AI

DFS with depth limit L

L> DFS Iterative Deeping Search

uniform cost search Algorith

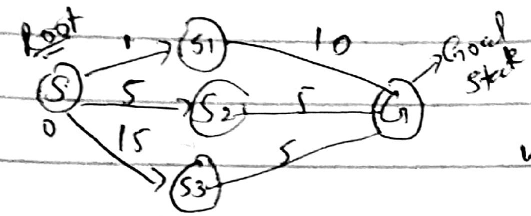
L UCS

Priority : g(n)

							0

queue

g(n) :- is the cost from root node to nth root.

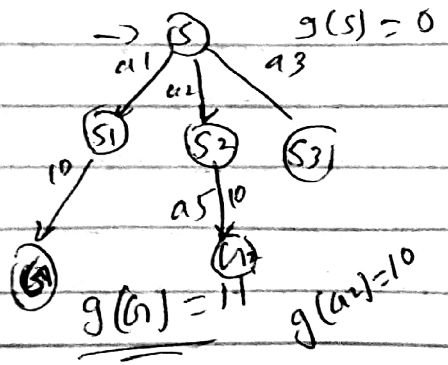


we want Rational Agent!

Priority: g(n)

S	S_1	S2	S3				
0	1	5	15				

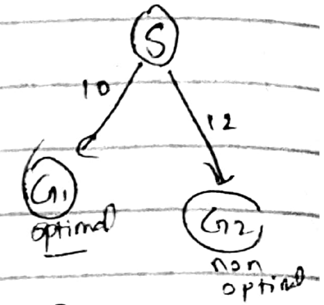
possible Action



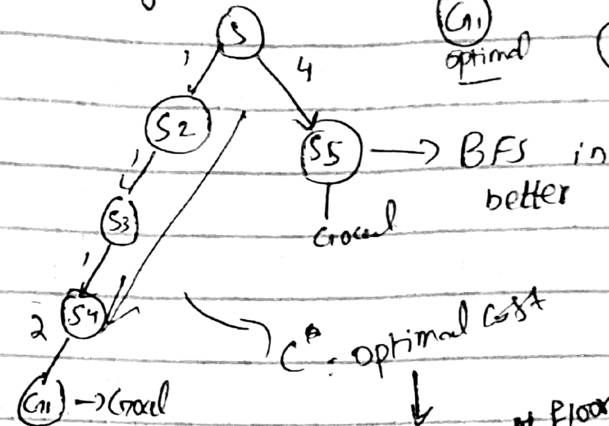
Completness

L> If all path have cost > 0

optimality:



Time complexity = UCS

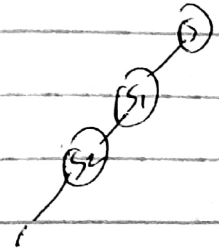


Time Complexity of UCS:

ε -> step cost

$O(b^{1+floor(C^*/\varepsilon)})$

expand those node this can sometime be worst than BFS



AI → 14-April

Chapter#03

↳ Informed Searches

Best Algorithms

↳ You have heuristics

↳ Greedy best first search

↳ A^*

↳ RBFS

↳ SMA^*

↳ limitation of uninformed searches.

↳ 8-Puzzle game

↳ branch factor → 3 (Avg)

↳ depth = 22

$3^{22} = 3.1 \times 10^{22}$ → states

↳ using BFS

IDS → 3.6 Million states

10^{24} → state → worst case.

↳ 100 years

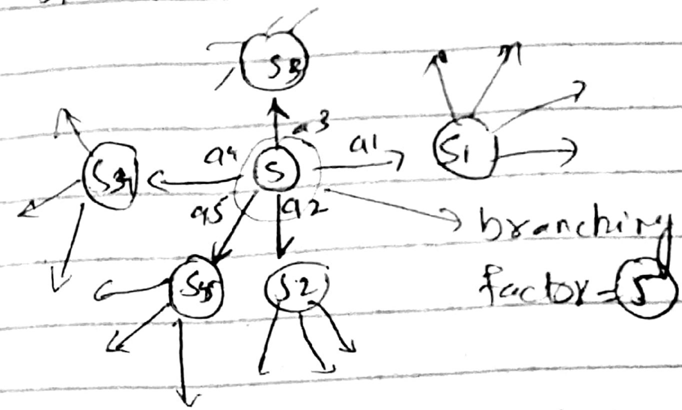
Tree

↳ pick node from queue base on the heuristics.

AI

→ BFS, DFS, IDS, UNF

Diff b/w informed / uninformed.



Informed → know the where is goal from heuristics function

↳ estimate where your goal.

↳ reduce the search spaces.

Best-first Search

↳ special case

↳ queue → fifo, lifo, g(n) f(n)

special case

↳ uniform cost search

↳ greedy

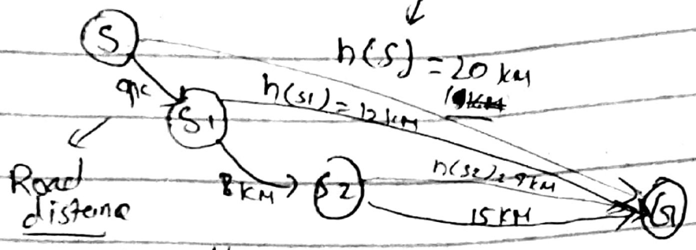
↳ A^*

A I

Heuristic function / Additional Information
estimate.

↳ give us Problem to understand

→ A h(n) is an estimate of goal node from the node n

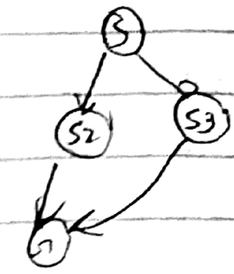


h(n) : stright line distance

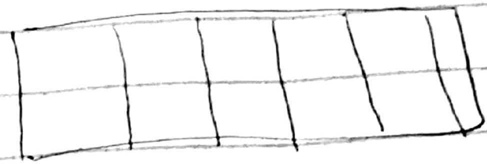
$h_2(n)$ = Aeroplane distance.
$h_3(n)$ = boat distance
$h_4(n)$ = Rail distence
$h_5(n)$ = walk distance

A I

Greedy best-first Search



f(n) = h(n)

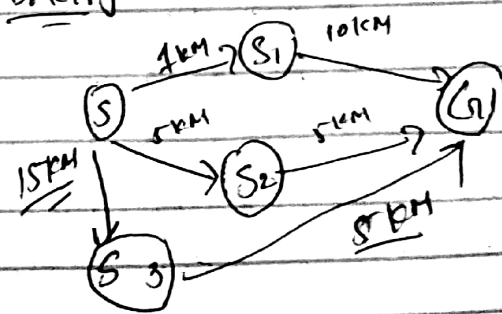


Romania distance problem
goal = Bachart
↳ Aminaction on Slides

Properties based on (h(n))
↳ false → comple
↳ false → false
↳ Time → exponential
↳ Space → exponential.

A* Search
↳ total path cost

Working



AI

$h(s)$ = straight line distances

$h(s) = 8KM$ $\quad h(s3) = 3KM$

$h(s1) = 7KM$ $\quad h(G) = 0$

$h(s2) = 4KM$

s	s1	s2	s3	
8	8	9	18	

$f(n)$

$f(n) = g(n) + h(n)$

$f(s) = g(s) + h(s)$

$= 0 + 8 = 8KM$

$f(s1) = 1KM + 7KM = 8km$

$f(s2) = 5KM + 4KM = 9KM$

$f(s3) = 15KM + 3KM = 18KM$

$f(G) = 11KM + 0 = 11KM$

$f(2) = 10 + 0 = 10KM$

AI

16 April

↳ question

Admissible heuristics

↳ $h(n) \le h^*(n)$

↳ Actual cost

A* optimality proof

→ For Admissible Heuristics

i.e $h(n) \le h^*(n)$

A* will be optimal

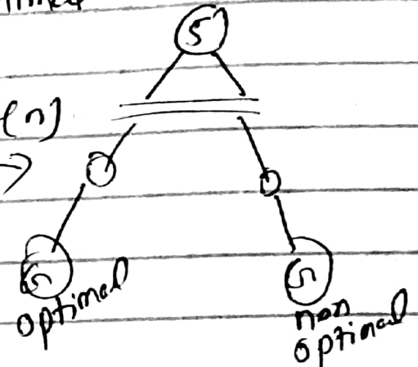
$f(G_2) > f(G) > f(n)$

$g(h_2) + h(h_2)$

$g(h_3) + h(h_3)$

↓

0



→ $h(n) \le h^*(n)$

$h(n) + g(n) \le h^*(n) + g(n)$

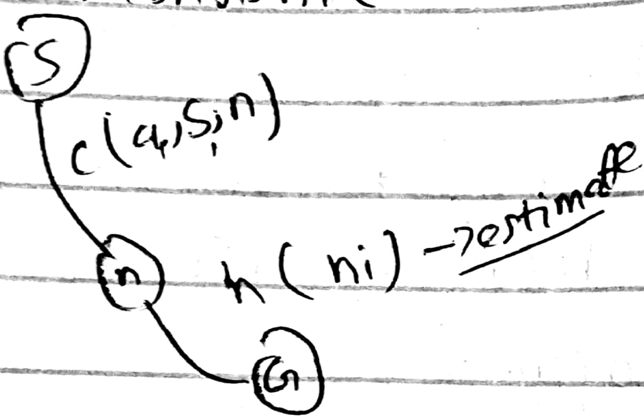
$f(n) \le g(n) + 0$

$f(n) < f(G) < f(G_2)$

$f(G_1) > f(G) > f(n)$

Optimality in graph of A^*

↳ heuristic should be Monotonic



$h(n_i) \leq h(s) + c$

AI

Recursive Best first Search (RBFS)

