Lakshmi Narain College of Technology, Indore

CONTENTS

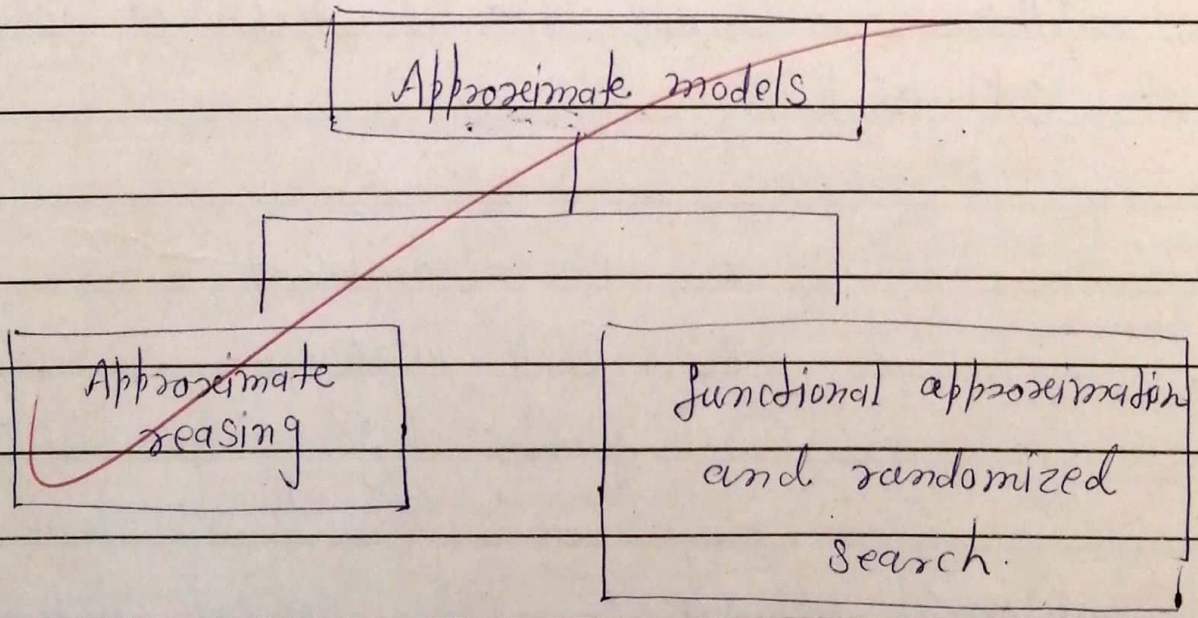
S. No.	Date of Experiment	Name of Experiment	Page No.	Date of Submission	Remarks
1	7/02/17	what is soft computing and various technique and also diffn b/w soft computing and artifical intelligent	1 to 6	14/02/17	
2	14/02/17	To Study BFS and DFS and also compare this	7 to 10	28/02/17	
3	28/02/17	To Study Hill-climbing algo with all its types	10-16	7/03/17	
4	7/3/17	A Study A* Algo	17-21	14/03/17	
5	14/03/17	To Study Ao* Algo	22-25	21/03/17	
6	21/03/17	Study neural network and also its types comparison b/w artifical and biological neural N/w	26-29	21/03/17	
7	21/03/17	To Study counter propogation network counter propogation also with its architecture and characteristic	30-33	18/04/17	
8	18/04/17	To Study fuzzy logic. and crisp set also compare with them and crisp	34-39	18/04/17	

CONTENTS

Experiment-1

What is Soft computing and various technique and also differentiate b/w soft computing and Artifical intelligence.

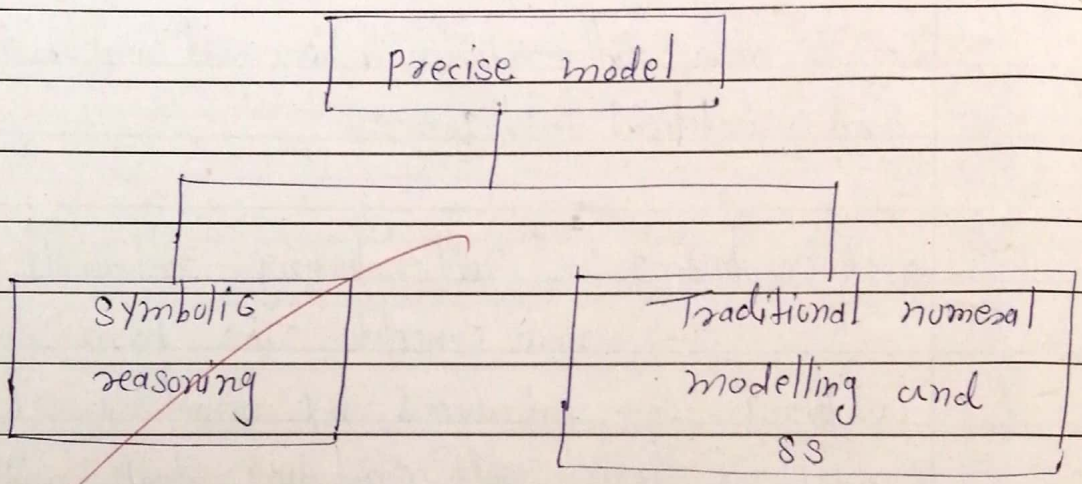
Soft computing :- Soft computing relatively a new concept. The term soft computing was introduced by professor lotfi-Zadeh in 1994. Soft computing deals with approximate models and gives soln to complex problem are as shown in figure



Two major problem-solving technology include

1. Soft computing
2. Hard computing

The ultimate goal is to be able to emulate the human mind as closely as possible soft computing involve partnership of several field. it is the combination of GA's, neural network and FL.



