of parents.

ⓔ [Mutation] With a mutation probability, mutate new offspring at each locus (position in chromosome).

ⓕ [Accepting] Place new offspring in the new population.

4. [Replace] Use new generated population for a further run of the algorithm.

5. [Test] If the end Condition is Satisfied, stop and return the best solution in Current population.

6. [Loop] Go to step 2.

<u>Note.</u> The GA performance is largely influenced by two operators Called Crossover and mutation. These two operators are the most important parts of GA

* <u>Encoding</u>: Before a genetic algorithm Can be put to work on any problem, a method is needed to encode potential solutions to that problem in a form so that a Computer Can process.

- One Common approach is to encode solutions as binary strings: sequences of 1's and 0's, where the digit at each position represents the value of some aspect of the solution.

<u>Example</u>: A Gene represents Some data (eye Colour, hair Colour, Sight, etc)

A Charomosome. is an array of genes. In binary form.

a Gene looks like (11100010)

a chromosome looks like

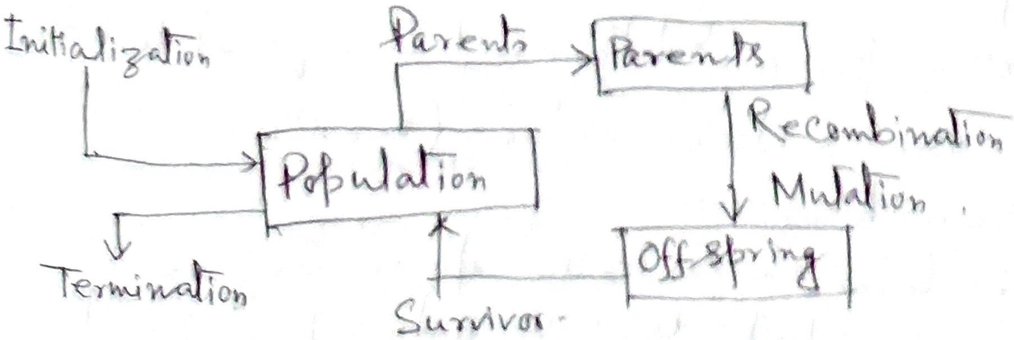
($\overset{G-1}{11000010}$, $\overset{G-2}{00001110}$, $\overset{G-3}{00111010}$, $\overset{G-4}{10100011}$)

- GAs are the ways of solving problems by mimicking processes nature uses; i.e., Selection, Cross over, Mutation and Accepting, to evolve a Solution to a problem
- GAs are adaptive heuristic search based on the evolutionary ideas of natural selection and genetics.
- GAs are intelligent exploitation of random Search used in optimization problems.
- GAs, although radomized, exploit historical information to direct the search into the region of better performance with in the search space.

* Biological Background - Basic Genetics -

- Every organism has a set of rules, describing how that organism is built.
  All living organisms consists of cells.
- In each cell there is same set of chromosomes. Chromosomes are string of DNA and serve as a model for the whole organism.
- A chromosome consists of genes, blocks of DNA.
- Each gene encodes a particular protein that represents a trait (feature), e.g. Color of eyes.
- Possible settings for trait (e.g. blue, brown) are called alleles.

- Each gene has its own position in the chromosome called its locus.
- Complete set of genetic material (all chromosomes) is called genome.
- Particular set of genes in a genome is called gendype.
- The physical expression of the gendype is called the phendype, its physical and mental characteristics, such as colour, intelligence etc.
- When two organisms mate they share their genes; the resultant offspring may end up having half the genes from one parent and half from the other. This process is called recombination (Cross over).
- The new created offspring can then be mutated. Mutation means, that the elements of DNA are a bit changed. This changes are mainly caused by errors in copying genes from parents.
- The fitness of an organism is measured by success of the organism in its life (Survival).



Pseudo Code:

```
Begin
Initialise population with random Candidate Solut
Evaluate each Candidate;
Repeat Until (termination Condition) is Satisfi
                                            DO.
    1. SELECT parents;
    2. Recombine pairs of parents;
    3. Mutate the resulting offspring;
    4. SELECT individuals or the next gene
                                  ation;
END.
```

* Search Space:

In Solving problems, Some solution will be best among others.

The space of all feasible solutions is Cal Search space - Each point in the Sea space represents one possible Solution

- Each possible solution can be 'mar by its value (or fitness) for the proble
- The GA looks for the best solution am a number of possible solⁿ represente one point in the Search space.
- Looking for a Solution is then equal t for some extreme value (maxⁿ or m the Search Space.
- At times the Search space may be defined, but usually only a few p in the Search space are known.

In using GA, the process of finding solutions generates other points (possible solutions) as evolution proceeds.

* Working Principles:

chromosome:- as set of genes; a chromosome contains the solution in form of genes.

Gene:- a part of chromosome; a gene contains a part of solution. It determines the solution.
e.g. - 16743 is a chromosome and 1, 6, 7, 4, and 3 are its genes.

- Individual: Same as chromosome.
- Population: number of individuals present with same length of chromosome.
- Fitness: the value assigned to an individual based on how far or close a individual is from the solution; greater the fitness value better the solution it contains.
- Fitness function:- a function that assigns fitness value to the individual. It is problem specific.
- Breeding: taking two fit individuals and then intermingling there choromosome to create new two individuals.
- Mutation: changing a random ~~space~~ gene in an individual.
- Selection - selecting individuals for creating the next generation.

<u>Working</u> <u>Principles</u>: Genetic Algorithms begins with a set of solutions called the population.

- Solutions from one population are taken and used to form a new population. This is motivated

by the possibility that the new population will
beter than the old One.

- Solutions are selected according to th
fitness to form new solutions (off sp
more Suitable they are, more chance
have to reproduce.

- This is repeated until some C
(e.g. number of populations or imp
of the best solution) is Satisfied

* Outline of the Basic GA:

1. [Start] Generate random populat
n chromosomes (i.e., Suitable
for the problem).
2. [Fitness] Evaluate the fitness f
each chromosome x in the populati
3. [New population] Crete a new pop
by repeating following steps until
population is Complete.

ⓐ [Selection] select two parent ch
from a population according to their fit
the fitness, bigger the chance to be

ⓑ [Crossover] With a Crossover prob
cross over the parents to form n
offspring (children). If no Crossover
offspring is the exact-

of parents.

ⓒ [Mutation] With a mutation probability, mutate new offspring at each locus (position in chromosome).

ⓓ [Accepting] Place new offspring in the new population

4. [Replace] Use new generated population for a further run of the algorithm.

5. [Test] If the end Condition is Satisfied, stop and return the best solution in Current population.

6. [Loop] Go to step 2.

Note. The GA performance is largely influenced by two operators called Crossover and mutation. These two operators are the most important parts of GA

Encoding: Before a genetic algorithm Can be put to work on any problem, a method is needed to encode potential solutions to that problem in a form so that a Computer Can process.

- One Common approach is to encode solutions as binary Strings: sequences of 1's and 0's, where the digit at each position represents the value of some aspect of the solution.

Example: A Gene represents Some data (eye colour, hair colour, Sight, etc)

A Chromosome is an array of genes. In binary form.

a Gene looks like (11100010)

of parents.

(e) [Mutation] With a mutation probability, mutate new offspring at each locus (position in chromosome).

(f) [Accepting] Place new offspring in the new population.

4. [Replace] Use new generated population for a further run of the algorithm.

5. [Test] If the end Condition is Satisfied, stop, and return the best solution in Current population.

6. [Loop] Go to step 2.

__Note.__ The GA performance is largely influenced by two operators called Crossover and mutation. These two operators are the most important parts of GA

* __Encoding:__ Before a genetic algorithm Can be put to work on any problem, a method is needed to encode potential solutions to that problem in a form so that a Computer Can process.

- One Common approach is to encode solutions as binary Strings: sequences of 1's and 0's, where the digit at each position represents the value of some aspect of the solution.

__Example:__ A Gene represents Some data (eye colour, hair colour, Sight, etc)

A Charomosome is an array of genes. In binary form.

a Gene looks like (11100010)

a chromosome looks likes,

G-1, G-2, G-3, G-4

(11000010, 00001110, 00111010, 10100011)

A chrosome should in some way conta
about solution which it represents;
res encoding. The most popular of
is a binary string like.

Chromosome 1: 11011001 0011 0110
Chromosome 2: 11011110 0001 1110

Each bit in the string represent s
-teristics of the solution.

- There are many other ways of
encoding values as integer or real
or some permutations and so on.

- The virtue of these encoding me
on the problem to work on.

* Binary Encoding:- is the most (
represent information contained, i
algorithms, it was first used b
of its relative simplicity.

- In binary encoding, every chromosom

Chromosome 1: 10110010110010101
Chromosome 2: 1111111000001100

- Binary encoding gives many possible
even with a small number of
possible settings for a trait (fea

- This encoding is often not nat

Ex: Let two variables x_1, x_2 as (1011 0110)

Every variable will have both upper and lower limits as $x_i^L \leq x_i \leq x_i^U$.

Because 4-bit string can represent integers from 0 to 15.

as (0000 0000) and (1111 1111) represent the points for x_1, x_2 as (x_1^L, x_2^L) and (x_1^U, x_2^U) respectively.

Thus, an n-bit string can be represent integers from 0 to 2^n-1, ie, 2^n integers.