Problem with A^*
↳ memory

→ To solve memory issue we have RbFS

$f(n) = g(n) + h(n)$ → Same As A^*

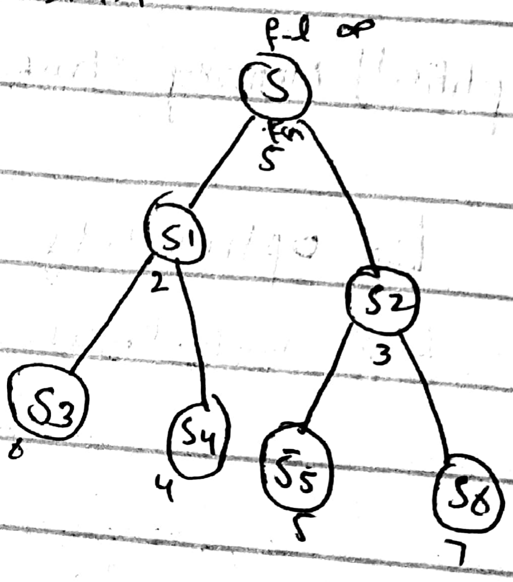
If $f(n) > f\text{-limit}$
If current node exceeds f-limit

queue: S, S1, ∞

S1 2, S2 3

S 4, S2 3, S5 5, S6 7

3



How Algo run?

Key points
↳ f-limit variable
↳ Recursion

} search one depth at a time

A^* $f(n) = g(n) + h(n)$

When you backtrack??
When your f(n) is smaller then all the fn Node & their child.

AI

Properties

optimal = Yes
Tree → admissible
graph → consistent

Complet → Yes
↳ If finite cost solution

time complexity.
↳ depend distribution cost

(Simplified) Memory bound A* (SMA*)

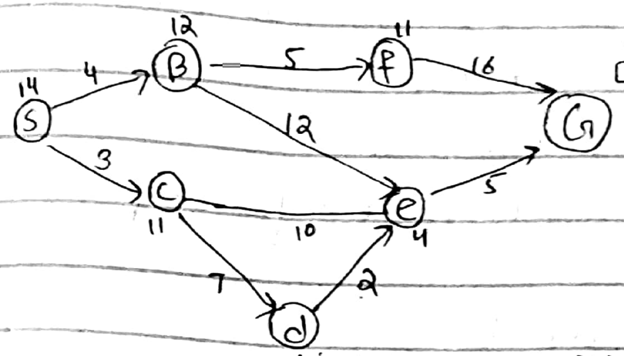
↳ optimality True
↳ completness → True
false in one condition

Design heuristic function

AI

AI

A* Algorithm → Informed Searches
Admisible Algorithm
↳ used heuristics
↳ $f(n) = g(n) + h(n)$
↳ Actual cost From start Node to n
↳ Estimate cost From n to Goal node



$f(S) = 0 + 14 = 14$

$F(S)$ — $f(n)=16$ B
— $f(n)=14$ C
↳ 3+11

SE $\xrightarrow{14}$ e → SCd
13 d

Scd → e

Scde →

Scde → G

$f(S) = 0 + 14 = f(S) = 14$

S → B
S → c

$f(SB) = 16$
$f(SC) = 14$
$f(SCe) = 17$
$f(SCd) = 16$

S 3 C 10 e
7 d

$f(SB) = f(Scd) = 16$
explore both

$f(SB)$ — B — f $= f(SBf) = 20$
B — e $= f(SBE) = 20$

$f(Scd) = 16$
$f(ScdeG) = 17$
$f(ScdeG) = 18$
$f($

How to make A^* admissible:

Case I

↳ $h(n) \le h^*(n)$ → underestimate

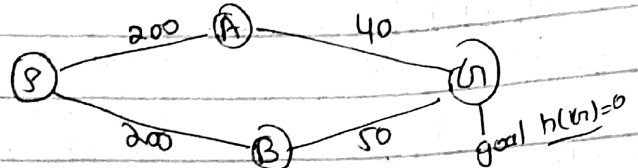
↳ $h(n) \ge h^*(n)$ → overestimate

$h(n)$ → estimate cost

$h^*(n)$ → Actual cost

example → Laptop buy from Shop.

Case I overestimate

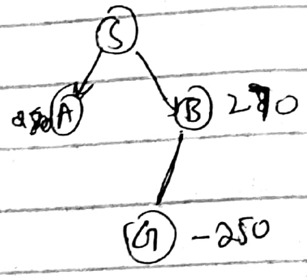


$h(A) = 80$

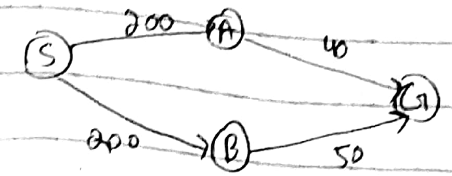
$h(B) = 70$] $> h^*$

$f(A) = 200 + 80 = 280$

$f(B) = 200 + 70 = 270$



Case II underestimate



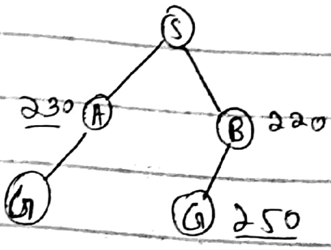
$h(A) = 30$

$h(B) = 20$

$f(A) = g(A) + h(A)$

$f(A) = 200 + 30 = 230$

$f(B) = 200 + 20 = 220$



$F(G) = 250 + 0$

$f(G) = 250$

$f(SAG) = 200 + 40$

$= 240$

Entry

AI

~~Reading Article~~

AI Genetic Algorithm

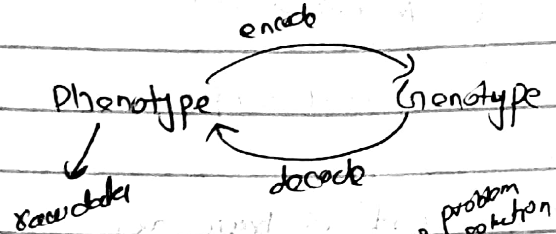
↳ Abstraction

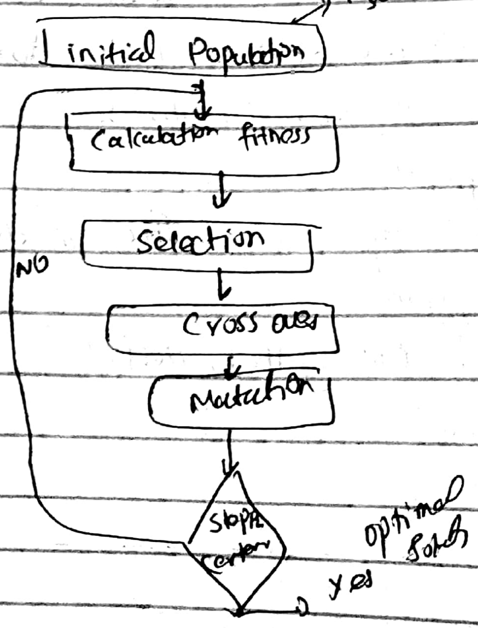
↳ real connection of genetic Algorithm

↳ we have all solution of Problem

↳ used where search space is very large.

↳ choose the best





AI

Make → Take → fitness value → 3

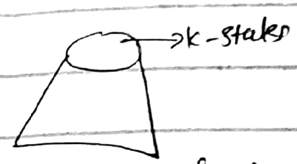
Abcd → fitness 1

↳ How much similar with target

local beam Search

↳ why we call local beam

↳ because search beam of light.



Idea → beam of light

↳ parallel search Algorithm

↳ start from Multiple state

- keep track k states instead one
  ↳ contraint by memory (RAM)
- initially : k randomly selected states.
- Next : determine all successor of k states
  ↳ $\boxed{K \times b}$ → pool of states
- If any of successor goal → finished.
- else select k best from All successor and repeat.

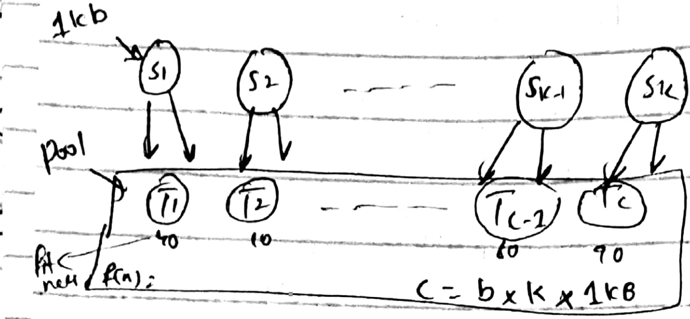
AI

→ Similar with random-restart search (X)

Local with pool

- Can suffer lack of diversity
  - ↳ at the local Maxima
  - ↳ Stochistic beam search

~~At~~ each



How we find value of k??