Various technique of soft computing :-

There are many type of technique of the soft computing.

1. neural network
2. fuzzy logic
3. genetic Algo.

Neural network :- The neural network is a processing device or an actual hardware. whose design and

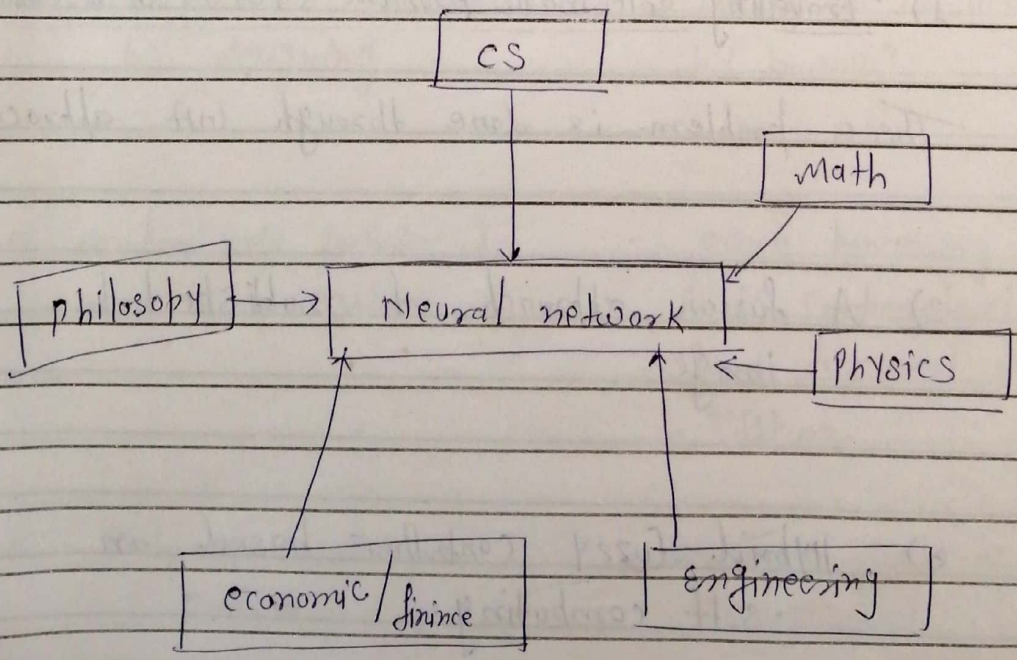
and functioning inspired by animal brain. it is also called a artifical neural network or neural net the neural network have the ability to learn by.

e.g.

which make them very flexible and powerfull These network are also well suited for real time system. because of their fast response and parallel architecture.

Advantage :-

- ✓ Adaptive learning
- ✓ soft organizing
- ✓ Real time operation
- ✓ fault tolerance



fuzzy logic:- it is developed by Bellman and Zadeh in (1970). fuzzy logic allow for manipulation of fuzzy system and membership functions

various concept of fuzzy logic

a) law associated with classical set

b) Premetive operation

c) law associated.

Soft computing (Application):-

There are many application of soft computing:-

1) Travelling salesman problem (TSP) optimization

These problem is done through GA approach

2) A fusion approach of multispectral image

3) Hybrid fuzzy controller based on soft computing.

Difference b/w soft computing and AI :-

	Artifical intelligence	Soft computing
(i)	Knowledge is represented at higher level, that is explicit knowledge or abstract knowledge	Knowledge is represented by numeric forms in terms of weight which have no relationship of weight
(ii)	it can explicitly correct error by remodified the facts and pulses	it cannot explicitly correct the error the network by itself modify their weight to produce the current output intelligence is obtained by training.
3.	intelligence is obtained by designing	intelligence is obtained by training
4.	comparatively inferior to real time system.	since processing is fast, comparatively good for real time system.
5.	Response time is consistent	Response time is inconsistent

6).	Symbolic representation is used	numeric representation is used
7).	processing speed is low	speed is fast due to its parallel processing and dedicated hardware
8)	sequential processing is used	distributed processing is used.

Experiment-2

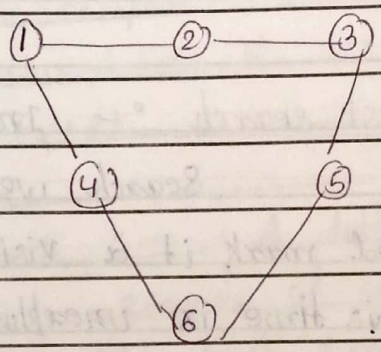
Aim :- To Study BFS and DFS and also Compare this

Breadth first search :- In breadth first Search we start at vertex v and mark it a visited. The vertex v is at this time is unexplored, since a vertex v is at this time is unexplored, since when all of its adjecnt are visited. in BFS all unvistied vertices adjecnt from v are visited next. These are new unexplored Vertices, exploration continues untill no unexplored Vertex is left. The list of unexplored Vertices are sorted in a queue.