The equivalent value for any 4-bit string can be obtained as

$$x_i = x_i^L + \frac{(x_i^U - x_i^L)}{(2^{n_i} - 1)} x$$

let $x_i^L = 2$, $x_i^U = 17$. , $x_i = (1010)$.

$$S_i = 10$$

$$x_i = 2 + \frac{(17-2)}{(24-1)} \times 10 = 12$$

The accuracy obtained with a 4-bit code is $\frac{1}{16}$ of search space.

By increasing the string length by 1-bit, accuracy increases to 1/32.

* Value Encoding: The value encoding can be used in problems where values such as real numbers are used. Use of binary encoding for this type of problems would be difficult.

1. In value encoding, every chromosome is

is a Sequence of Some values.

2. The values Can be anything Connected to the problem, Such as: real numbers, character or objects.

Ex.

Chromosome A: 1.2324 5.3243 0.4556 2.329 2.45

Chromosome B: ABDJEIFJDHDIERJFDLDFLEFGT.

Chromosome C (back), (back), (right), (forward), (le

3. value encoding is often necessary to develop Some new types of crossovers and mutations specific for the problem.

Permutation Encoding: Permutation encoding Can be used in ordering problems, Such as travelling Salesman problem or task ordering problem.

1. In permutation encoding, every chromosome is a String of numbers that represent a position in a Sequence.

Chromosome A 1 5 3 2 6 4 7 9 8

Chromosome B 8 5 6 7 2 3 1 4 9

2. Permutation encoding is useful for ordering problems. For Some problems, crossover and mutation Corrections must be made to leave the chromosome Consistent.

Ex: 1. The Travelling Salesman problem.

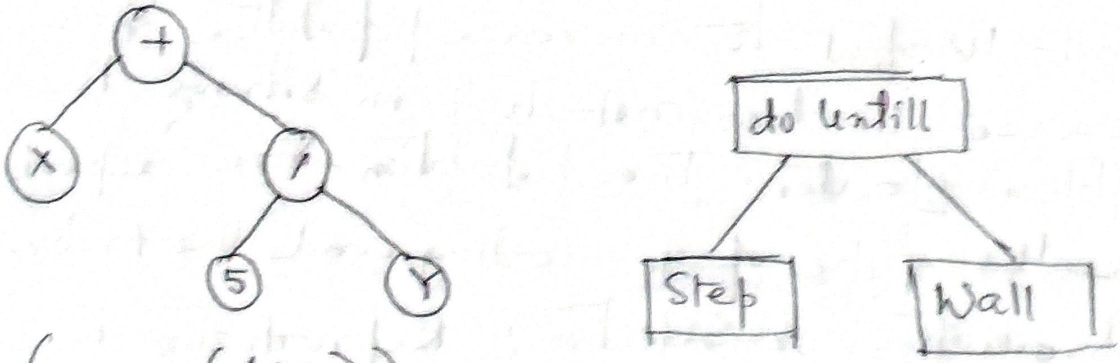
2. The Eight Queens problem.

Tree Encoding: Tree encoding is used mainly for evolving programs or expressions.

For genetic programming:

- In tree encoding, every chromosome is a tree of some objects, such as functions or commands in programming language.
- Tree encoding is useful for evolving programs or any other structures that can be encoded in trees.
- The crossover and mutation can be done relatively easy way.

Ex: Chromose A   Chromosome B



(+ x (/ 5 y))   (do until Step wall).

Fig. Example of chromosome with tree encoding

Operators of Genetic Algorithm: -

Genetic operators used in genetic ~~programming~~ algorithm maintain genetic diversity. Genetic diversity or variation is a necessity for the process of evolution. Genetic operators are analogus to those which occur in the natural world:

- Reproduction (or Selection).

In addition to these operators, there are some parameters of GA.

One important parameter is population size.

- Population size says how many chromosomes are in population (in one generation)

- If there are only few chromosomes, then GA would have a few possibilities to perform crossover and only a small part of search space is explored.

- If there are many chromosomes, then GA slows down.

- Research shows that after some limit, it is not useful to increase population size, because it does not help in solving the problem faster. The population size depends on the type of encoding and the problem

Reproduction or Selection: Reproduction is usually the first operator applied on population. From the population, the chromosomes are selected to be parents to crossover and produce offspring.

The problem is how to select these chromosomes?

According to Darwin's theory "Survival of the fittest" - the best ones should survive and create new offspring.

- The Reproduction operators are also called

- Selection means extract a subset of genes from an existing population, according to any defination of quality. Every gene has a meaning, so one can derive from the gene a kind of quality measurement called fitness function. Following this quality (fitness value), selection can be performed.

- Fitness function quantifies the optimality of a solution (chromosome) so that a particular solution may be ranked against all the other solutions. The function depicts the closeness of a given 'solution' to the desired result.

Many reproduction operators exists and they all essentially do same thing. They pick from current population the strings of above average and insert their multiple copies in the mating pool in a probabilistic manner.

The most commonly used methods of selecting chromosomes for parents to crossover are.

- Roulette wheel selection - Rank Selection
- Boltzmann Selection - Steady State Selection
- Tournament Selection

Example of Selection:

Evolutionary Algorithm is to maximize the function $f(x) = x^2$ with x in the integer interval $[0, 31]$.

i, $x = 0, 1, 2, \ldots, 31$.

1. The first step is encoding of chromoso
use binary representation for integers; 5
are used to represent integer up to 31.

2. Assume that the population size is 4.

3. Generate initial population at random.
are chromosomes or genotypes;
e.g. 01101, 11000, 01000, 10011.

4. Calculate fitness value for each individual.

ⓐ 01101 → 13; 11000 → 24; 01000 → 8; 10011 →

ⓑ Evaluate the fitness according to
$f(x) = x^2$,
13 → 169; 24 – 576; 8 → 64; 19 → 361

5. Select parents (two individuals) for crossover based on their fitness in pi
. Out of many methods for selecting the best chromosomes, if roulette-wheel selection is used, then the probability of the ith string in the population is

$$p_i = F_i / \left(\sum_{j=1}^{n} F_j\right), \text{ where}$$

Fi is fitness for the string i in the populat
expressed as f(x),
pi is the probability of the string i bei

n is no of individuals in the population, is population size, n = 4.
n * pi is expected Count.

String no	Initial Population	X-Value	Fitness Fi f(x) = x^2	pi = Fi/ΣFi	Expected Count n * pi
1	01101	13	169	0.14	0.56
2	11000	24	576	0.49	1.96
3	01000	8	64	0.06	0.24
4	10011	19	361	0.31	1.24
Sum			Σ 1170	1.00	4.00
Average			293	0.25	1.00
Max			576	0.49	1.96

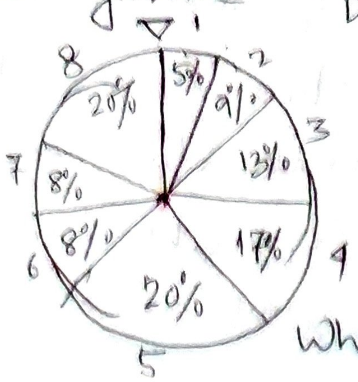
The string no 2 has maximum chance of selection

* Roulette Wheel Selection:- (Fitness-Proportionate Selection) is a genetic operator, used for selecting potentially useful solution for recombination.

In fitness-proportionate selection:

- the chance of an individual's being selected is proportional to its fitness, greater or less than its competitors' fitness.

- Conceptually, this can be thought as a game of Roulette.



The Roulette-wheel simulates 8 individuals with fitness values Fi, marked at its circumference, e.g. - the 5th individual has a higher fitness than others, so the wheel would choose the 5th individual more than the other individuals.

— the fitness of the individuals is calc
as the wheel is spun $n = 8$ times, eac
selecting an instance, of the string, ch
by the wheel pointer.

Probability of ith string is $p_i = F_i / (\sum_{j=1}^{n} F$

n = no. of individuals, called populat

p_i = prob. of ith string being selected

fitness for ith string in the populat

Because the circumference of the w
marked according to a string's fitness
Roulette-wheel mechanism is expected
$\frac{F}{\bar{F}}$ copies of the ith string

Average fitness = $\bar{F}$ F_j/n ; Expected
$= (n-8) \times p_i$

Cumulative Probability $= \sum_{i=1}^{N=5} p_i$

Crossover: Crossover is a genetic ope
that combines (mates) two chromosomes (
to produce a new chromosome (offspring
idea behind crossover is that the ne
some may be better than both of the p
if it takes the best characteristics fro
the parents. Crossover occurs du

chromosomes and creates a new offspring.

The crossover operators are of many types.
- One simple way is, One-Point crossover.
- the others are Two point, uniform, Arithmetic, and Heuristic crossovers.

The operators are selected based on the way chromosomes are encoded.

One-point crossover:

One point crossover operator randomly selects one crossover point and then copy everything before this point from the first parent and then everything after the crossover point copy from the second parent. The crossover would then look as shown.

Consider the two parents selected for crossover

Parent-1 | 11 0 11 | 0010011 0110 |

Parent-2 | 11 0 11 | 11000011110 |

After crossover point.

The offspring produced are.

Offspring-1 | 11 011 11 0000111 10 |

Offspring-2 | 11 011 001 00110110 |

* Two point crossover:

Two point crossover operator randomly slects two crossover point within a chromo

then interchanges the two parent
between these points to produ
offspring.

