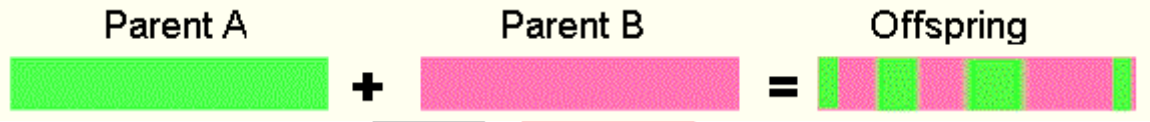
Two crossover point are selected, binary string from beginning of chromosome to the first crossover point is copied from one parent, the part from the first to the second crossover point is copied from the second parent and the rest is copied from the first parent.



Example: 11001011 + 11011111 = 11011111

III) Uniform Crossover:

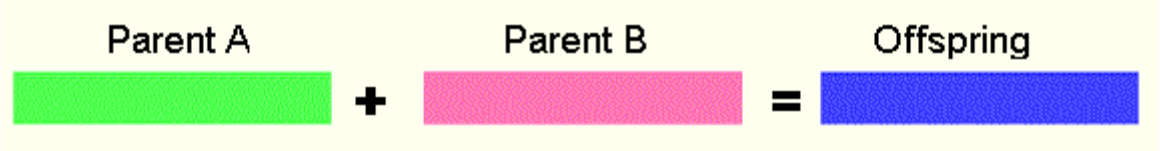
Bits are randomly copied from the first or from the second parent



Example: 11001011 + 11011111 = 11011111

IV) Matrix Crossover:

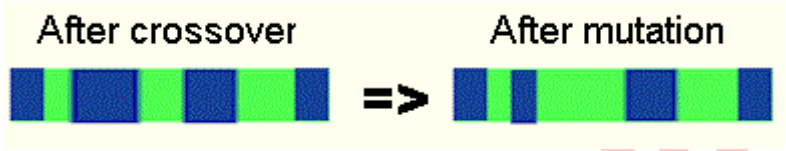
Some arithmetic operation is performed to make a new offspring



Example: 11001011 + 11011111 = 11001001 (AND)

MUTATION

Bit inversion: Selected bits are inverted.



Example: 11001011 → 10001011