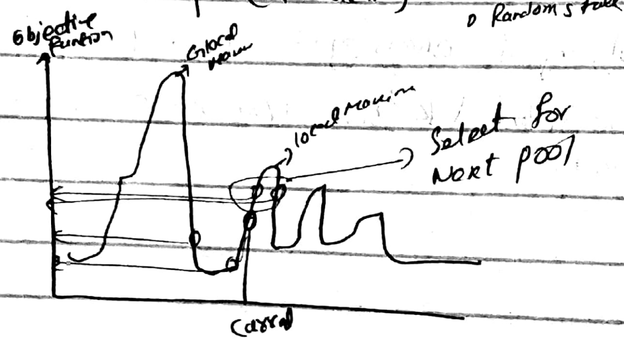
↳ depend upon the RAM

b → may be vary of each Node

↳ Choose Max b

Search landscape (Problem) → local beam search

o Random start

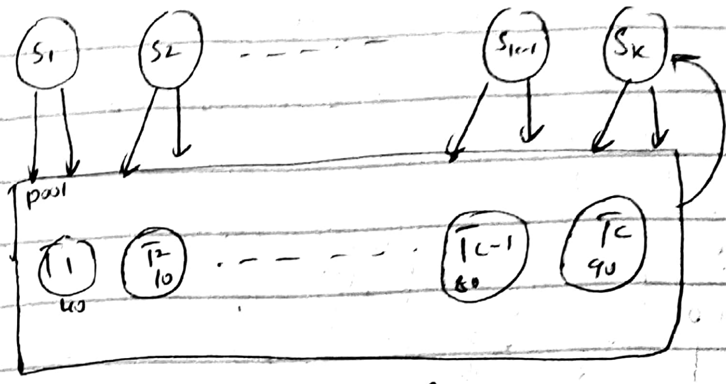


→ To avoid the problem in local maxima with help of randomness

Stochastic Local beam Search

k → depend upon the memory

b = 2



- check Tc is goal that fine
- else → create new pool
  - ↳ but with randomness instead Picking best Two
  - ↳ calculate the probility

$k = 2$

$b = 2$

$c = k \times b = 4$

How calculate Prob

$$P_i = \frac{\text{fitness}(T_i)}{\sum_{i=1}^{c} \text{fitness}(T_i)}$$

↳ fitness of pool

$$T_1 = \frac{40}{40+10+60+90} = \frac{40}{200} = \frac{1}{5} = 0.20$$

28 Minute (N.T)

$P_{T_2} = \frac{10}{200} = \frac{1}{20} = 0.05$

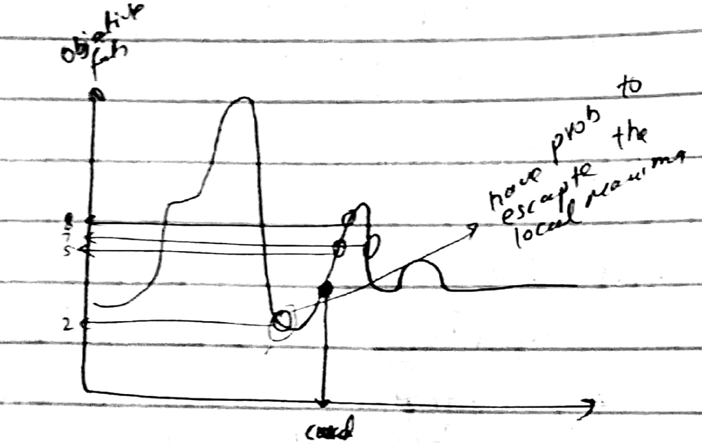
$P_{T_3} = \frac{64}{200} = \frac{6}{20} = 0.30$

$P_{T_4} = \frac{94}{200} = \frac{9}{20} = 0.45$

Implementation
Random (1, 100)

1 more to select T1	more sel to T2	more select to T3	select T4
20	20 5 25	30 55	45
3	4	2	1

Stochastic beam search



8+cd+01

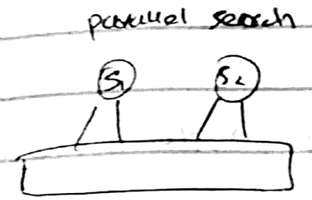
↑ Iteration → Converge ↑ → optimum value

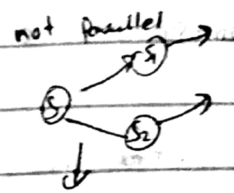
Black box

Genetic Algorithm (optimization)
↳ inspired by biological evolution
↳ Human → gens
↳ evolved with passage of time

Adaptive processes

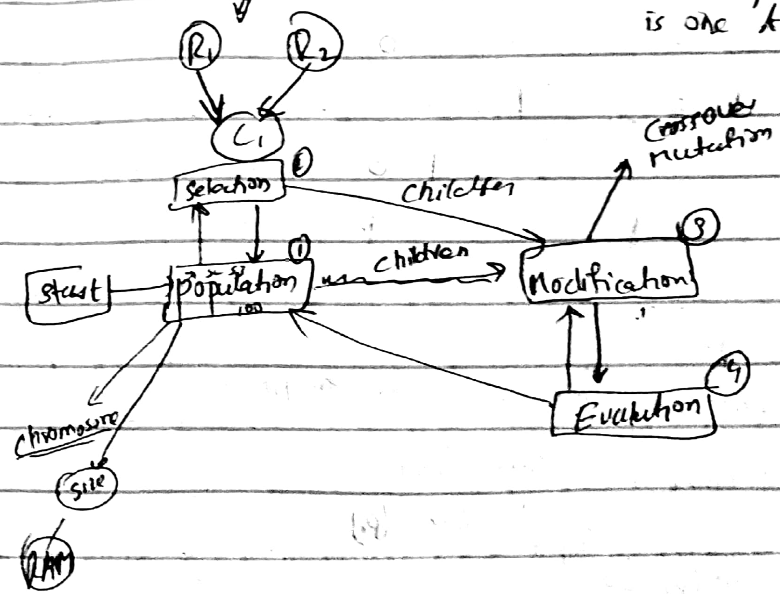
Steps of genetic Algorithm (Parallel Search Algo)





sexual Representation

Asexual representation
↳ child generate is one state)



AI

Artificial Intell (AI)

Problem

Image Processing

black Image → white Image

0[1]	0[2]	0[3]	→	1	1	1
0[4]	0[5]	0[6]		1	1	1
0[7]	0[8]	0[9]		1	1	1

Chromosone 1D = 0 0 0 0 0 0 0 0 0

"every chromosone is the solution of your Problem"

① initial popution (Randomly selecting parent = 1) — choose fitness

		fitness
P1	0 0 0 0 0 0 0 0 0	0
P2	0 1 0 0 0 1 0 0 0	2
P3	0 0 0 0 0 0 0 1 0	1
P4	1 0 0 1 0 0 0 0 1	3

② Selection

selection is randomly and is based on fitness

$$\text{Select-Prob}_i = \frac{\text{fitness}(p_i)}{\sum_{j=1}^{\text{Pusize}} \text{fitness}(p_j)}$$

Select-Prob $(P_1) = 0$

Select-Prob $(P_3) = 1/6$

Select Prob $(P_2) = 2/6$

Select-Prob $(P_4) = 3/6$

Select-Prob $(P_1) = 0$

Implementation

P2 P3 P4

① Modification [Cross over]

↳ control the convergess of your Algorithm

P2	0 1 0	0 0 1 0 0 0	↓ swap
P3	0 0 0	0 0 0 0 1 0	↑

C1 0 1 0 0 0 0 0 1 0

C2 0 0 0 0 0 1 0 0 0

↳ children must be valid state

e.g ⇒ 2nd parent and last parent not found

↳ then this non valid state.