Parent-1	11011	0010011	0110
Parent-2	11011	1100001	1110

Offspring-1	11011 1100001 011
Offspring-2	11011 0010011011

<u>Uniform Crossover</u>: Uniform crossov
(with some probability – Knew as
ratio) which parent will contribute
values in the offspring chromoson
over operator allows the parent
to be mixed at the gene level,
segment level (as with one an
crossover).

Parent-1	110110010 011 0110

The possible set of offspring after uniform crossover would be

offspring-1 | 1 1 0 1 1 1 1 0 0 0 0 1 1 1 1 0 |

offspring-2 | 1 1 0 1 1 0 0 1 0 0 1 1 0 1 1 0 |

* **Arithmetic**: Arithmetic crossover operator linearly combines two parent chromosome vectors to produce two new offspring according to the equations:

offspring-1 = a * parent1 + (1-a) * parent2

offspring-2 = (1-a) * parent 1 + a * parent 2

Where a is a random weighting factor chosen before each crossover operation.

Consider two parents (each of 4 float genes) Selected for crossover:

Parent 1	(0.3)	(1.4)	(0.2)	(7.4)
Parent 2	(0.5)	(4.5)	(0.1)	(5.6)

Applying the above two equations and assuming the weighting factor a = 0.7, applying above equations, we get two resulting offspring. The possible set of offspring after arithmetic crossover would be:

of the two ~~parenty~~ parent chromoso
mine the direction of search.
The offspring are created accord
equation.

Offspring-1 = Best parent + r * (r

offspring-2 = Best parent.

Where r is a random between

It is possible that offspring-1 wi
feasible. It can happen if r is c
One or more of its genes fall out
allowable upper or lower bound
reason, heuristic crossover ha
defined parameter n for the
of times to try and find a
results in a feasible chromosome
chromosome is not produced aft
, the worst parent is returne
spring-1.

Mutation: After a crossover is pe
lation takes place. Mutation is a gene
used to maintain genetic diver
one generation of a population of
to the next.

Mutation occurs during evolution a

first choice.

Mutation alters one or more gene values in a chromosome from its initial state. This can result in entirely new genes values being added to the gene pool. With the new genera values being added to the gene pool. With the new gener values, the genetic algorithm may be able to arrive at better Solution than was previously possible.

Mutation is an important part of the genetic search, helps to prevent the population from stagnating at any local optima. Mutation is intended to prevent the search falling into a local optimum of the state space.

The mutation operators are of many types.
- One Simply way is, Flip Bit
- The others are Boundary, Non-uniform, Uniform, and Gaussian.

The operators are selected based on the way chromosomes are encoded.

Flip Bit: The mutation operator Simply inverts the value of the chosen gene i.e, 0 goes to 1 and 1 goes to 0.

Consider the two original off-springs selected for mutation.

Original Offspring-1	110111100001110

The Mutated off-spring produced ar

Mutated off-spring-1

1 1 0 0 1 1 1 0 0 0 0 1
1 1 0 1 1 0 1 1 0 0 1 1

* Boundary :- The mutation opera the value of the chosen gene wit the upper and lower bound fo (chosen randomly).
This mutation operator can only b integer and float gene.

* Non-Uniform: The mutation o eases the probability s.t. the of the mutation will be close t the generation number increases. lion operator prevents the popu stagnating in the early stages lution then allows the genetic to fine tune the solution in th stages of evolution.
This mutation operator can o used for integers and float

pecified upper and lower bounds
ne:
n operator Can only be used for inte-
float genes

The mutation operator adds a unit
stributed random value to the chosen
new gene value is clipped if it falls
the user-specified lower or upper
r that gene.
ion operator Can only be used for
float genes.

e Algorithm:
demonstrate and explain: Random
itness, Selection, Crossover, Mutation

pling
$f(x) = x^2$ over the range of
om 0 --- 31.
a means to represent a solution
blem: Assume, we represent
ive-digit unsigned binary integers

3. Coding – Binary and the strin
GAs often process binary represe
solutions. This works well, bec
and mutation can be clearly d
binary solutions. A binary st
5 can represents 32 numbers

4. Randomly generate a set of
Here, Considered a population of
However, larger populations are
applications to explore a larger
search. Assume, four random
solutions as: 01101, 11000, 0100
These are chromosomes or ge

5. Evaluate the fitness of eac
the population: The calculated f

ⓐ for each individual are –
01101 → 13; 11000 → 24; 01000 →

ⓑ Evaluate the fitness accordi
13 → 169, 24 → 576, 8 → 6

ⓒ [illegible]

6. Produce a new generation of solutions by picking from the existing pool of solution with a preference for solutions which are better suited than others:
We divide the range into four bins, sized according to the relative fitness of the solution which they represent

Strings	probi	Associated bin
01101	0.14	0.0 — — 0.14
11000	0.49	0.14 — — 0.63
01000	0.06	0.63 — — 0.69
10011	0.31	0.69 — 1.00

By generating 4 uniform (0,1) random values and seeing which bin they fall into we pick the four strings that will form the basis for the next generation:

Random no	Falls into bin	Chosen string
0.08	0.0 - - - 0.14	01101
0.24	0.14 - - - 0.63	11000
0.52	0.14 - - - 0.63	11000
0.87	0.69 - - - 1.00	10011

7. Randomly pair the numbers of the new generation:- Random number generator decides

are a mixture of the parents:

For the first pair of strings 01101
- We random select the crossover
to be after the fourth digit. Cross
two strings at that point yields.

01101 ⇒ 0110|1 ⇒ 01100
11000 ⇒ 110010 ⇒ 11001

For the 2nd pair of strings : 1100
- we randomly select the crosso
to be after the second digit.
Crossing these two strings at that

11000 ⇒ 111000 ⇒ 11011
10011 ⇒ 101011 ⇒ 10000

9. Randomly mutate a very small
of genes in the population:
With a typical mutation probabilit
bit it happens that none of the
in our population are mutated.

10. Go back and re-evaluate fitne
population (new generation).
This would be the first step in g
generation of solutions. Howe

~~mple~~. Sample.

o	Initial population (chromosome)	x-value (phenotypes)	Fitness f(x) = x²	prob (i)	Expected Count
	01100	12	144	0.082	0.328
	11001	25	625	0.356	1.424
	11011	27	729	0.416	1.664
	10000	16	256	0.146	0.584
,			1754	1.00	4.000
e			438	0.250	1.000
			729	0.416	1.669

– That:

ial populations: ~~A~~ were

01, 11000, 01000, 10011

ne cycle, new populations, at step 10,

100, 11001, 11011, 10000.

e total fitness has gone from 1170 to 1754

Single generation.

Algorithm has already come up with

ng 11011 (i.e, $x = 27$) as a possible

$y = \sqrt{x}$, $1 \le x \le 16$.

P_c = ·70 . Pos (for crossover) = 2 .

Random no. for Crossover

0.62, 0.80, 0.50, 0.47, 0.75

P_m = 0.03

Random nos for mutation

0.61, ·21, ·75, ·08, ·04, ·91, ·45, ·11

·05, ·09, ·12, ·41, ·51, ·62, ·78, ·84,

·33, ·07, ·06, ·55, ·15, ·29, ·37,

·02, ·61, ·82, ·92, ·83

Ans

String no.	Initial population	~~x value~~ Decode value	x val	Fitness value	Prob (i)
1	100101	37	9.89	~~6.083~~ 3.15	·18
2	011010	26	7.19	~~5.099~~ 2.68	·15
3	010110	22	6.29	~~4.690~~ 2.50	·14
4	111010	58	14.9	~~7.616~~ 3.55	·22
5	101100	44	11.8	~~6.633~~ 3.39	·19
6	001101	13	4.0	2.02	·12
				$\sum$ 17.57	1.00

Average = 2.93

$$x_i = x_i^{min} + \frac{x_i^{max} - x_i^{min}}{2^l - 1} \times \text{Decode value}$$

~~Here~~ ~~Now we~~ Here Chromosome - 4 is

One as chromosome - 6 is the we

Now we proceed to Roullette whee

we spin the Roullette wheel 6 times

are select a single chromoso

Let us consider that a sequence of random numbers from the range [0, 16] is

0.15, 0.27, 0.64, 0.52, 0.79, 0.70

The 1st random no $r = 0.15 < q_1 = .18$ Hence the 1st chromosome is selected for the new populat

$r = 0.27 < q_2 = .33, > q_1 = .18$ so ~~v_2~~ 2nd chromosome is selected

∴ new population is

$v_1' = 100101 \; (= v_1)$ $v_3' = 111010 \; (= v_3)$

$v_2' = 011010 \; (= v_2)$ $v_4' = 111010 \; (= v_4)$

$v_5' = 101100 \; (= v_5)$ $v_6' = 101100 \; (= v_5)$.

Now apply crossover. in $P_c = .70$, 70% of chromosome, i.e., $P_c \times N = .70 \times 6 = 4$ chromosomes undergo

Crossover.

No. of parents $= P_c \times N = 4$.

Selection of parents

For each chromosomes in the new population we generate a random no. from the rage [0, 1]: The sequence of random no's are

As $r = .62 < .70 \; (= P_c)$, v_1' is selected as a parents. Similarly v_3', v_4', v_6' are selected. Here the pairwise chromosome as (v_1', v_3') & (v_4', v_6').

$v_1' = 10|0101$ $v_1'' = 101011$

For 2nd pair.