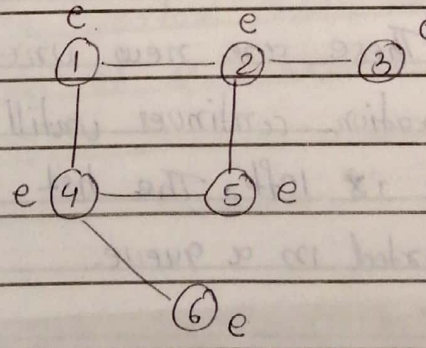
Algorithm :-

```
BFS (v)
1.  u := v
2.  visited [v] := 1
```

3. repeat
4. for all vertices w adjacent to v
5. if (visited [w] = 0)
6. Add w to Φ // w is unexplored
7. visited [w] := 1
8. if Φ is empty
9. Return // No unexplored vertex
10. delete v from Φ
11. untill (flase)



Sample undirected graph



Comments :- A breadth first search of G is out beginning at vertex v 'G' is assumed to be Global, that is why

it is not passed as an argument in BFS() for any node 'i', visited[i]=1, if i has already been visited, else visited[i]=0 (initialised with '0')

visited[] is also global

Q is queue to hold unexplored vertices

Complexity of Algo :- here we assume that graph G is represented as adjacency list The loop at (3) and (4) will take together O(V+E) time since visiting adjency list or whole will take O(2E) time, since a node in adjency list repeated twice and checking for every vertex whether visited or will take constant time i.e O(1).

So compexity of BFS, when adjacency list representation is used is O(1) + O(V) + O(2E) = O(V+E)

Depth first search :- A depth first search of a graph differ from a breadth first search in that the exploration of a vertex v is suspended as soon as a new vertex u begins. when this new vertex has been explored the exploration of v continues

In depth first search not all adjcent of a vertex is visited, but search goes in depth first and then backtype to traverse remaining adjcent of vertex v.

Algo :-

DFS (v)

1. visited [v] := 1
2. for each vertex to adjcent from v
3. if visted [w] = 0) DFS (w)

Comment :- Given an undirected (or directed) Graph G = (v, E) with n vertices and an array visted [] initally set to zero. This alg. visted all vertex reachable from v. G and visited [] are global

Complexity :- since DFS visits each node at once only. so loop at 2. Recursive all at 3. will take at whole o(V+E) time. in adjency list representation of graph.

Experiment-3

Aim :- To Study Hill-climbing algorithm with all its types.

Hill climbing Algorithm :- Hill climbing is an optimization algo that belongs to the family of local search. it is an iterative algo that start with an arbitary soln to a problem. then attempt to find a better soln by increase— mentally changing a single element of the soln. if the charge is found to be better than the previous soln then, an incremental change is made to the new soln, this is reapted untill no further improvements can be found Hill climbing is good at finding a local optimum but it is not findind granted to find the best possible soln out of all possible soln.

In general the advantage of hill-climbing search is that it requires memory only for the current state to be stored, thus it is easy to implement efficiency. Furthermore if one or more soln exist in the search space, hill climbing will be effective in finding it.

The important drawback of local search is its inability to detect the unsolvability of problem it means that if no soln exist, a local search method will simply continue to make and modify and loop forever. In hill climbing we use a heuristic fun^n whose value signifies the amount of distance the node is form the goal.

Hill-climbing algo :-

1. pick inital state s, and evaluate value of f(s)

2. take x in eighbours (s), which has the largest f(x)

3. if f(x) ≤ f(s), then step and return s

4 s=x Goto step 2

The above algo is a very greedy algo and easily stuck.

Type of hill climbing algo :-

→ in simple hill climbing, the first closer node chosen, whereas in steepest ascent hill climbing all successor are compared and the closest to the soln is chosen. both forms fail if there is no closer maxima in the search space which are not soln steepest ascent hill climbing is similar to best-first search, which tries all possible extensions of the current path instead of only one.