After cross over

C1 ______

C2 ______ → Mutation

C3 ______

C4 ______

Section L-A

AI

Cross over → explore Local Maximum

Mutation → explore and change the state of the area on Local Maximum

① Evalution

Check the current state is goal state or not

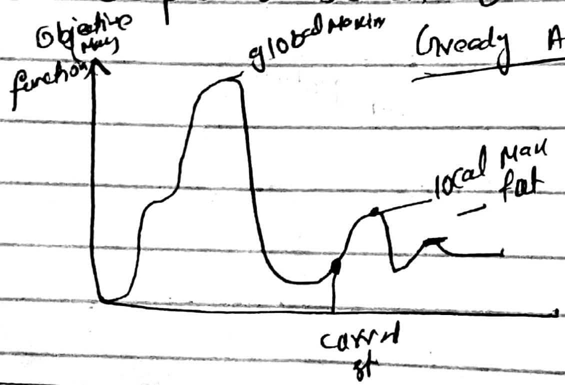
Application of genetic Algorithm
↳ feature selection
↳ Engineering Design

Local Search & optimization Problems

① Hill climbing
② Simulated Anealing

Landscape of Search 2D

Greedy Approach



AI

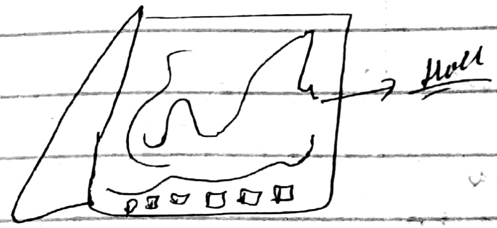
→ to get local Max we will use all the Randomize

Simulated Annealing Search
↓
cooling process
↳ physical process

Physical Analgy:
↳ Physical process

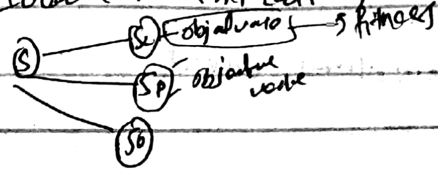
Anealing
↳ cooling process
↳ Heat up → motion particle → Randomly move
↓
remove Heat
↓
slowly cool
stable state ← energy loss particle ← slowly cool

Control Temperature

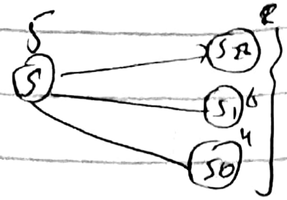


Search using Simulated Anealing (2)

→ basic Idea (Maximization)



AI



Select Randomly

Calculate Δ

$\Delta = f(S_2) - f(S) = 8-5 = 3$ ✓ move Step

else

$\Delta = f(S_3) - f(S) = 3-4 = -1$ ✓

$P_{S_3} = e^{\Delta E / T}$

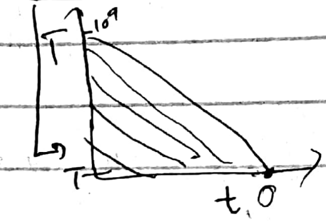
↓ prob ↓ energy

time

$T = $ schedule (t)

$\boxed{T \propto 1/t}$

function SA (problem, Schedule)



local variable ✓

L current and

next

T → Temperature

t → time

P_K =

for t ← 1 to ∞

if T = schedule(t)

if T = 0 return current

AI

$P_K = e^{\Delta E_K / T}$

↳ for maximization Problem ΔE is negative

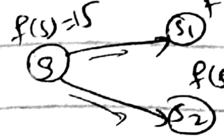
$P_K \propto T$ → Directly proportional

$P_K \propto \frac{1}{\Delta E_K}$ → Indirectly prop

'crow effect

$P_K = e^{\Delta E_K / T}$

let $T = 1$;



Want Maximum

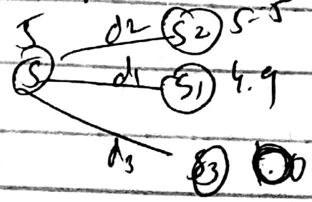
$P_K = e^{\Delta E_K / T}$

$P_{S_1} = e^{-5/1} = 1/e^5$ $\quad \Delta E_{S_1} = 10 - 15 = -5$

$P_{S_2} = e^{-10/1} = 1/e^{10}$ $\quad \Delta E_{S_2} = 5 - 10 = -10$

$\frac{1}{e^{10}} << \frac{1}{e^5}$; $e = 2.71$

example



(A.I)

By some Problem become stone slow
↳ find the value of T and
and How we decrease currently
Researching stating.

Practicle example:

Start

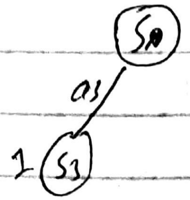
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

f(start) = 0

goal state

| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

0 0 0 0 0 0 0 0 0 0 → Pick a3



AI Chapter = 4

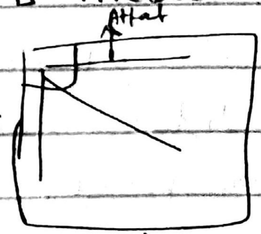
Local Search and optimization

1. Hill - Climbing
2. Simulated Annealing ←
3. Local beam Search ←
4. Stochastic beam Search ←
5. Genetic algorithm ←

Hill Climbing

why local search and optimization??
↳ previous Algo
↳ Systematically explore the Search space.
↳ path to goal is solution to problem.

→ Some Problem → path is irrelevant
↳ 8 - queens
↳ goal no one Attack

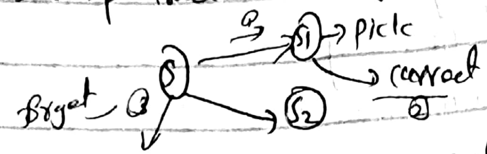


↳ factory floor layout
↳ Increase throughput
↳ Automatic Programming
↳ Integrated circuit design
↳ Vehicle routing.

AI

Local search and optimization.

- Local Search
  - ↳ keep track of single current state



no need to save passed node

  - ↳ Move only to neighbour state
  - ↳ Ignore path.

→ Advantage
  - ↳ utilize memory (little)

- Pure optimization Problems:
  - ↳ All state have an objective function
    - ↳ value.
  - ↳ gola → state find max or min
  - ↳ some problem do not quit fit into path-cost / goal-state Into.

nature provide
  - ↳ reproduce fitness

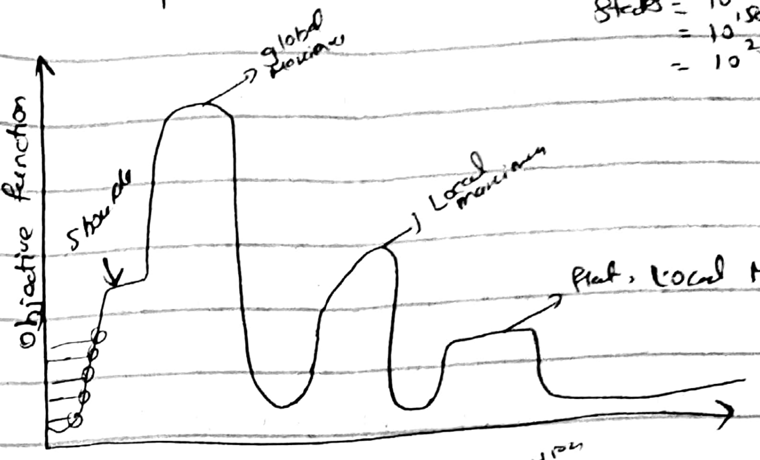
↳

AI

Landscape of Search

o → state

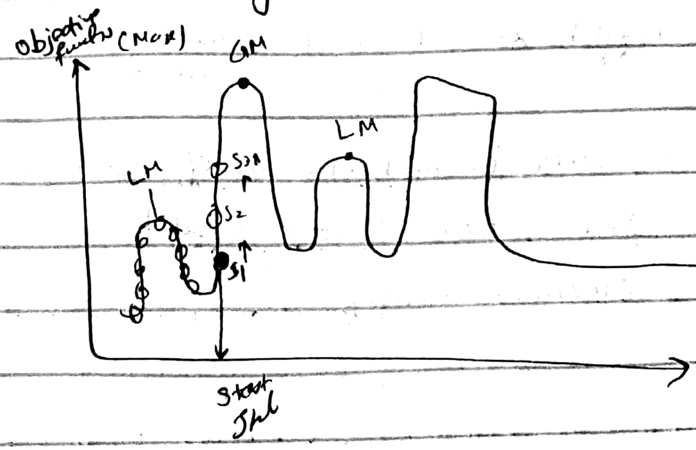
States = 10^{50}
= 10^{150}
= 10^{250}



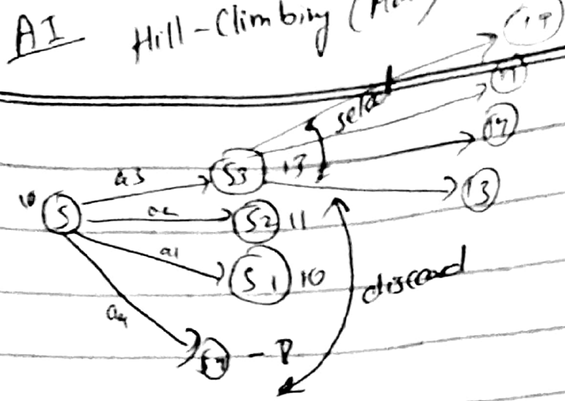
2D Representation → Dimension

↳ for simplicity → 2D Representation.