$v_4' = 1111010$ $\Rightarrow$ $v_4'' = 111100$

$v_6' = 1011100$ $\quad$ $v_6'' = 101010$

After Crossover the new population is

$v_1'' = 101010$

$v_2'' = 011010 \; (= v_2')$

$v_3'' = 110101$

$v_4'' = 111100$

$v_5'' = 101100 \; (= v_5')$

$v_6'' = 101010$

Mutation :

Here mutation is performed on a bit

Number of bits to be mutated = $P_m \times$

$= .03 \times ($

$= 1.08 \simeq$

Here 1 bit is mutated in one gener

. Again we generate random numb from the range [0, 1] for each bit . random no.s are given in the problem

As $(r = .02) < P_m = 0.03$, The 32 bit is mutated i.e, 2nd bit in 6-th chromosome is mutated

$v_6'' = 101010 \Rightarrow v_6'' = 111010.$

Hence after 1st generation we have

Population String no	Population	Decoded value	x value	fitness value
1	101010	42	11.00	3.32
2	011010	26	7.67	2.77
3	110101	53	13.62	3.69

Now it is repeated for several times for Mangen iteration

For each pair of coupled chromo

a random number Pos (say) from

range [1, 2, ---, m-1] m being the

length - i.e, number of bits in chro

<u>Mutation</u>: For binary representation :

ⓘ Generate a random number (flo

the range [0, 1].

ⓘⓘ If $r \le P_m$, mutate the bit,

<u>Ex</u>: Find the max $f(x) = x^3 - 12x^2 +$

(0, 4) with N = 5 (no. of popu

$P_c = 0.4$, $P_m = 0.2$

<u>Initial population</u> : 1.852, 3.828, 1.380, 1.47

Random Nos for Selection - 0.46, 0.30, 0.82

" " " Cross-over - 0.346, 0.130, 0.98

~~$\alpha = 0.346$~~

Random no. of mutation 0.19, 0.59, 0.65,

$r = \cdot 59$, $\Delta = 1.20$.

<u>Ans</u>.

Chromosome No.	Initial Population	Fitness	prob. (p_i)	Cu
0	1.852	48.53	·207	
1	3.828	52.51	·224	
2	1.380	41.88	·179	
	[illegible]	43.43	·186	

p-2) Selection of mating pool.

$.46 < .610$, the chromosome-2 is Selected

$.30 < .431$ " " -1 is Selected.

$0.82 < 1.00$ " " -4 is Selected

$0.90 < 1.00$ " " -4 is Selected

$0.56 < 0.610$ " " 2 is Selected.

ere the selected chromosome are $(2, 1, 4, 4, 2)$

ter selection, New population is

.380, 3.828, 1.776, 1.776, 1.380.

sover: Let the crossover probability be No. of mosome in the mating pool is no. of parents are

$= P_c \times N = 0.4 \times 5 = 2$.

) Selection of chromosome as parents:

– the random nos (between 0 & 1) be 0.346, 0.130

982, 0.090, 0.656.

$\alpha = 0.346 < .4 (= P_c)$ the 0-th chromosome is

cted as parents

$0.130 < .4 (= P_c)$ So 1-th chromosome is Selected

nce chromosome Selected as parents are 0, 1

ce parents are: 1.38, 3.828.

4) Let $\lambda = 0.346$

g arithmetic crossover; children are

$= \lambda x_0 + (1-\lambda) x_1 = 0.346 \times 1.38 + 0.654 \times 3.828$

$= 2.97$

$= (1-\lambda) x_0 + \lambda x_1 = 2.224$.

population after

, 2.97 is mutated by the method
mutation

) = Proriginal + (r − 0.5) Δ

x. value of Permulation = 1.20

97 + (0.55 − 0.5) × 1.20

97 + .05 × 1.20 = 2.97 + 0.06 = 3.03.

n is i, after One generation
3.03, 2.224, 1.776, 1.776, 1.380.

Fitnes	~~P~~i (pi)	qi	Avg. valu
54.00	0.222	0.222	1.110
51.73	0.213	0.435	2.175
47.67	0.196	0.631	3.155
47.67	0.196	0.827	4.135
41.88	0.172	0.999	4.995
42.95	.999	1.00	15.570
	≃1.00 Avg:-	0.200	3.114.

membership functions define the functi
mapping of the system.
btain the membership functions ~~define the~~
~~al~~ n with best fitness value.

ion Reasoning: Law of induction

a set of irreducible out comes of an experim-
e induced probabilities are those probabilities
t with all available information that maximi-
entropy of the set
nduced probability of a set of independent
ions is proportional to the probability density
nduced probability of a single observation.

induced rule is that rule consistent with all
e information of that minimizes the entropy.

rd law widely used for the developement of
ship functions. The membership functions
luctive reasoning are generated as
zzy threshold is to be established between
l - data.
entropy minimization screening method, first
the threshold line.
start the segmentation process.
egmentation process results into two classes.
partitioning the first two classes one
e, We obtain three different classes.
rtitioning is repeated with threshold value

Defuzzification: Defuzzification is a mapping process from a space of fuzzy Control action defined over an output universe of discourse into a space of Crisp Control actions.

1. Lambda-Cuts for fuzzy sets (Alpha-Cuts)

Consider a fuzzy set $\underset{\sim}{A}$. The set A_λ $(0<\lambda<1)$, Called the Lambda (λ)-Cut (or alpha $[\alpha]$-Cut) set, is a crisp set of the fuzzy set and is defined as $A_\lambda = \{x; \mu_{\underset{\sim}{A}}(x) \geq \lambda\} : \lambda \in [0,1]$

The set A_λ is Called a weak lambda-cut set if it Consists of all the elements of a fuzzy set whose membership functions have values greater than or equal to a specified value.

A string λ-Cut set is given by

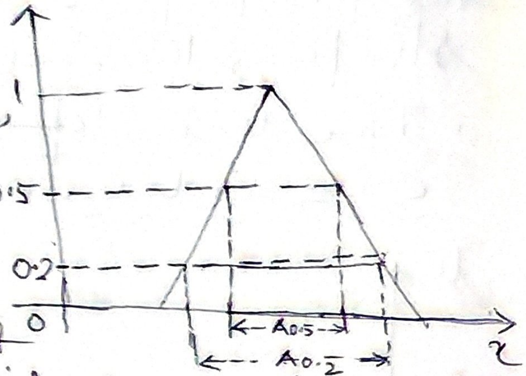
$$A_\lambda = \{x; \mu_{\underset{\sim}{A}}(x) > \lambda\} : \lambda \in [0,1]$$

All the λ-Cut sets form a family of crisp set. It is important to note ~~that~~ the λ-Cut set A_λ does not have a tild Score, because it is a crisp set derived from parent fuzzy set $\underset{\sim}{A}$. Any particular fuzzy set $\underset{\sim}{A}$ Can be transformed into an infinite no. of λ-cut-sets, because there are infinite no. of values λ Can take inthe interval [0,1

The properties of λ-cut sets are as

1. $(\underset{\sim}{A} \cup \underset{\sim}{B})_\lambda = A_\lambda \cup B_\lambda$

$\leq \beta$, where $0 \leq \beta \leq 1$, it is true that
Where $A_0 = X$.



r $\lambda = 0.2$ and $\beta = 0.5$,
er domain than $A_{0.5}$
$0.2 \leq 0.5$), $A_{0.5} \subseteq A_{0.2}$

The $\lambda = 1$-cut set
of A is the λ-cut
re $\lambda = 0+$, and it can be defined as
$(x) > 0\}$. The interval $[A_{0+}, A_1]$ forms
of the fuzzy set A

ts for Fuzzy relation :- The λ-cut
ations is similar to that for fuzzy
be a fuzzy relation where each row
al matrix is considered a fuzzy set.
in a fuzzy relation matrix R denotes
embership function for a fuzzy set
y relation can be converted into a
$R_\lambda = \{(x,y) ; \mu_R(x,y) \geq \lambda\}$. where
t relation of the fuzzy relation R.

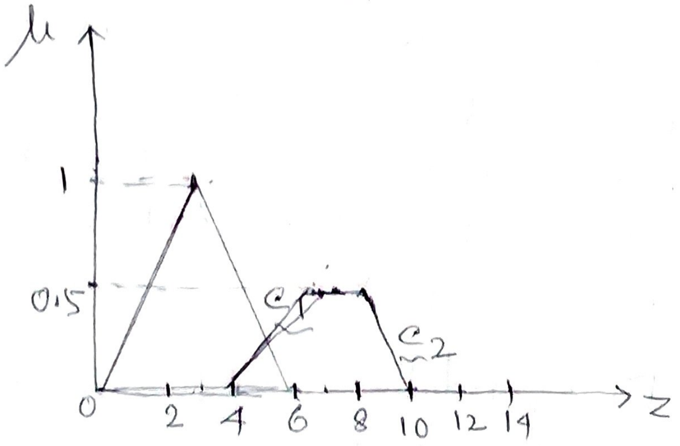
$R \cup S)_\lambda = R_\lambda \cup S_\lambda$

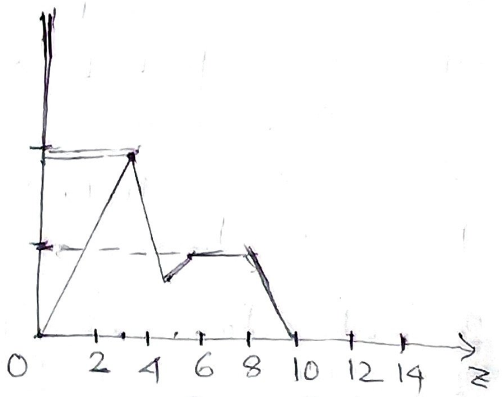
$S)_\lambda = R_\lambda \cap S_\lambda$

$(\overline{R_\lambda})$ except when $\lambda = 0.5$.
y $\lambda \leq \beta$, $0 \leq \beta \leq 1$, it is true that

n Methods : This is the process of
a fuzzy quantity into a precise
he output of a fuzzy process
on of two or more fuzzy mem-
tions defined on the universe
riable.