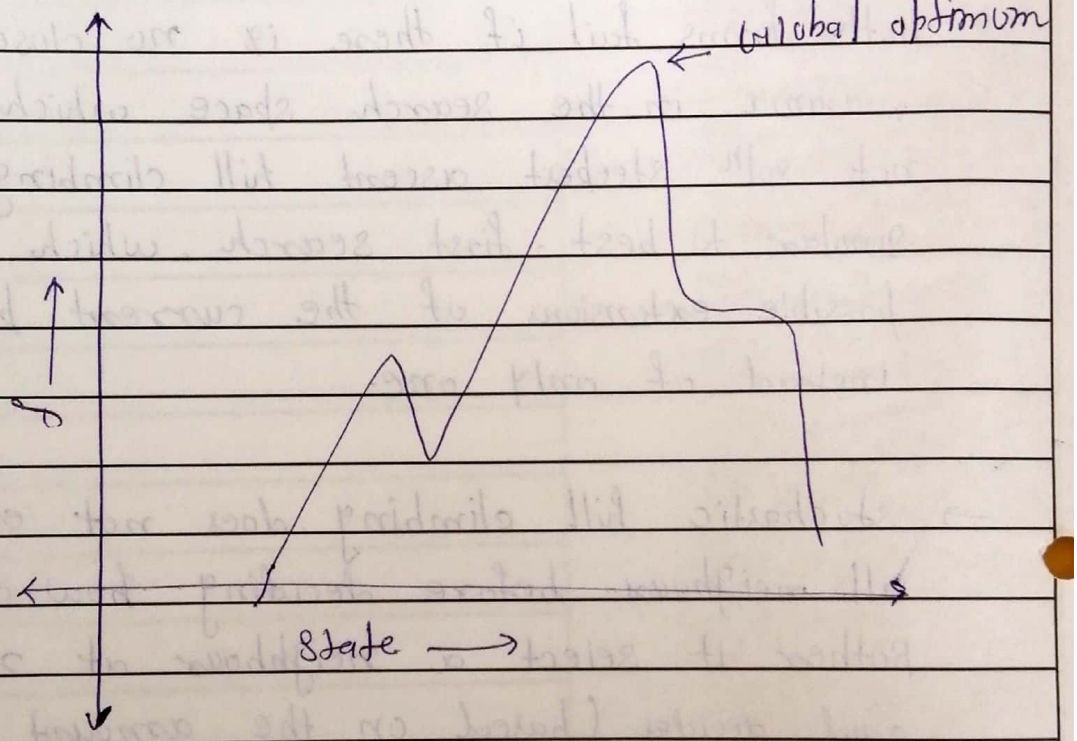
→ stochastic hill climbing does not examine all neighbour before deciding how to move Rather it select a neighbour at random and decides (based on the amount of improvement in that neighbour) whether to move to that neighbour or to examine another.

→ Random - restart hill climbing is a meta-algorithm built on top of the hill climbing algo. it is also klas shotgun hill climbing it iteratively does hill climbing, each time with a random inital condn.

Drawbacks of hill climbing algo:-

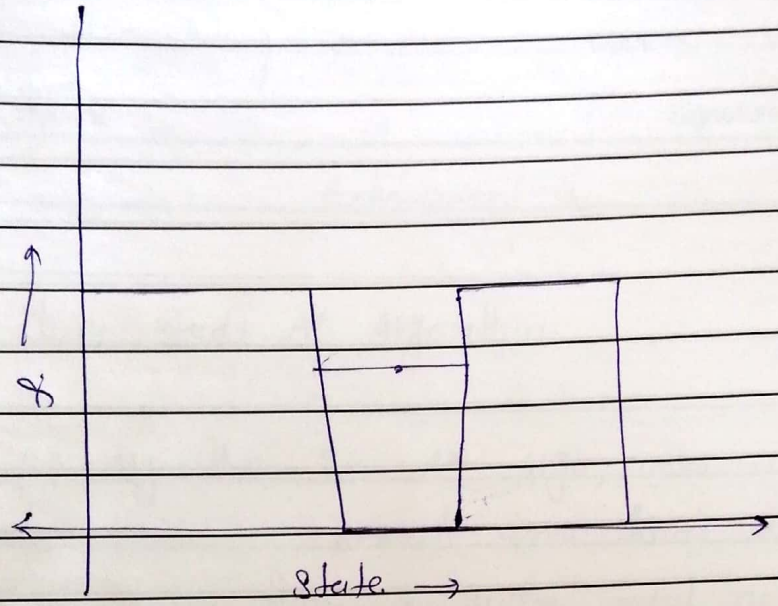
local optima in hill climbing

→ The global optimum is the point where we want to reach as it corresponds to soln state.



2 plateau

→ in case of plateau the search space become flat or sufficiency flat that the value returned by the target fun^n is indistinguishable from the value returned by nearby regions.



Experiment-4

Aim study A* Algorithm.

A* Algorithm :- A* Algo uses a best-first search and finds a least cost path from a given inital node to one goal node.

It uses huristic fun^n to determine the order in which the search visit node in the bee. The huristic is a sum of two fun^n : (1) the path-cost fun^n, which is the cost from the starting node to the current node (usually denoted by g(x)), and (2) and admissible "hurisdic estimate" of the distance to the goal (usually denoted h(x)). The h(x) part of the f(x) fun^n must be an admissible hurisdic; that is it must not oversites the distance to the goal.

A* Algo traverse the graph, it follows a path of the lowest known cost, keeping a sorted priority queue of alternate path segment along the way. if at any point, a segment of the path being traversed has a higher cost than another encountered path segment, it abandons the higher cost path segment instead. This process continues until the goal is reached.
for this type of problem we need to maintain two lists types:

(i) open :- nodes on the open list are node that have been generated and hold the huristic function applied to them but which have not been expanded.

(ii) closed :- Nodes on the closed list are nodes that have been expanded and whose children are available to the search program.