→ Hill - Climbing Search (Greedy Approach)
  - ↳ Always walk Increasing
- Hill climbing → randomly successor → best



AI Hill-Climbing (Max)



Algorithm

Function Hill-Climbing (Problem) return state

Input : Problem

Local variable : current → node

neighbour → node

current ← Make-Node(initial state)

loop do

neighbour ← a highest valued successor of current

If value [neighbour] ≤ value(current)

then return state [current]

current ← neighbour

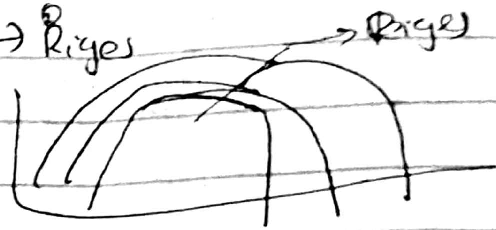
AI AI 2+2+4+5

Hill climbing example → 8 queen

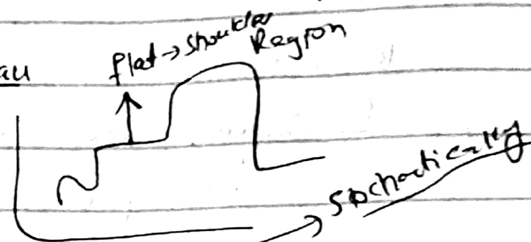
↳ Problem with Hill Climbing

↳ Local Maximum stuck

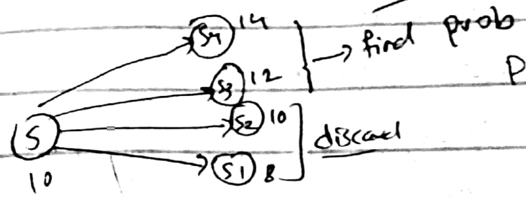
↳ Ridges



↳ Plateau



stochastically

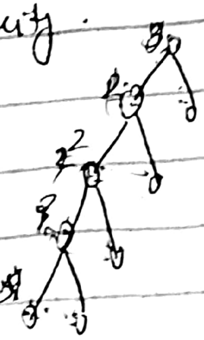


$P_3 = \frac{12}{44}$

$= \frac{14}{14+12+10+8}$

Prob = $\frac{f(S_3)}{\text{Total Fitness of up hills.}}$

DFS Time complexity.



$2^5 2$

$2 \times 4 = 8$

$(2 \times 4) + 2$

$8 + 10$

comparing IDS and BFS

IDS → b = 10 d = 5

L

$(10 \times 5) + (4) 10^2 + (3) 10^3$

Uniform cost Search

Bfs
↳ fif

Dfs
↳ LiB

limit-dept
↳ Restrict the limit

IDS
↳ increase limit if not find in the given limit

$C^* \rightarrow$ optimal cost

$O(b^{[1 + floor(C^*/\epsilon)]})$

Bi-direction Search

↳ forward and backward

start → goal

↳ goal → start

when Applicable??

IDS
↳ variable cost nahi deta

IDS → optimal
non-decreasing
function

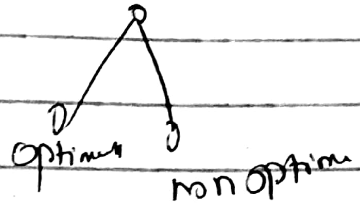
Uniform cost Search

If you reached at the goal but there are some node in queue whose has less cost then goal then UCS will never return the goal

why uniform cost search??

↳ because it unify the path cost of the node.

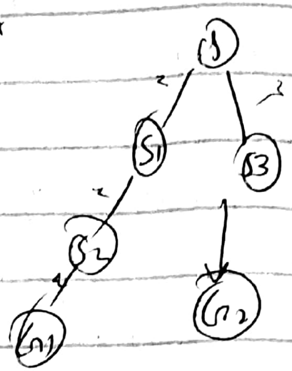
proof uniform cost search is optimum??



Time complexity:

$\epsilon \rightarrow$ Step cost — optiminal cost

$O(b^{1+C^*/\epsilon})$



Bi-Direction Search

forward — backward Search How

we can start from start state to goal state and vice versa. but there are lots of difficulty, It is not easy to back from goal state to start state.

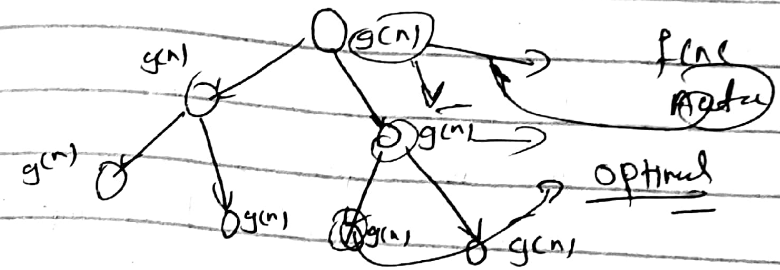
Best - First Search Algorithm

↳ Idea In Greedy Best First Search

Idea

↳ use estimation cost plus Actual cost. select that node which is closet to the Goal Node.

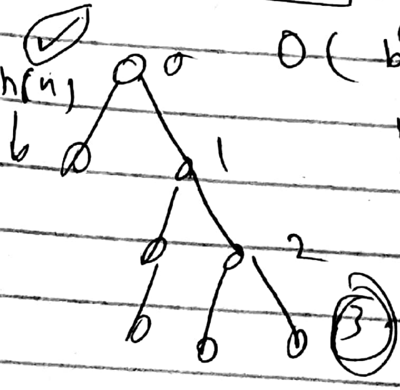
↳ But not give us the Actual cost.



→ Best - first - Search - Algorithm

↳ Can stude in infinete when visit the visited Node.

$f(n) = g(n) + h(n)$



$O(b^m)$

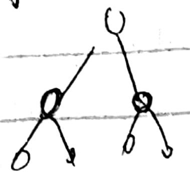
m → Max dept

2> A* Search Algorithm.

$f(n) = g(n) + h(n)$

↳ heuristic

$h(n) \rightarrow$ always be Admissible



) 9

$f(n)$ c*

RBFS

Simulated Annealing

If Temperature is High:

↳ Pro(Locally bad move) is High

If Temperature is Low:

↳ Prob(Locally bad move) is Low

T ∝ Algorithm (run)

T decrease Algorithm will longer run

Local Searches and optimization Problems

↳ not relevent for path.

↳ Local Search can help here.

↳ Hill Climbing

Simulated Annealing ↓ } Maximization problem solve hoti hai

↳ track → current state

Move only neighbour node

$P_k = e^{\frac{-E_k}{T}}$

If T = 50

$P_k = e^{\frac{\Delta E_k}{50}}$

If, T = -50

$P_k = e^{\frac{\Delta E_k}{-50}}$

$\boxed{P_k = \frac{1}{e^{\frac{E_k}{50}}}}$

$d_1 = -0.1$ $d_3 = -5$

$d_2 = 0.5$

T = 1

d_2 = Picked

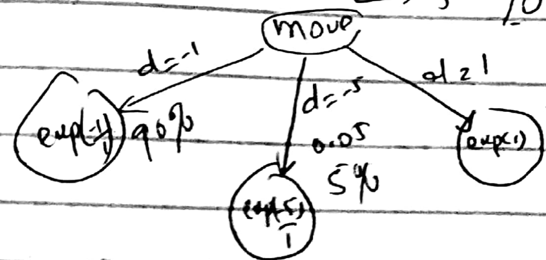
↳ move there

If d_1 and or d_3 = picked

$\exp\left(\frac{-0.1}{1}\right) = 0.9$, 90% accept this move

$mov_3 = \exp(-5) = 0.05$

↳ 5% will accept this

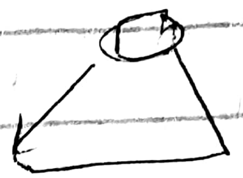


↳ move there

AI

③ Local beam Search

↳ Parallel Search



- keep k state (track)
- determine all successor of k state (branching factor) (k × b)

; else k best All successor and repeat

↳ Stuck in Local Maxima

pool S1 S2 Sk−1 Sk

T1 T2 Tc−1 Tc

70 10 60 90

c = b × k

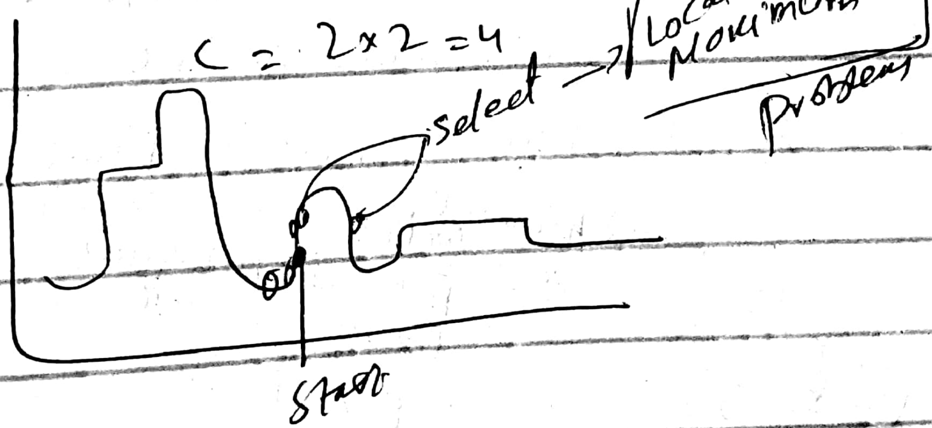
↓

ram depend

Search Landscape (Problem)

k = 2 b = 2

c = 2 × 2 = 4



AI

Stochastic Local Beam search

$P_i = \frac{\text{Fitness}(T_i)}{\text{Total fitness}}$

$PT_1 = \frac{40}{200} = 0.20$ $PT_2 = \frac{1l}{200}$

$PT_{3-1} = \frac{60}{200}$ $PT_8 = \frac{90}{200}$

select T_1	select T_2	select T_3	select T_4
20%	25%	55%	45%

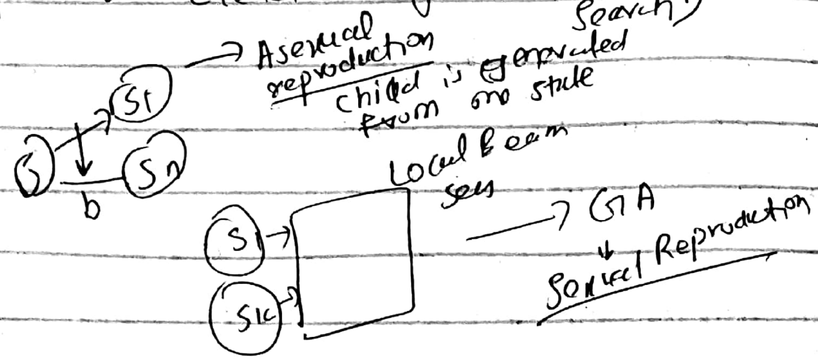
Genetic Algorithm

↳ biological evolution

↳ with the passage of time

nature evolve

Steps of Genetic Algorithm (parallel search)