Consider a fuzzy output comprising two parts; The first part, $\underset{\sim}{C}_1$, a triangular membership shape, the second part $\underset{\sim}{C}_2$, a trapezoidal shape. The union of these two membership functions $\underset{\sim}{C} = \underset{\sim}{C}_1 \cup \underset{\sim}{C}_2$ involves the max-operator



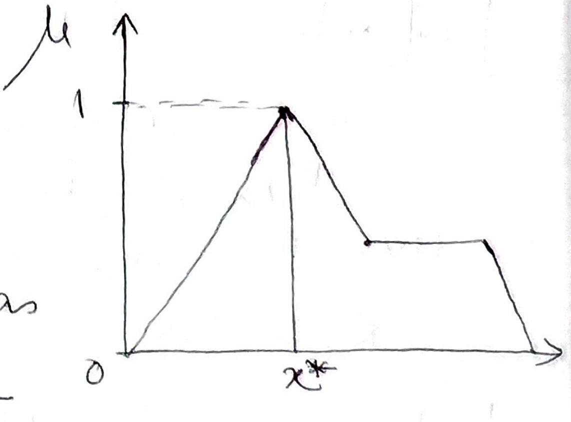


$\underset{\sim}{C} = \underset{\sim}{C}_1 \cup \underset{\sim}{C}_2$

Similarly $\underset{\sim}{C}_n = \bigcup_{i=1}^{n} \underset{\sim}{C}_i = \underset{\sim}{C}$.

1. Max-Membership Principle.

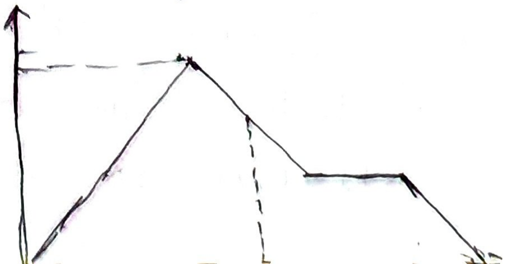
This method is also known as height method and is limited to peak output functions. This method is given by the algebric expression $\mu_{\underset{\sim}{C}}(x^*) \geq \mu_{\underset{\sim}{C}}(x) \ \forall \ x \in X$.



2. Centroid Method.

This method is also known as centre of mass, centre of area or centre of gravity method. It is the most commonly used defuzzification method. The defuzzified output x^* is defined as

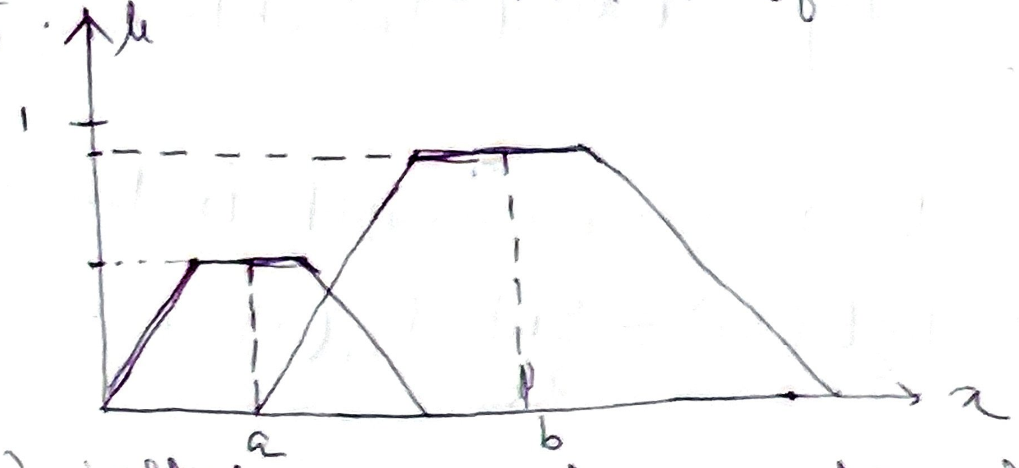
$$x^* = \frac{\int \mu_{\underset{\sim}{C}}(x) \cdot x \, dx}{\int \mu_{\underset{\sim}{C}}(x) \, dx}.$$



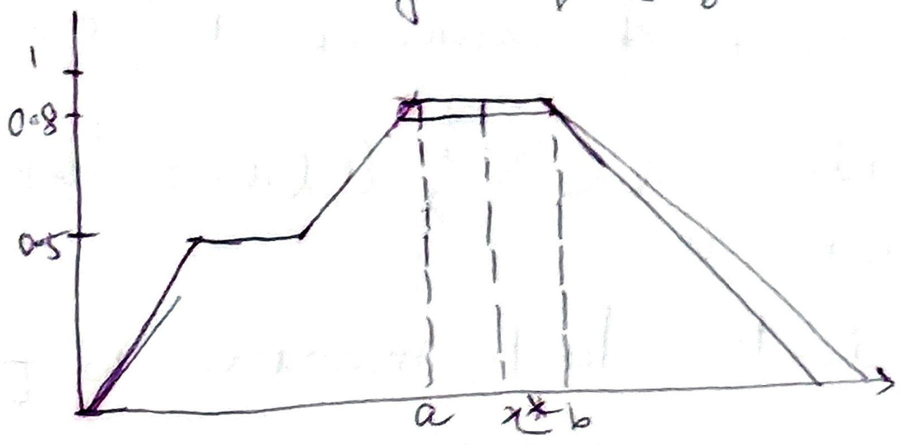
3. Weighted Average Method is valid for symmetr output membership functions only. Each memb -ship function is weighted by its maximum membership value. The output in this case.

$$x^* = \frac{\sum \mu_{\underset{\sim}{C}}(\bar{x}_i)\cdot \bar{x}_i}{\sum \mu_{\underset{\sim}{C}}(\bar{x}_i)}$$

~~Here~~ $\bar{x}_i$ is the maximum of the ith membership fu ction.



Weighted average defuzzification method.



4. Mean - Max Membership:

This method is also known as the middle of the max ma.

$$x^* = \frac{\sum_{i=1}^{n} \bar{x}_i}{n}.$$

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

5. Centre of Sums:

$$\int x \sum^{n} \mu_{\underset{\sim}{C}_i}(x)\, dx$$

7. First of Maxima (Last Maxima)

This method uses the overall out of all individual output fuzz for determining the smallest val domain ~~which~~ with maximized me in $\underset{\sim}{C_i}$. The steps used for obt are

1. Initially, the maximum height i is found: $hgt(\underset{\sim}{C_i}) = \sup\limits_{x \in X} \mu_{\underset{\sim}{C_i}}(x)$

2. Then the first ~~of~~ maxima is

$$x^* = \inf_{x \in X} \left\{ x \in X ; \mu_{\underset{\sim}{C}}(x) = \right.$$

3. After this the last maxima

$$x^* = \sup_{x \in X} \left\{ x \in X ; \mu_{\underset{\sim}{C}}(x) = \lambda \right.$$

Ex.1 Consider two fuzzy sets both defined on X, given as

$\mu(x_i X)$	x_1	x_2	x_3	x_4	x_5
$\underset{\sim}{A}$	0.2	0.3	0.4	0.7	0.1
$\underset{\sim}{B}$	0.4	0.5	0.6	0.8	0.9

S The following λ-Cut sets

$2, x_3, x_4, x_5\}$

$= \max [\mu_{\underset{\sim}{A}}(x), \mu_{\underset{\sim}{B}}(x)]$

$\{ \frac{0.4}{x_1} + \frac{0.5}{x_2} + \frac{0.6}{x_3} + \frac{0.8}{x_4} + \frac{0.9}{x_5} \}$

$= \{ x_3, x_4, x_5 \}$

$) = \min \{ \mu_{\underset{\sim}{A}}(x), \mu_{\underset{\sim}{B}}(x) \}$

$\frac{0.2}{x_1} + \frac{0.3}{x_2} + \frac{0.4}{x_2} + \frac{0.7}{x_4} + \frac{0.1}{x_5} \}$

$0.5 = \{ x_4 \}$

$) = \max \{ \mu_{\underset{\sim}{A}}(x), \mu_{\bar{\underset{\sim}{A}}}(x)]$

$\frac{3}{1} + \frac{0.7}{x_2} + \frac{0.6}{x_3} + \frac{0.7}{x_4} + \frac{0.9}{x_5} \}$

$.7 = \{ x_1, x_2, x_4, x_5 \}.$

$= \min \{ \mu_{\underset{\sim}{B}}(x), \mu_{\bar{\underset{\sim}{B}}}(x) \}$

$\{ \frac{0.4}{x_1} + \frac{0.5}{x_2} + \frac{0.4}{x_3} + \frac{0.2}{x_4} + \frac{0.1}{x_5} \}$