Algorithm :-

(i) start with open containing only the initial node set the node's g value to 0, its h value to whatever it is and its f' value to h' + 0, or set

Closed to the empty list

(ii) until a goal node is found, repeat the follower procedure - if there are no nodes on OPEN, repeat failure otherwise, pick the node and OPEN with the lowest f' value, call it BESTNODE, Remove it from OPEN place it on CLOSED. see if BESTNODE is a goal node. if so, exit and report a soln (either BESTNODE, if all we want is the node or the path that has been created b/w the inital state and the BESTNODE if we are instrested in the path). otherwise, generate the successor of BESTNODE but do not set BESTNODE to point to them yet. (first we need to see if any of them have already been generated) for each such SUCCESSOR do the following

a) set SUCCESSOR to point back to BESTNODE These backward links will make it possible to recover the path once a soln is found.

b) compute g(SUCCESSOR) = g(BESTNODE) + the cost of getting from BESTNODE to SUSSESSOR

c) See if SUCCESSOR is the same

as any node on open (i.e it has already been generated but not processed) if so, call that node OLD, since, this node already exist in the graph, we can throw SUCCESSOR away and add old to the list of BESTNODE'S SUSSCSSOR.

Now we must decide wether old's parent link should be if the path we have the current best path to old.

So see wether it is cheaper to get to OLD via its current parent or to SUCCESSOR via BEST-NODE by comparing their g value. if OLD is cheaper (or just as cheap). then nothing.

if successor is cheaper, then reset OLD's parent link to point BESTNODE, record the new cheaper path in g(OLD) and update f'(old)

d). If SUCCESSOR was not OPEN, see if it is on CLOSED if so, call the node on CLOSED OLD and add OLD to the list of BESTNODE'S SUCCESSORS. check to see if the list of BESTNODE'S successor. check to see if the new path or the old path is better just as in step ii (c)

and set the parent link and g and value appropriately. if we have just found a better path to OLD's we must propogate the improvement to OLD's successor's

e) if successor was not already on either open or closed, then put it on OPEN and add it to the list of BESTNODE's successor. compute $f'(successor) :=$

$$g(successor) + h'(successor).$$

Experiment - 5

Aim :- To study Ao* algo

Ao* Algorithm :- The Ao* algo uses a single structure Graph representing the part of the search graph that has been explicitely generated so far. each node in the graph will point both down to its immediate successor and upto its immediate predecessor each node in the graph will also have associated with it an 'h' value, an estimate of the cost of a path from itself to a goal node also h will serve as the estimate of the cost of a path from itself to a set of goal nodes. also, h' will serve as the estimate of goodness of node.

Algorithm :-

(i) let GRAPH consist only of the node representing the inital state. (call this node INIT) compute h' (INIT)

(ii) Untill INIT is labeled solved or until INIT & h' value become greater than fUTILITY, repeat the following procedure.

(a) Trace the labeled arcs from INIT and select for expansion one of the as yet unexpanded nodes that occour on this path. call the seleted node NODE.

b) generate the successor of NODE. if there are none then assign fUTILITY as the h value of NODE. This equivalent to saying that NODE is not solvable. if there are successor, then for each one (called successor) that is not also an acentor of node do the follow

1) Add successor to Graph

2) if successor is a terminal node, label it solved and assign it an h value 0

3) if successor is not terminal node

compare its value.

c) propogate the newly discovered information up to graph by doing the following - let S be a set of node that have been labeled SOLVED or whose h values have been changed and so need to have value propogated back to their parents. initilize S to NODE until S is empty. repeat the following procedure

(1) if possible, select from S node none of whose desendants in Graph occours in S. if there is no such node, select any node from S. call this node current, and remove it from S.

(2) Compute the cost of each of the arcs emerging from CURRENT. The cost of each arc is equal to the sum of the h' values of each of the node at the end of the arc plus whatever the cost of the arcs itself is assign as current new h values the minimum of the costs Just compute for the arcs emerging from it.

(3) Mark the best path out of current by marking the arc that had the minimum

cost as computed in the previous step.

4) mark current solved if all of the node connected to it through the new labled arc have been labeled SOLVED.

5) if CURRENT has been labeled solved or if the cost of CURRENT was just chand, then its new status must be propogate back up the graph. so add all of the ancestor of current to S.

AO* Search procedure :-

- place the start node on open

- using the search tee, compute the most promising soln tee TP.

3. select n that is both on open and a part of tp, remove n from open and place it no closed.

4. if n is a goal node, label n as solved. if the start node is solved with success where tp is the soln tee, remove all nodes from open with a solved ancestor.

Experiment - 6

Aim :- Study Neural network and also its types. Compassion b/w artifical neural network and Biological neural network.

Neural network :- neural network are simplified models of the biological nervous system and therefore have drawn their motivation from the kind of computation performed by a human brain. A neural network, in general, is highly interconnected network of a large no. of processing element called neurons is an architecture motivated by the brain. A neural network can be massively parallel and therefore is said to be exhibit parallel distributed processing.

Neural network can be taught to perform complese tasks and do not require programming as conventional computers. They are massively

parrell, extremely fast and inherically fault tolerence They require significantely less development time and can respond to situations unspecified or not previously envisaged.

In simple terms, a neural network is made up of no. of processing element called neurons whose interconnections are called synapses each neurons accepts inputs from either the external world or from the output of other neurons. output signals from all neurons eventually propogate their effect accross the entire network to the final layer where the result can be output to the real world. The synapses have a processing value or weight which is learnt during training of the network. The functionality and power of the network primarily depend on the no. of neurons in the network, the interconnection or topology, and the value of the weight assigned to each synapse.