Sexual Reproduction

Child created from two state like real wife and husband.

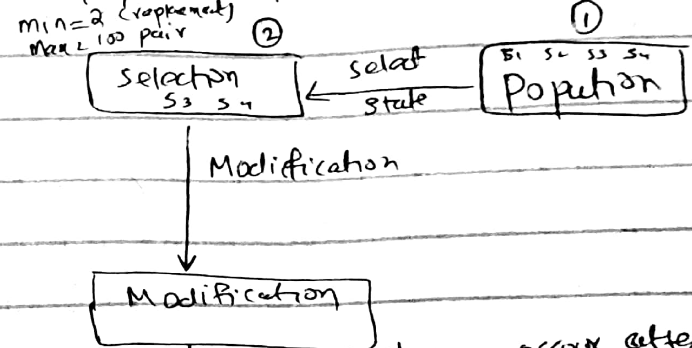
Population (state)

↳ Chromosome hoty thai

Population size constraints → memory dependance

average population = 100 (enough)

min = 2 (replacement)
Max = 100 pair



↳ Cross over changes occur after this

↳ Mutation changes occur after this

the process start until we can achieve global Maxima.

AI

every chromosome is the solution to your problem

Problem

0	0	0		1	1	1
0	0	0	→	1	1	1
0	0	0		1	1	1
start					goal	

ID

Chromosome 0 0 0 0 0 0 0 0 0

fitness → the number of 1's

Fitne (start) = 0   fitness (goal) = 9

① initial population

Randomly chrome some   chrome some fitness

P1	0	0	0	0	0	0	0	0	0	fitness (P1) = 0
P2	0	1	0	0	0	1	0	0	0	fitness (P2) 2
P3	0	0	0	0	0	0	0	1	0	fitness (P3) = 1
P4	1	0	0	1	0	0	0	0	1	fitness (P4) 3

② selection

↳ Probability = $\frac{\text{Fitness (Pi)}}{\text{Total fitness}}$

Prob (P1) = $\frac{0}{1+2+3}$ = 0   Prob (P2) = 2/6

Prob (P3) = 1/6 = 0.33   Prob (P4) = 3/6 = 0.5

Implemen

P2 P3 P4

2 3 6

Suppose we select two ~~the~~ parent using max (prob) (Modification) → cross over

↳ control the convergence of the Algorithm → lead to the solution → global Maximum

P2 = 0 1 0 | 0 0 1 0 0 0

P3 = 0 0 0 | 0 0 0 1 0

⇓ cross over

Children be valid child (valid states)

C1 = 0 1 0 0 0 0 1 0

C2 = 0 0 0 0 0 1 0 0 0

P4 = 1 0 0 1 0 0 0 0 1

P3 = 0 0 0 0 0 0 0 0 1

↓ Cross over

C3 = 1 0 0 1 0 0 0 0 1

C4 = 0 0 0 0 0 0 0 0 1

AI

→ Controls the exploration

Mutation

↳ control

↳ ~~converge~~ explore karwata Hai'.

	After Mutation
C1 = 0100000010	C1 = 0100100010
C2 = 0000001000	C2 = 0000001000
C3 = 100100010 ⇒	C3 = 100100110
C4 = 0000000001	C4 = 0100000011

we change chromosome randomly using predefined probability = 0.01

(4) Evaluation

~~C1~~ fitness

f(c1) = 3

f(c2) = 1

f(c3) = (3)

f(c4) = 2

↳ check children in goal

else

Children are replaced

existing children → become parent

then create new population

5	Q	0	0	0	Q
4	0	0	Q	0	0
3	0	0	0	Q	0
2	0	Q	0	0	0
1	0	0	0	0	0

(1)

Chromosome = [5, 2, 4, 3, 5] (positions 1, 2, 3, 4, 5)

(2) initial process

↳ Population

C1 [5, 2, 4, 3, 5]

C2 [4, 3, 5, 1, 4]

C3 [2, 1, 3, 2, 4]

C4 [5, 2, 3, 4, 1]

let take our initial population

5	Q	0	0	Q	
4	0	0	Q	0	0
3	0	0	0	Q	0
2	0	Q	0	0	0
1	0	0	0	0	0

5	0	0	Q	0	0
4	Q	0	0	Q	Q
3	0	Q	0	0	0
2	0	0	0	0	0
1	0	0	Q	Q	0

C1 [5, 2, 4, 3, 5]   C2 [4, 3, 5, 1, 4

Genetic Algorithm

[0, 1, 1, 0] → population
[1, 0, 1, 1] → Chromosome
[0, 1, 1, 1] each

→ Gene value
→ Gene location

Evolutionary Algorithm
↳ optimaization
↳ knn → finding value of k → optimization Problem.

optimization techniques
→ Combinatrial
→ constrained opti
→ Multimodal
→ Multiobjective

Plane → birds
Submarine → Fish
radar → bats
→ natural Process

Evalutionary Algorithm

Algorithm
→ traditional
→ EA
↳ dynamic evolve over time

EA → main Characteristics
→ Population
→ Fitness
→ Variation - Driven

Population based
(Set of Solution) → Create population
Every Solution has an fitness value;
↓
How good the solution

Variation - Driven
↳ If the desired solution is not present in the population then we can create Randomly.

How working of Genetic Algorithm
↳ Population → chromosome
↓
fitness value
↓
higher fitness has ~~big~~ best Solution
↓
← Mating pool

Genetic Algorithm

→ Population
↓
fitness Cal..
↓
Mating pool
↓
parent selection
↓
Mating → Cross over
→ Mutation
↓
OFFspring

Chromosome Representation

↳ Binary
↳ permutation
↳ value

3 2 1 0 → locus

| 0 | 1 | 1 | 0 |

↳ allele

Representation types

↳ genotypes → set of genes
↳ phenotypes → actual physical representation

eg 7
↳ phenotypes
↳ 0111 → genotype

Variation Operator

↳ Cross over
↳ Mutation

Cross over

↳ Single-Point-Cross-over

Implementation

↳

~~equation_Inputs = [4, -2, 3.5, 5, -11, -4, 7]~~

~~equation_Inputs = [4,~~

$Y = w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + w_5x_5 + w_6x_6$

How to maximize $x \to$ Input ($x_1 \to x_6$)

$w \to$ weight ($w_1 \to w_6$)

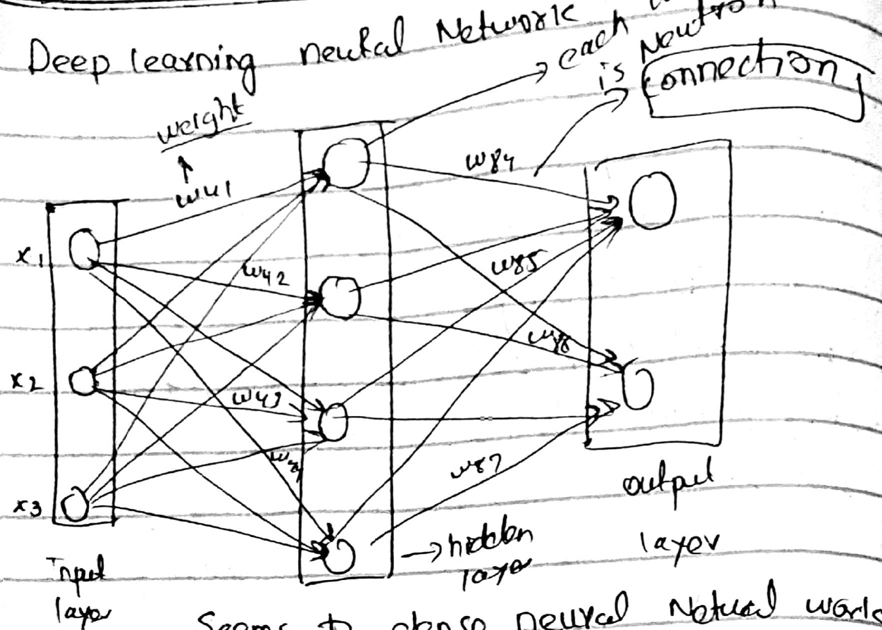
equation_Inputs = [4, -2, 3.5, 5, -11, -4, 7]

num of weight = 6

* find Initial Population

#01 NLP

Deep learning neural Network



Seems to dense neural Natual work

In order to make deep neural network we have to Increase the hidden layer. at least more then one.