$3 = \{ x_1, x_2, x_3 \}.$

$) = 1 - \mu_{(\underset{\sim}{A} \cap \underset{\sim}{B})} = \{ \frac{0.8}{x_1} + \frac{0.7}{x_2} + \frac{0.6}{x_3} + \frac{0.3}{x_4} + \frac{0.9}{x_5} \}$

$0.6 = \{ x_1, x_2, x_3, x_5 \}.$

$= \max [\mu_{\bar{\underset{\sim}{A}}}(x), \mu_{\bar{\underset{\sim}{B}}}(x)]$

$\frac{0.8}{x_1} + \frac{0.7}{x_2} + \frac{0.6}{x_3} + \frac{0.3}{x_4} + \frac{0.9}{x_5} \}$

$8 = \{ x_1, x_5 \}.$

the two fuzzy sets,

$\frac{0}{0.2} + \frac{0.8}{0.4} + \frac{1}{0.6} \}$ and $\underset{\sim}{B} = \{ \frac{0.9}{0.2} + \frac{0.7}{0.4} + \frac{0.3}{0.6} \}$

ch notations, express the fuzzy sets into

for $\lambda = 0.4$ and $\lambda = 0.7$ for the follow

3. Consider the discrete fuzzy set defined on the universe $X = \{a, b, c, d, e\}$ as

$$\underset{\sim}{A} = \left\{ \frac{1}{a} + \frac{0.9}{b} + \frac{0.6}{c} + \frac{0.3}{d} + \frac{0}{e} \right\}.$$

Using Zadeh notation, find the λ-cut sets for $\lambda = 1, 0.9, 0.6, 0.3, 0^+$ and 0.

Ans. The fuzzy set given on the universe of discourse is

$$\underset{\sim}{A} = \left\{ \frac{1}{a} + \frac{0.9}{b} + \frac{0.6}{c} + \frac{0.3}{d} + \frac{0}{e} \right\}.$$

The λ-cut set is given as. $A_\lambda = \{x ; \mu_{\underset{\sim}{A}}(x) \geq \lambda\}$

ⓐ $\lambda = 1, \; A_1 = \left\{ \frac{1}{a} + \frac{0}{b} + \frac{0}{c} + \frac{0}{d} + \frac{0}{e} \right\}$

ⓑ $\lambda = 0.9, \; A_{0.9} = \left\{ \frac{1}{a} + \frac{0.9}{b} + \frac{0}{c} + \frac{0}{d} + \frac{0}{e} \right\}$

ⓒ $\lambda = 0.9 \;\; A_{0.6} = \left\{ \frac{1}{a} + \frac{1}{b} + \frac{1}{c} + \frac{0}{d} + \frac{0}{e} \right\}$

ⓓ $\lambda = 0.3 \;\; A_{0.3} = \left\{ \frac{1}{a} + \frac{1}{b} + \frac{1}{c} + \frac{1}{d} + \frac{0}{e} \right\}$

ⓔ $\lambda = 0+, \;\; A_{0+} = \left\{ \frac{1}{a} + \frac{1}{b} + \frac{1}{c} + \frac{1}{d} + \frac{0}{e} \right\}$

ⓕ $\lambda = 0, \;\; A_0 = \left\{ \frac{1}{a} + \frac{1}{b} + \frac{1}{c} + \frac{1}{d} + \frac{1}{e} \right\}$

4. Determine the crisp λ-cut relation when $\lambda = 0.1, 0^+, 0.3$ and 0.9 for the relation

$$\underset{\sim}{R} = \begin{bmatrix} 0 & 0.2 & 0.4 \\ 0.3 & 0.7 & 0.1 \\ 0.8 & 0.9 & 1.0 \end{bmatrix}$$

Ans.

$$R_\lambda = \{(x,y) ; \mu_{\underset{\sim}{R}(x,y)} \geq \lambda\}$$

$$= \{1 ; \mu_{\underset{\sim}{R}(x,y)} \geq \lambda ; 0 : \mu_{\underset{\sim}{R}(x,y)} < \lambda\}.$$

$$R_{0.1} = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

$$\begin{matrix} & 0.5 & 0.3 \\ 55 & 1 & 0.6 \\ 6 & 1 & 0 \\ 1 & 0.3 & 0 \end{matrix}\Bigg]$$

find the λ cut relation for $\lambda = 0^+$, 0.1, 0.4 and 0.8.

ation of a fuzzy equivalence
p equivalence relation.

$$\begin{matrix} 0.4 & 0.5 & 0.8 \\ 0.4 & 0.5 & 0.9 \\ 1 & 0.4 & 0.4 \\ 0.4 & 1 & 0.5 \\ 0.4 & 0.5 & 1 \end{matrix}\Bigg]$$

, $\mu_{\underset{\sim}{R}}(x_2, x_5) = 0.9$, $\mu_{\underset{\sim}{R}}(x_1, x_5) = 0.8$ —①

min $[\mu_{\underset{\sim}{R}}(x_1, x_2), \mu_{\underset{\sim}{R}}(x_2, x_5)]$

$= \min[0.8, 0.9] = 0.8$ —②

transitive property satisfied

quivalence relation. Now assume
elation formed is

ip function as shown in Fig.

fied output value by seven

Ans, The defuzzified output value can be obtained

1. Centroid method: The two points (0,0) and (2,0.7). The straight line is given by

$$(y - y_1) = m(x - x_1) \Rightarrow y - 0 = \frac{0.7}{2}(x - 0).$$

$A_{11} \Rightarrow y = 0.35x,$

$A_{12} \Rightarrow y = 0.7.$

$A_{13} \Rightarrow$ not necessary

$A_{21} \Rightarrow$ the two points are (2,0), (3,1).

$y = x - 2$

$A_{22} \Rightarrow y = 1$

$A_{23} \Rightarrow$ the two points are (4,1), (6,0).

$y = -0.5x + 3.$

Ⓐ From A_{12} we obtain $y = 0.7,$

Ⓑ From A_{21} we obtain $y = x - 2$, on substituting the value $y = 0.7$ in Ⓑ, we obtain

$$x - 2 = 0.7 \Rightarrow x = 2.7, \; y = 0.7.$$

$$x^* = \frac{\int \mu_{\underset{\sim}{C}}(x)\, x\, dx}{\int \mu_{\underset{\sim}{C}}(x)\, dx} = \frac{\int_0^2 0.35x^2 dx + \int_2^{2.7} 0.7x\, dx + \int_{2.7}^{3} (x^2 - 2x)dx + \int_3^4 x\, dx + \int_4^6 (-0.5x^2 + 3x)dx}{\int_0^2 0.35x\, dx + \int_2^{2.7} 0.7\, dx + \int_{2.7}^{3} (x - 2)dx + \int_3^4 dx + \int_4^6 (0.5x + 3)dx}$$

$$= \frac{10.78}{3.445} = 3.187.$$

* Weighted average method $x^* = \dfrac{2 \times 0.7 + 4 \times 1}{0.7 + 1}$

$= 3.176.$

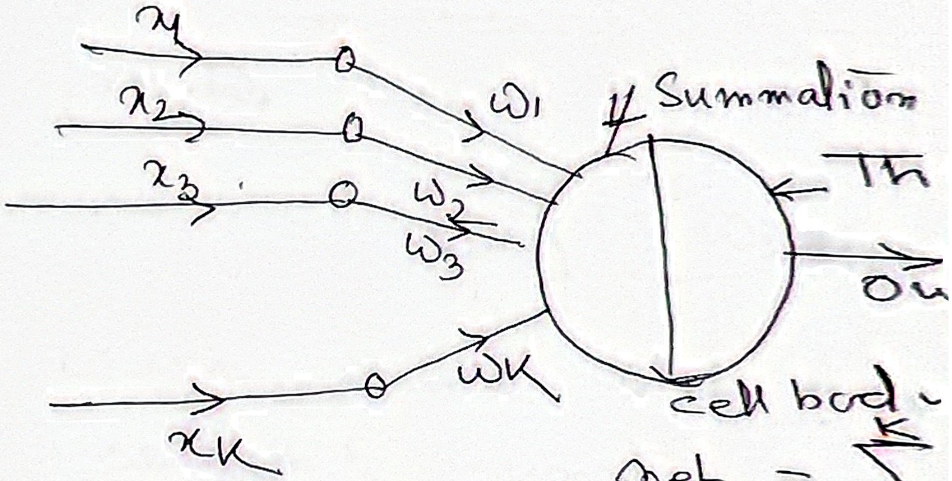
* Mean-max method:

$$x^* = \frac{a+b}{2} = \frac{2.5 + 3.5}{2} = 3.$$

Artificial Neural Net

Model of an artificial

We Know that the huma Complex Structure viewed highly enter Cannected n processing elements Called of a neuron Can be ca as below. Here every bears a direc analogy to of a biological neuron as artificial neuron.



$net_K = \sum_{i}^{K}$

<u>Input</u>: Let $X = (x_1, x_2,$ the artificial neuron.

let $W = (w_1, w_2, -- w$ ched to the input links.