There are two types of neural network:

1. Artifical neural network
2. Biological neural network

Comparison blw artifical and biological neural network

Artifical neural network	Biological neural network
1. Neural network are faster in processing information	Biological neurons are slow in processing information
2) Many programs have large no. of instructions and they operate in a seqential mode one instruction after another on a conventional computer	Biological neural network can perform massively parallel operation.
3) Artifical nets are inherently not fault tolerant	They exhibit fault tolerant since the information is distributed in the connetn throught the network
4. There is a conbal unit, which monitor all the activities of computing	There is no conbol unit for processing information in the brain
5. These do not involve as much computational neurons.	Neural n/w have large no. of computing element and the computing is not restricted to within neurons.

Experiment - 1

Aim :- To study counter propogation network als with its architecture and characteristics.

Counter Propogation network :- Counter propogation is diff^n from back propogation and provide sol^n for those problem that cannot longer operate in long paining session.

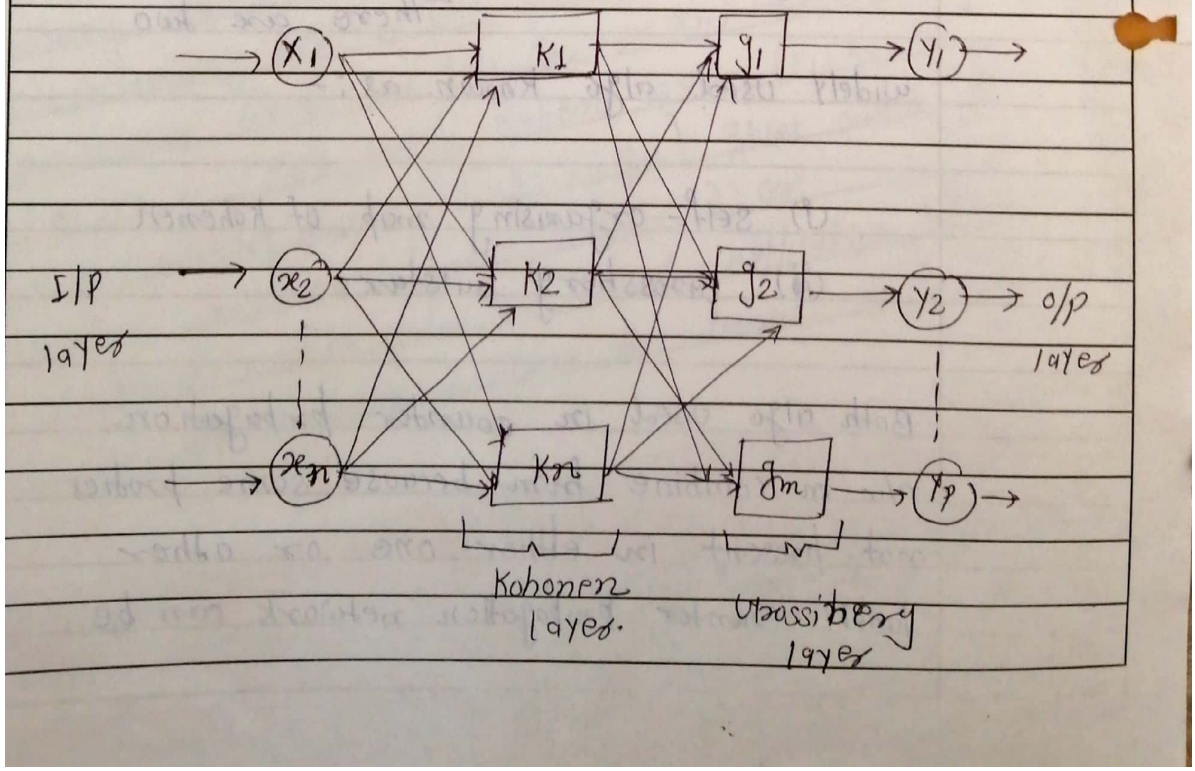
There are two widely used algo known as :-

(i) self - organising map of Kohenen
(ii) Grossberg outstar.

Both algo used in counter propagation n/w in combine from because some proties not present in either one or other lonely counter propogation network can be

trained to counter perform associative mapping much faster than typical two layer network. it can also be used for prototyping of a mapping and speed up system development because they require orders of magnitude fewer training cycles than is needed error back propagation network

Architecture :- Counter propogation network architecture differ in two layers from simple neural network's hidden layer counter propagation network uses Kohonen layer and Grossberg layers as hidden layer, and input layer, output layer like as neural network so, counter propogation network contain four layer in feed forward version architecture. Architecture of CPN (counter propogation network)



in fig

→ $x_1, x_2 \dots, x_n$ are input
→ $K_1, K_2 \dots, K_n$ are Kohen neuron
→ $g_1, g_2 \dots, g_n$ are grossing neurons
→ $Y_1, Y_2, \dots Y_p$ are desired o/p

Counter propogation network works in two modeds:

(a) Normal mode – in which it accepts input and given.

output vector $(Y_1 \dots Y_p)$

(b) Training mode – in which input vector $(x_1 \dots x_n)$ is applied and the weight are adjusted to obtian desied o/p

Functionality of counter propogation network

→ counter propagation network fun^n based on look up table capble of generalization. The training process associates input vector with corresponding o/p vector as input vector to knonon layer as $(x_1, x_2 \dots x_n)$ and $(K_1, K_2 \dots K_n)$ and Kohen layer $(K_1, K_2 \dots K_n)$ the Grossberg layer $(g_1, g_2, \dots g_m)$ gives output vector may be binary (0 or 1) or continous. when n/w is trained, application of an input vector produces the desired output

layers

→ counter propogation network is a hybrid network in nature.