Note: You have enough data to train the deep neural network. $w = \emptyset$

$f(x_1 \times \emptyset_{41} + x_2 \emptyset_{42} + x_3 \emptyset_{43})$

↳ look like linear model.

#02 video

→ Propagation function

↳ $f(x_1 \times \emptyset_{41} + x_2 \emptyset_{42} + x_3 \emptyset_{43})$

↳ putting into the Second layer

↳ most use Sigmoid function.

NLP #02

Hyper parameter

↳ Network

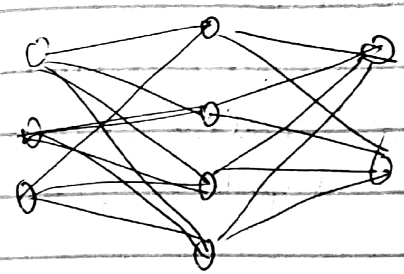
↳ have to define before the networ

↳ learning Rate

↳

↳ Batch size → number of Instance consumed the model.

Cost function



[0.3 - 1]
How much we away from one → cost function return this

Back propogation

↳ In to order to become near the 1 we change change the weight if the value is positive then decrease weight and if the value is negative we have to decrease the weight.

Artificial Intelligence

↳ Machine learning

↳ memorization → like store exact copy in mind.

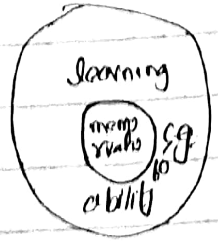
↳ Learning

↓

is generalization

memorization = learning

↳ learning on depend some features



Why learning Imp for Agent?

↳ to tetuke real data from world and feed back and Improve the Performance with passage of time.

eg Application of ML

↳

Human is object

Agent is learning

Two eye

Two tooth → hair

Two ear teeth

Features

Hypothesis

(AI)

Why Machine Learning?

↳ bcz Stock Market continuously Changes then Human can't be predict continuously.

Types of learnings

↳

Supervised learning → binary classification

↳ x → Features

↳ $f(x)$ → target value.

Error function

↳ How much our output away from the Actual output.

Multi-layer Neural Networks — Deep learning.

↳ KNN → Video share

↳ perceptrons,

Supervised learning

non-parametric classifier

need training data

α Class complexity ↑

KNN, decision Tree,

, SVMs → RBF

parametric classifier

↳ finite parameter

↳ 1 M parameter

↳ ANN, linear

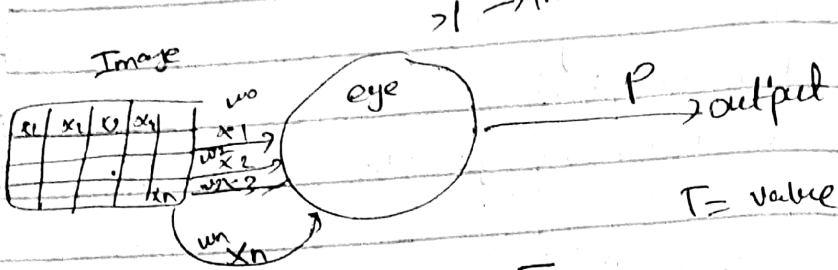
Support vector

(1) Artificial Neural Network

→ Neuron Network

Persptron → Neuron

for working of perceptron



$w_1x_1 + w_2x_2 + w_3x_3 \quad > T$

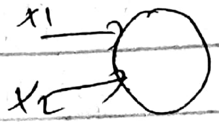
$-T + w_1x_1 + w_2x_2 + w_nx_n \quad > 0$

$w_1x_1 + w_2x_2 \quad \text{---} \quad w_nx_n \quad > T$

$-T + w_1x_1 + \text{---} \quad w_nx_n \quad > 0 \rightarrow 1$

else zero

$-w_0 + w_1x_1 + \text{---} \quad w_nx_n > 0$



x_i → fixed value

w_i → are variable

because w_0 → variable we can change

$w_0 + w_1x_1 + \text{----} \quad w_nx_n \quad > 0$

Assume $x_0 = 1$ → baiso Input

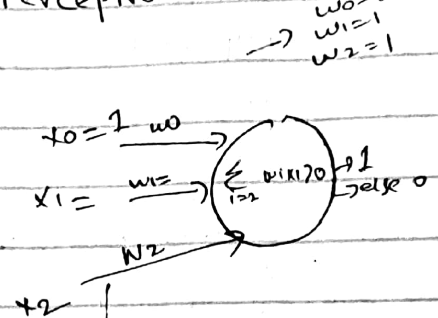
$w_0x_0 + w_1x_1 + w_nx_n > 0$

$\sum_{i=0} w_ix_i > 0 \; ; \; 1$

else 0

$1 \times 1 + 0 \times 1 + 0 \times 1 \Rightarrow 1 + 0 + 0 = 1 > 0$

Perceptron



x_1	x_2	t
0	0	0
0	1	1
1	0	1
1	1	1

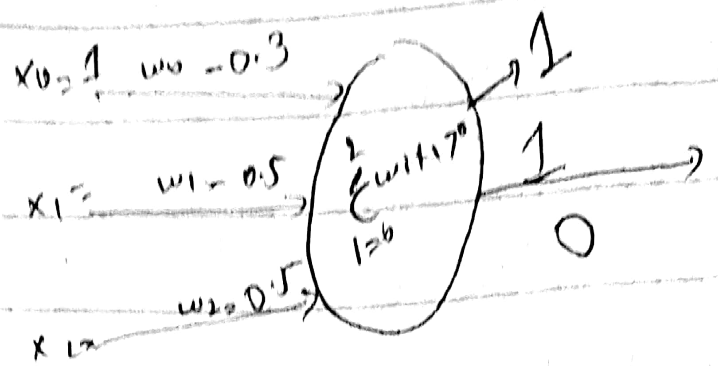
↳ learning is converyess of weight vector.

↳ $[w_0, w_1, w_2]$

w_i weight vector $[0, 1, 1]$

Perceptron

OR Gate



x_1	x_2	t
0	0	0
0	1	1
1	0	1
1	1	1

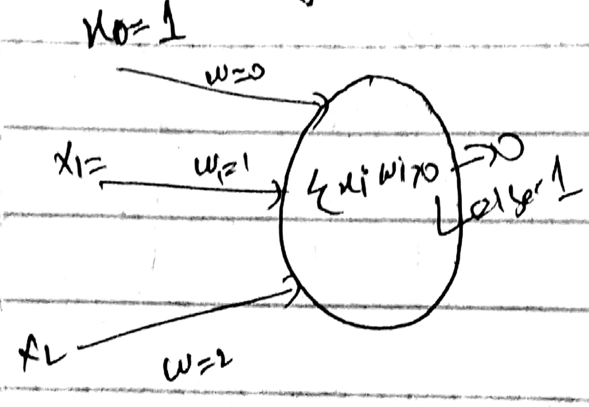
$-0.3 \times 0 + 0 = -0.3 < 0$

-0.3 x

We did above learning of OR Gate using Neuron

↳ What about AND Gate Neuron learning

AND Gate



x_1	x_2	t
0	0	0
0	1	0
1	0	0
1	1	1

What would be weight vector for AND gate Neuron learning

February 2020					
M		3	10	17	24
T		4	11	18	25
W		5	12	19	26
T		6	13	20	27
F		7	14	21	28
S	1	8	15	22	29
S	2	9	16	23	

March 2020					
M	30	2	9	16	23
T	31	3	10	17	24
W		4	11	18	25
T		5	12	19	26
F		6	13	20	27
S		7	14	21	28
S	1	8	15	22	29

April 2020					
M		6	13	20	27
T		7	14	21	28
W	1	8	15	22	29
T	2	9	16	23	30
F	3	10	17	24	
S	4	11	18	25	
S	5	12	19	26	

24 Friday 115/251

Machine learning = 01

8.00 am

what is Machine learning = ?

8.30

↳ Ranking Page
↳ Spam email
↳ face book App
↳ Rabot Clean Home
↳ Searching Algorithm
↳ Recongniation Friend on facebook
↳ learning Algorithm
↳ neural netwoke

9.00

9.30

10.00

10.30 How the work is working ??

11.00 ↳ Grew out of work in AI
↳ New Capability for Computer

11.30

example
↳ Database Mining

Noon

e.g web click data, medical records, biology, engineering.

12.30

1.00 → Application can't Program by hand
e.g, Auto nous helicopter, hand writting

1.30 recognition, most of NLP, Computer Vision.

2.00 →
→ Self-Customized Program
↳ Amazon recommended products

2.30

→ understand Human brain.