Here the total input I

filter called Activation function / Threshold function / Transfer function / Squash function which released output $y = \phi(I)$

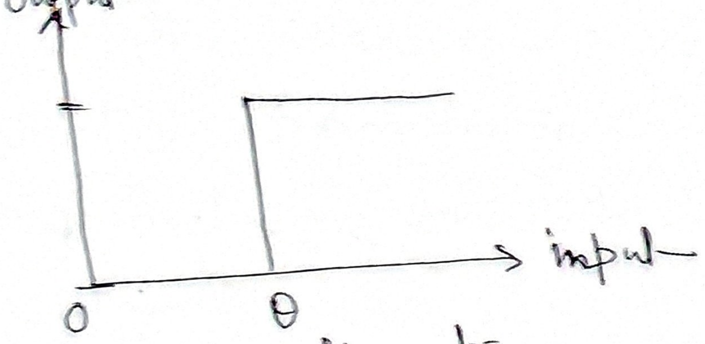
$$= \phi(net), \quad net = x^T w \qquad — ②$$

A very commonly used activation function is the thresholding function: ~~The Su~~ Here the sum is compared with ~~the~~ a threshold value θ. ~~The neuron~~ If the value of I is greater than θ then output is 1, else 0.

$$\text{i.e., } y = \phi\left\{\sum_{i=1}^{n} w_i x_i - \theta\right\} \qquad — ③.$$

where ϕ is a step function known as Heaviside function and is such that

$$\phi(net) = \phi(I) = \begin{cases} 1 & \text{if } I \geq \theta \\ 0 & \text{if } I < \theta. \end{cases}$$

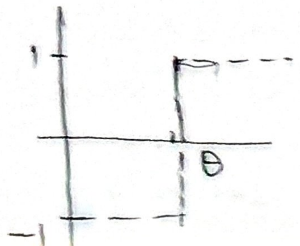


Threshold function

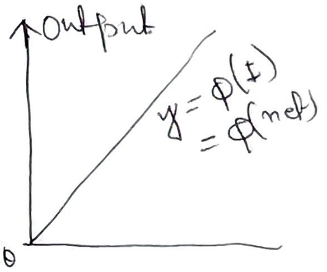
Other choice for Activations function.

Signum functions

$$\phi(I) = \begin{cases} 1, & I \geq \theta \\ -1 & I < \theta \end{cases}$$



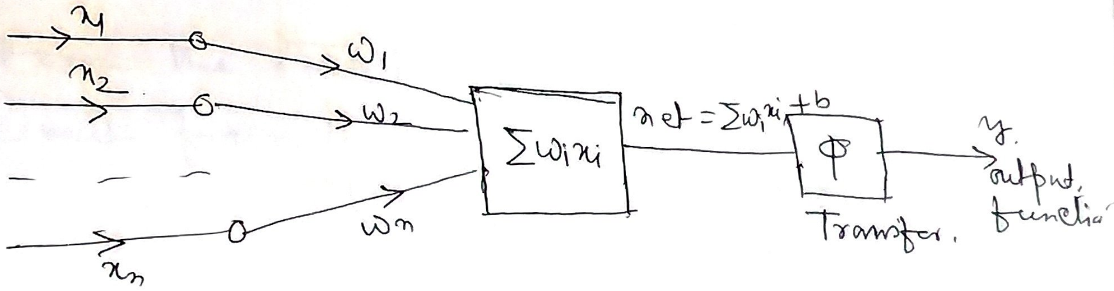
Linear function :



Hard limit function

$$\phi(net) = \begin{cases} +1 & \text{is } net \geq \theta \\ 0 & \text{if } net < \theta \end{cases}$$

Bias: The bias input the performance of the neuron network if bias is present then



Here both w and b are adjustable ; scalar parameter of the neuron.

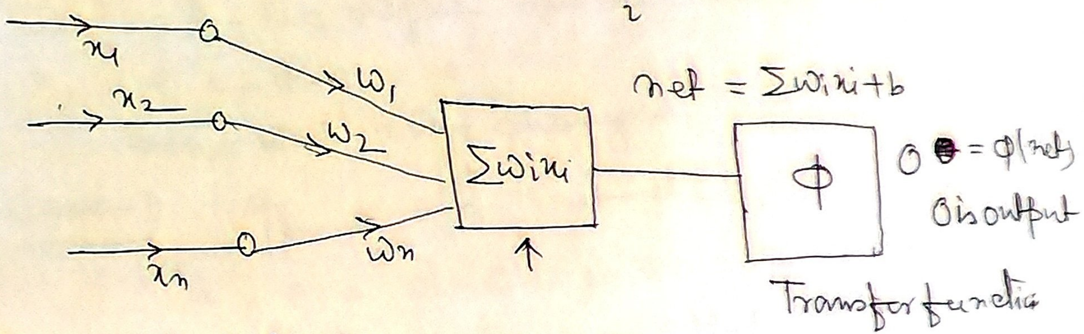
The central idea of neural network is that these parameter can adjusted. so that network exhibits sum desire behavior.

Perceptrons: In 1950s. Frank, Roseblatt and others developed a class of neural network ~~which~~ were called perceptrons. The perceptrons would learn when initialise with random variables for its there

and bias. Generally

Single Layer Perceptron Architecture.

$net = I = sum + b = \sum_i w_i x_i + b.$



Here the transfer function ϕ may be Hard limit function.

$$\phi(net) = 1, \quad net \geq 0$$
$$= 0, \quad net < 0.$$

Perception Learning Rule:

Let d be the target value of the perceptron, O be the output of the present perceptron, e be the error,

$$e = d - O.$$

Now, the objective is to reduce the error e. The perceptron learning rule Calculate the desired changes to the perceptrons weights and bias, given an input vector $\bar{x}$ and associated error e. The target vector d must contain values either '0' or '1' because the perceptrons with hard limit transfer function Can only output either '0' or '1'.

The perceptrons rule improves to Converge

There are three Condition that can recou

Single neuron.

Case-1 If the output of the neuron is

ie, $e = d - 0 = 0$

then w is not changed.

Case-II If $o = 0$ but $d = 1$,

ie, $e = d - 0 = 1 - 0 = 1$.

Then input vector x is added to w.

Case-III If $o = 1$ but $d = 0$.

ie, $e = d - o = 0 - 1 = -1$.

Then input vector x is Subtracted

if weight vector changing rule is

$$W_{new} = W_{old} + \Delta W.$$

Then learning rules for Cases ar

Case-1. If $e = 0$, then make a

Case-II If $e = 1$, " " "

Case-III If $e = -1$, " " "

The alteration (change) rule for

$$b_{new} = b_{old} + e.$$

Ex: Let the classification is like as

$\{X_1^T = [2\ 2],\ d_1 = 0\}$, $\{X_2^T = [1$

$\{X_3^T = [-2, 2],\ d_3 = 0\}$, $\{X_4^T = [-1,$

Single vector

Iteration — 1.

Step-$net_1 = X_1^T W(0) + b(0)$.

$= \begin{bmatrix} 2 & 2 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \end{bmatrix} = 0$

Output $= 0 = \phi(net_1) = \phi(0) =$ hard limit $(0) = 1$.

But target value ~~e~~ $d_1 = 0$.

$\therefore e_{~~0~~} = d_1 - 0 = 0 - 1 = -1$.

$\therefore \Delta W = e x_1 = (-1) \begin{bmatrix} 2 \\ 2 \end{bmatrix} = \begin{bmatrix} -2 \\ -2 \end{bmatrix}$

$\Delta b = e = -1$.

$W_{new} = W_{old} + \Delta W = \begin{bmatrix} 0 \\ 0 \end{bmatrix} + \begin{bmatrix} -2 \\ -2 \end{bmatrix} = \begin{bmatrix} -2 \\ -2 \end{bmatrix}$

$b_{new} = b_{old} + \Delta b = \begin{bmatrix} 0 \end{bmatrix} + \begin{bmatrix} -1 \end{bmatrix} = -1$

Step ~~Iteration~~ Step - 2. $\{ X_2^T = \begin{bmatrix} 1 & -2 \end{bmatrix}, d_2 = 1 \}$.

$net_2 = X_2^T W(1) + b(1) = \begin{bmatrix} 1 & -2 \end{bmatrix} \begin{bmatrix} -2 \\ -2 \end{bmatrix} + \begin{bmatrix} -1 \end{bmatrix}$

Output $= 0 = \phi(net_2) = \phi(1) \overset{!}{=} 1$.

Target $d_2 = 1$.

error $= e = d_2 - 0 = 1 - 1 = 0$.

Hence no change in W and b i.e., $W(2) = \begin{bmatrix} -2 \\ -2 \end{bmatrix}$

$b(2) = -1$.

~~Iteration~~ Step - 3 $\Rightarrow$ $X_3^T = \begin{bmatrix} -2, 2 \end{bmatrix}$, $d_3 = 0$.

$net_3 = X_3^T W(2) + b(2) = \begin{bmatrix} -2 & 2 \end{bmatrix} \begin{bmatrix} -2 \\ -2 \end{bmatrix} + (-1)$

$= -1$

Output $= 0 = \phi(net_3) = 0$.

target value $d_3 = 0$.

error $= d_3 - 0 = 0 - 0 = 0$

Hence no change in weight W and b. as

$W_3[3] = W[2] = \begin{bmatrix} -2 \\ -2 \end{bmatrix}, b(3) = b(2) = -1$

Step - 4. $\{ X_4^T = \begin{bmatrix} -1 & 1 \end{bmatrix}, d_4 = 1 \}$ $\begin{bmatrix} -1 & 1 \end{bmatrix} \begin{bmatrix} -2 \\ -2 \end{bmatrix} + (-1) = -1$

output $= 0 = 0$

But $d_4 = 1$.

error $= d_4 - 0 = 1 - 0 = 1$.