Application :-

→ modelling
→ Artifical neural network (ANN)
→ Star identification
→ Steal girder bridge.

Main used of counter propogation network in such as following area given as following

(i) Counter propogation network used in artifical neural network using middle layer

(ii) it also used in fiuld of modelling to get desired of model system

(iii) it application in neural network for modelling

(iv) Counter propogation network also used for star identification

(v) This network also used for solving power flow problem.

Experiment 8

Aim :- To study fuzzy logic and crisp set also compare with term and crisp relation and fuzzy relation

Fuzzy logic set :- In case of fuzzy logic, truth values are multivalued, i.e. partly true, very true, absolutly true, abasulty false, partly false etc are numerically equivalent to 0-1

fuzzy propositions :- A fuzzy proposition is a statement which hold a fuzzy truth value of R be a fuzzy proposition then T(R) will represent the truth value (0-1) attached to R in general fuzzy proposition is associated with fuzzy set

So, $T(R) = \mu_A(x)$ where $0 \leq \mu_A(x) \leq 1$

Fuzzy operators :-

following operators :-

(i) Disjunction (∨)

(ii) conjunction (∧)

(iii) Negation (−)

(iv) implication (⇒)

Details of fuzzy logic operators are given below

Symbol	operator	Usage	Details
∨	Disjunction	$\underset{\sim}{P} \vee \underset{\sim}{R}$	$\max(T(\underset{\sim}{P}), T(\underset{\sim}{R}))$
∧	Conjunction	$\underset{\sim}{P} \wedge \underset{\sim}{R}$	$\min(T(\underset{\sim}{P}), T(\underset{\sim}{R}))$
−	Negation	$\overline{\underset{\sim}{R}}$	$1 - T(\underset{\sim}{R})$
⇒	Implication	$\underset{\sim}{P} \Rightarrow \underset{\sim}{R}$	$\max(1-T(\underset{\sim}{P}), T(\underset{\sim}{R}))$

In case of fuzzy logic "⇒" operator represent the IF-THEN statement as :

if x is $\underset{\sim}{A}$ Then y is $\underset{\sim}{B}$ and equivalent to

$$R = (\underset{\sim}{A} \times \underset{\sim}{B}) \cup (\overline{\underset{\sim}{A}} \times Y)$$

The membership fun^n of $\underset{\sim}{R}$ is given by

$$\mu_R(x, y) = \max(\min(\mu_{\underset{\sim}{A}}(x), \mu_B(y)), 1 - \mu_{\underset{\sim}{A}}(x))$$

for the compound implication If x is $\underset{\sim}{A}$ Then y is B else y is $\underset{\sim}{C}$ the relation $\underset{\sim}{R}$ is equivalent to

$\underset{\sim}{R} = (\underset{\sim}{A} \times \underset{\sim}{B}) \cup (\overline{\underset{\sim}{A}} \times \underset{\sim}{C})$

The membership fun^{n} of $\underset{\sim}{R}$ is given as,
$\mu_{\underset{\sim}{R}}(x,y) = \max(\min(\mu_{\underset{\sim}{A}}(x), \mu_{\underset{\sim}{B}}(y)), \min(1 - \mu_{\underset{\sim}{A}}(x), \mu_{\underset{\sim}{C}}(x)))$

CRISP Logic set :- If we take the statement "SKY" is "Red" and "SKY" is "Blue" then argeement or disagement with the BC statement is indicated "True" or "false" value to the statement is it first statement take "True" and second statement take "false" value then we can say that either statement is true or false but can say that either statement is true or false but can not both at a time is called proposition indicated by Upper case letter such as P, Q, R etc.

There are five major operator are given below for propositional logic :-

Symbol	operator	usage	details
$\wedge$	and	$P \wedge Q$	P and Q are True
$\vee$	or	$P \vee Q$	either P or Q are true
$\neg$ or $\sim$	not	$\sim P$ or $\neg Q$	P is not true
$\Rightarrow$	implication	$P \Rightarrow Q$	P implies Q is true
$=$	equality	$P = Q$	P & Q are equal is true

where :-

P = SKY is Red

Q = SKY is blue are proposition.

laws of propositional logic

If P, Q, R are proposition then propositional logic supports following :-

(a) Associatively :-

$(P \vee Q) \vee R = P \vee (Q \vee R)$

$(P \wedge Q) \wedge R = P \wedge (Q \wedge R)$

(b) commutatively :-

$(P \vee Q) = (Q \vee P)$

$(P \wedge R) = (R \wedge P)$

(c) identity :-

$P \vee false = P$

$R \wedge True = R$

$R \wedge false = false$

$P \wedge True = True$

(d) Distributively :-

$(P \vee Q) \wedge R = (P \wedge R) \vee (Q \wedge R)$

$(P \wedge Q) \vee R = (P \vee R) \wedge (Q \vee R)$

(e) idempotence :-

$P \vee P = P$

$R \wedge R = R$

(f) Absorption :-

$R \wedge (R \vee \phi) = R$

$R \vee (R \wedge \phi) = R$

(g) Negation :-

$R \wedge \neg R = \text{false}$

$R \vee \neg R = \text{True}$

(h) involution :-

$\neg(\neg R) = R$

(i) Demorgans law :-

$\neg(R \vee \phi) = (\neg R \wedge \neg \phi)$

$\neg(R \wedge \phi) = (\neg R \vee \neg \phi)$

CRISP Relations :- The concept of relations b/w sets is a built on the cartison product operator of sets.