$\Delta w = e x_4 = 1 \cdot \begin{bmatrix} -1 \\ 1 \end{bmatrix} = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$, $\Delta b = e = 1$.

$W_{new} = W_{old} + \Delta w$

$= \begin{bmatrix} -2 \\ -2 \end{bmatrix} + \begin{bmatrix} -1 \\ 1 \end{bmatrix} = \begin{bmatrix} -3 \\ -1 \end{bmatrix} = w(4)$

$b_{new} = b(4) = b_{old} + \Delta b = -1 + 1 = 0$

Iteration-2.

Fuzzy Rule Base and Approximate Reasoning:

1. Truth values and Tables in Fuzzy Logic:-

Fuzzy logic uses linguistic variables. The values of a lingistic variables are words or sentences in a natural or artificial language. For example, height is a linguistic variable if it takes values such as tall, medium, short and so on. The linguistic variable provides approximate characterization of a complex problem. The name of the variable, the universe of discourse and a fuzzy subset of universe of discourse characterized a fuzzy variables A linguistic is a variable of a higher order than a fuzzy variable and its values are taken to be a fuzzy variables. A linguistic variable is charaterized by

(i) name of the variable (x);
(ii) term set of the variable t (x);
(iii) syntactic rule for generating the values of x;
(iv) semantic rule ~~for~~ associating each ~~a~~ value of x with its meaning.

Apart from the linguistic variables, there exists what are called as linguistic hedges (linguistic modifiers). For example, in the fuzzy set "very tall", the word 'very' is a linguistic hedge. A few popular linguistic hedges includes : very, highly, slightly, moderately, plus, minus, fairly, rather.

Reasoning has logic as its basis, whereas propo text sentences expressed in any

Where Z is the symbol of the Subject and F the predicate designing the characteristics of Subject. For example, "London is in Uni Kingdom" is a proposition in which 'Lon is the Subject and 'in United Kingdo is the predicate, which specifies a propert of 'London' ie, its geographical location United Kingdom. Every proposition has it osite, called negation. For assuming oppo truth values, a proposition and its negat are required.

Truth tables define logic functions of two -positions. Let X and Y be two propositions -es of which can be true and false. Y basic logic operations performed over the pro -sitions are the following

1. Conjunction ($\wedge$) : X AND Y
2. Disjunction (V) : X OR Y.
3. Implication or Conditional ($\Rightarrow$) : IF X TH
4. Bidirectional or equivalence ($\Leftrightarrow$) : X AND ~~OR~~ ONLY IF Y.

On the basis of these operations on proposi inference rules can be formulated. Few info rules are as

$$[X \wedge (X \Rightarrow Y)] \Rightarrow Y$$

$$[\overline{Y} \wedge (X \Rightarrow Y)] \Rightarrow \overline{X}$$

The above rules produce certain propositions that are always ~~are at~~ true irrespective of the truth values of propositions X and Y. Such propositions are called tautologies. An extension of set-theoretic bivalence logic is the fuzzy logic where the truth values are terms of the linguistic variable 'truth'.

The truth values of propositions in fuzzy logic are allowed to range over the unit interval [0,1]. A truth value in fuzzy logic 'very true' may be interpreted as a fuzzy set in [0,1]. The truth value of the proposition "Z is A" or simply the truth value of A, denoted ~~value~~ by tv(A) is defined by a point in [0,1] (called the numerical truth value) or a fuzzy set in [0,1] (called the linguistic truth value)

The truth value of a proposition can be obtained from the logic operations of other proposition whose truth value are known. If tv(X) and tv(Y) are numerical ~~val~~ truth values of propositions X and Y, respectively,

then $tv(X \text{ AND } Y) = tv(X) \wedge tv(Y) = \min[tv(X), tv(Y)]$

$tv(X \text{ or } Y) = tv(X) \vee tv(Y) = \max[tv(X), tv(Y)]$

$tv(\text{NOT } X) = 1 - tv(X)$

$tv(X \Rightarrow Y) = tv(X) \Rightarrow tv(B) = \max[1 - tv(X), \min[tv(X), \cancel{tv}$

Fuzzy propositions: For extendin
-bility, fuzzy logic uses fu
fuzzy-predicate modifiers,
~~in the fuzzy~~ and fuzzy qua
propositions.
The fuzzy propositions make
differ from classical logic.
-sitions are
① Fuzzy predicates: In fuzzy l
Can be fuzzy, for example
Hence, we have proposition li
It is obvious that most of the
ral language a fuzzy rather
(ii) Fuzzy-predicate modifiers: I
, there exists a wide range
modifiers that act as hedges,
very fairly, moderately, rather
predicate modifiers are nece
ating the values of a linguisti
ple - "climate is moderately Co
rately' is the fuzzy predicate.
(iii) Fuzzy quantifiers: The fuz
Such as most, several, many
used in fuzzy logic. Fuzzy

Fuzzy qualifiers:- There are four modes of qualification in fuzzy logic.

* Fuzzy truth qualification: It is expressed as " x is τ." in which τ is a fuzzy truth value. A fuzzy truth value claims the degree of truth of fuzzy propositions. Ex: (Paul is Young) is ~~not~~ NOT VERY True.

Here the qualified proposition is (Paul is Young) and the qualifying fuzzy truth value is " NOT Very True".

* Fuzzy probability qualification: It is denoted as " x is λ" where λ is fuzzy probability. In conventional logic, probability is either numerical or an interval. In fuzzy logic, fuzzy probability is expressed by terms such as likely, very likely, unlikely, around and som.

Ex: (Paul is Young) is Likely.

Here the qualifying fuzzy probability is " Likely". These probabilities may be interpreted as fuzzy numbers, which may be manipulated using fuzzy arithmetic.

* Fuzzy possibility qualification: It is expressed as " x is π," where π is a fuzzy possibility and can be of the following forms: possible, quite possible, almost possible.

* Fuzzy usuality qualification:- It is expressed as " usually (X) = usually $(X$ is $F)$," in which the subject X is a variable taking values in a universe of discourse U and the predicate

F is a fuzzy subset of U and interpreted a Com a usual value of X denoted by $U(X) = F$. The propositions that are usually true or the events that have high probability of occurance are related by the concept of usuality qualification.

Formation of Rules: The general way of representing human Knowledge is by forming natural language expressions given by

IF antecedent THEN Consequent.

The above expression is referred to as the IF-THEN rule-based form. There are three general forms that exist for any linguistic variable. They are ⓘ assignments statements; ⓘⓘ Conditional statements; ⓘⓘⓘ unconditional statements.

The Canonical form of fuzzy rule-based system.

Rule 1 : If Condition $\underset{\sim}{C_1}$, THEN restriction $\underset{\sim}{R_1}$

Rule 2 : If Condition $\underset{\sim}{C_2}$, THEN restriction $\underset{\sim}{R_2}$

— — — —

Rule n: If Condition $\underset{\sim}{C_n}$, THEN restriction $\underset{\sim}{R_n}$.

1. Assignment statements:-

y = Small.

Orange Colour = Oragne.

$a = 8$.

The statement '=' for assignment.

ol THEN Stop;

ents use the 'IF-THEN'

ent:

es:

a Collection of many
d together. Any Com-
ture may be decompos-
, a number of Simple
ms. The rules are gene-
atural language represen-

antecedents:

e $, \underset{\sim}{A}_n$ THEN y is $\underset{\sim}{B}_m$

$\cap \underset{\sim}{A}_n$

$\underset{\sim}{A}_1 \cap \underset{\sim}{A}_2 \cap \cdots \cap \underset{\sim}{A}_n$

in $[\mu_{\underset{\sim}{A}_1}(x), \mu_{\underset{\sim}{A}_2}(x), \cdots \mu_{\underset{\sim}{A}_n}(x)]$

THEN $\underset{\sim}{B}_m$.

antecedents,

THEN

Conditional Statements (With ELSE and UNLESS)

IF $\underset{\sim}{A_1}$ THEN ($\underset{\sim}{B_1}$ ELSE $\underset{\sim}{B_2}$)

It is decomposed.

If $\underset{\sim}{A_1}$ THEN $\underset{\sim}{B_1}$

OR

If NOT $\underset{\sim}{A_1}$ THEN $\underset{\sim}{B_2}$

If $\underset{\sim}{A_1}$ (THEN $\underset{\sim}{B_1}$) UNLESS $\underset{\sim}{A_2}$.

Can be decomposed as

IF $\underset{\sim}{A_1}$ THEN $\underset{\sim}{B_1}$

OR

IF $\underset{\sim}{A_2}$ THEN NOT $\underset{\sim}{B_1}$

IF $\underset{\sim}{A_1}$ THEN ($\underset{\sim}{B_1}$) ELSE IF $\underset{\sim}{A_2}$ THEN ($\underset{\sim}{B_2}$).

Can be decomposed into the form.

IF $\underset{\sim}{A_1}$ THEN $\underset{\sim}{B_1}$

OR

IF NOT $\underset{\sim}{A_1}$ AND IF $\underset{\sim}{A_2}$ THEN $\underset{\sim}{B_2}$.

Nested IF - THEN rules

The rule "IF $\underset{\sim}{A_1}$ THEN [IF $\underset{\sim}{A_2}$ THEN ($\underset{\sim}{B_1}$)]" can be of the form

IF $\underset{\sim}{A_1}$ AND $\underset{\sim}{A_2}$ THEN $\underset{\sim}{B_1}$.