Cartesion product :- The Cartesion product of two set A & B denoted by $A \times B$ is the set of all ordered pairs such that the first element in the pair

ie $A \times B = \{(a,b) \mid a \in A, b \in B\}$

if $A \neq B$ and A & B are non-empty then

$A \times B \neq B \times A$

observe that

$$\left| \bigtimes_{i=1}^{n} A_i \right| = \prod_{i=1}^{n} |A_i|$$

$$\bigtimes_{i=1}^{n} A_i = \{ (a_1, a_2 \cdots a_n) \mid a_i \in A_i \text{ for every } i = 1, 2, \cdots n \}$$

Experiment - 9

Aim :- To study Genetic Algo with fundamental concept and working principal also give its comparsion with traditional method.

Genetic Algo :- Genetic Algo GA's were invented by Jhon Holland in the 1960's and were developed at University of michigan in 1960's and 1970's

Gentic algo are search and optimization technique based on Darwin's principal of natural selection. GA's are adaptive heurtic search algo based on the evolutionary ided of natural selection and gentic. GA represent an intelligent exploitation of a random search used to solve optimization problem. The basic tehnique of the GA's are deigned to simulate process in natural system necessary for evolution, specially those follow the

principal first said down by charles Darwin of survival of the fittest"

GAs are :-

1. Directed search algo based on the mechanics of biological evolution

2. Provide efficient, effective technique for optimization and machine learning application

According to John Holland, university of michigan (1970's) GA is used.

1. To understand the adaptive process of natural system.

2. To design artifical system software that retains the robustness of natural system

Working principal :-

To illustrate the working principal of GAs an unconstrained optimization problem is considered.

let us consider following maximization problem

maximize $f(x)$, $x_i^l \leq x_i \leq x_i^u$,

$i = 1, 2, \ldots, N$.

where x_i^l & x_i^u are the lower and upper bound the variable x_i can take. Although a maximization problem.

minimize $f(x)$, for $f(x) > 0$, then we can write the obj fun^n as

maximize $1/f((x)+1)$

Difference & similarties b/w GAs and other Tradiitional method.

Difference b/w GA's Algo and other tradiitional method are :-

(i) GAs work with a string of coding of variables insted of the variable.

(ii) GA also work on discrete search space even though the fun^n may be continous.

(iii) GA works with a population of point instad of a single point

(iv) GA is population based search algo. and multiple optimal sol^n can be obtained.

Some similarties b/w GA and traditional method are.

(i) cross over operator is similar to directional/ traditional search method with an exception that search direction is not fixed for all point

(ii) The mutation operator is used to create a point in the vicinity of current point.

Aim :- To study natural language processing also compare with monotonic and non-monotonic.

Natural language processing :- NLP is a computer based approach of analysing that is based on set of technologies and set of theories.

NLP is a computational technique for analysing and representing naturally occuring texts at one or more levels of linguistic analysis for the purpose of achiving human like language processing for a variety of application.

The NLP it is useful to partion entire language processing problem into two tasks.

a) processing of written text, using Syntatic, Semantic and lexical knowledge of language.

b) Processing spoken language using all the information needed above plus some additional knowledge.

The full NLP System should be able to:-

1. Paraphase an input text
2. Translate the text into another language
3. Draws inference from the text
4. Answering q^n about the contents in the text.

Levels of NLP:-

1. Phonology :- This level deals with the interpredetion of speech-sounds within and across. There are in fact three types of rules in this type of analysis.

a) Phonetic rules

b) Phonemic rules :-

c) Prosodic rules :-

2. Morphology :- This level deals with nature of words that is componedial, they are composed of morpheme that is the smallest units of meaning.

3. Lexical :- meaning of individual words are interpreted by human as well NLP system

4. syntatic :- it is a sequence of word that show how the word relate to each other.

Application of NLP :-

1. information Retrival
2. Information extraction
3. Question Answering
4. Machine Translation
5. Summarization

Monotonic Reasioning :- Monotonic logic is a logic in which truth of the proposition does not change when new information are added. Traditional logic is logic are valued deduction and they remain forever.

logic based system are monotonic in nature

i.e if a proposition is made, whose is truth value.

Advantage :-

(i) In monotonic reasioning new axioms are asserted new used formed formula became provable, but the old proof never become invalid.

(ii) it gives valied deduction that remain so forever

Disadvantage :-

(i) Due to limited expressive power real world situation cannot be expressed properly

(ii) it cannot express thing that are uncertain.

Non-monotonic Reasoning :-

In non-monotonic type of reasoning type of reasoning the axioms or the rule of inferences are extended to make it possible to reason with incomplete information. however, this system has the property that at any instant of time a statement is either assumed or believed to be true or believed to be false, or not believed to be either.