3.00

3.30

4.00

4.30

5.00

I would love to understand what other think?

Machine learning

8.00 am

What is Machine learning??

8.30

Def => ML -> field of study that gives computer the ability to learn without being explicitly Programmed

9.00

9.30

2nd def

10.00

TOM - Mitchell Well-posed learning Problem. A computer Program is said to learn from experience E with respect to some Task T and performance measure P. If its Performance on T as measured by P, improve with experience E.

10.30

11.00

11.30

Noon

-> E-mail Quizz

12.30

Machine learning

1.00

- Supervised learning
  - I teach the machine How you will do?
- Un-supervised learning
  - learn itself
- Reinforcement learning
  - -> Something else Not yet

1.30

2.00

2.30

for example Playing Checker

3.00

E = the experience of Playing many games of Checker
T = the task of Playing checker
P = Probability that the Program will win the next game

3.30

4.00

In general ML can be assigned of two broad Classification
-> Supervised
-> un-supervised

4.30

5.00

6.00 pm

Supervised learning

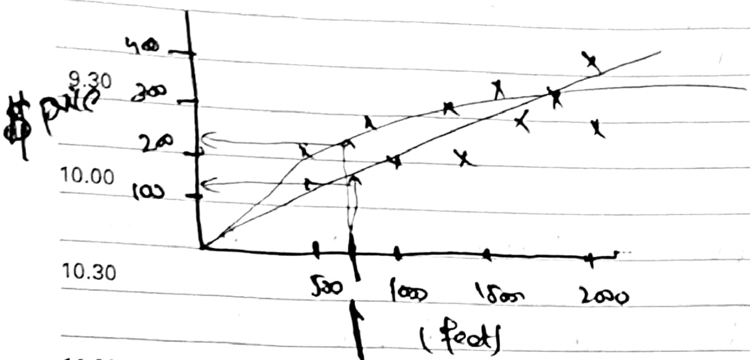
Machine learning

Supervised learning

House price Prediction

Supervised learning

$\boxed{\text{right Answer were given}}$



This is Regression Problem:
↳ predict continuous valued output (Price)

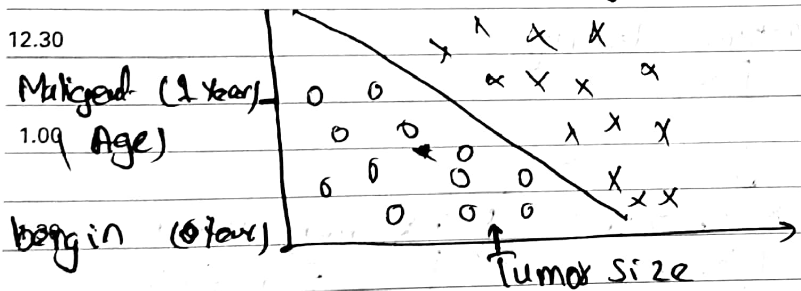
what about this → make straight line
↳ make quadratic equation line

Tumor example

Classification problem

$\boxed{\text{Malignant/ begin}}$

begin → 0
Malignant → 1



- Clump Thickness
→ uniformity of cell size
→ uniformity of cell shape.

Here we have only Two feature Age, Tumor size but In ML Problem we have lot

→ we predict in real word infinit Number of feature
↳ computer run out memory
↳ Support vector Machine
↳ Math tip's

quiz?

May 2020

M		4	11	18	25
T		5	12	19	26
W		6	13	20	27
T		7	14	21	28
F	1	8	15	22	29
S	2	9	16	23	30
S	3	10	17	24	31

June 2020

M	1	8	15	22	29
T	2	9	16	23	30
W	3	10	17	24	
T	4	11	18	25	
F	5	12	19	26	
S	6	13	20	27	
S	7	14	21	28	

July 2020

M		6	13	20	27
T		7	14	21	28
W	1	8	15	22	29
T	2	9	16	23	30
F	3	10	17	24	31
S	4	11	18	25	
S	5	12	19	26	

Anzac Day Holiday (NZ) 118/248 Monday 27

Machine learning → unsupervised learning

8.00 am

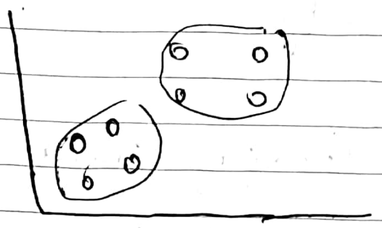
Unsupervised Learnings

8.30

↳ In unsupervised learning we don't know about the output we have just lot of data.

9.00

9.30



then unsupervised learning Algo made cluster of similar data.

10.00

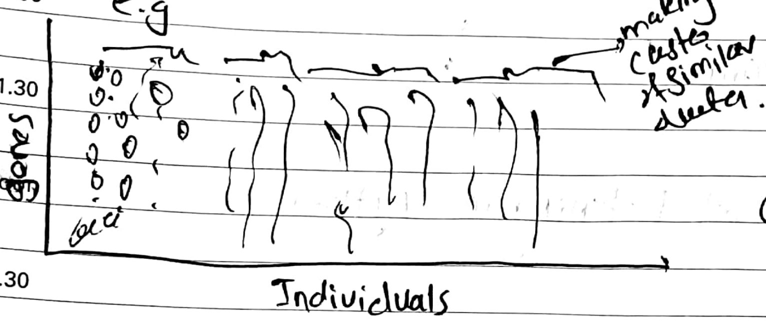
10.30

Other example of un-supervised learning

11.00

1 → Organize computing cluster
2 → Social Network analysis
3 → Market Segment Analysis
4 → Astronomical Analysis

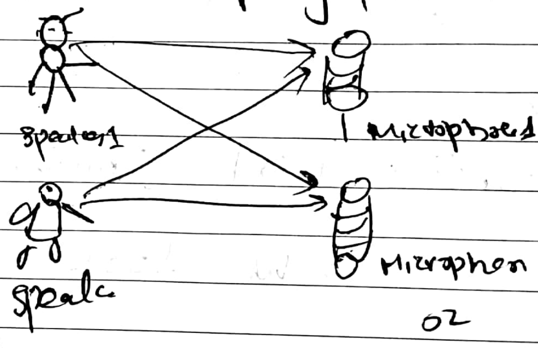
e.g



11.30

Noon

Cocktail Party problem



12.30

1.00

1.30

2.00

2.30

3.00

3.30

4.00

4.30

5.00

6.00 pm

March 2020					
M	30	2	9	16	23
T	31	3	10	17	24
W		4	11	18	25
T		5	12	19	26
F		6	13	20	27
S		7	14	21	28
S	1	8	15	22	29

April 2020					
M		6	13	20	27
T		7	14	21	28
W	1	8	15	22	29
T	2	9	16	23	30
F	3	10	17	24	
S	4	11	18	25	
S	5	12	19	26	

May 2020					
M		4	11	18	25
T		5	12	19	26
W		6	13	20	27
T		7	14	21	28
F	1	8	15	22	29
S	2	9	16	23	30
S	3	10	17	24	31

4 Monday 125/241

May Bank Holiday (IE) Greenery Day (JP) May Day Bank Holiday (UK)

8.00 am Perceptron

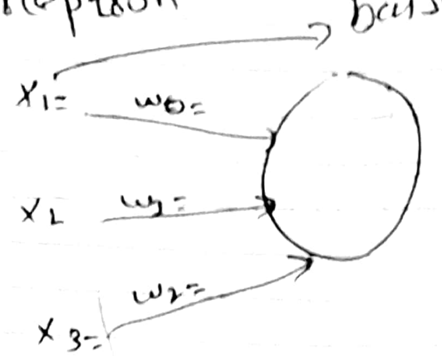
Input

x_1	x_2	t
0	0	0
0	1	1
1	0	1
1	1	1

8.30

9.00

9.30



10.00

what will be weight vector

10.30

$W = [w_0, w_1, w_2]$

11.00

$w = [-0.3, 0.5, 0.5]$

11.30

Simple example → Imp

Noon

Key equation

learning Rate

corresponding output

12.30

$$\Delta w_i = \Delta w_i + \eta (t - o) x_i$$

1.00

① initialize the weight to random values.

1.30

$\llcorner w_i \rightarrow w_n = 0$ (w = weights)

2.00

2.30

	x_1	x_2	Actual td	Perceptron output
d_1	0	0	0	0 → ✓
d_2	0	1	1	0 → x
d_3	1	0	1	1 → ✓
d_4	1	1	1	0 → x

3.00

3.30

4.00

can compute error.

4.30

$$E_{rror} = \sum_{d \in D} |t_d - O_d| \leftarrow$$ ERROR 50% Absolute error

5.00

6.00 pm $$\text{Error} = \frac{1}{2} \sum_{d \in D} |t_d - O_d|^2 \rightarrow$$ Squared ERROR epoach → one time of the exection of the code is called epoach.

8.00 am

If error > 0
↳ enter the code

8.30

9.00

η → learning Rate
$= 0.01$ $\eta = 0.1$

9.30

↳ ① It control the phase of perceptron learning.

10.00

10.30

② It effect the solution convergence.

11.00

η → Small → good convergence.

11.30

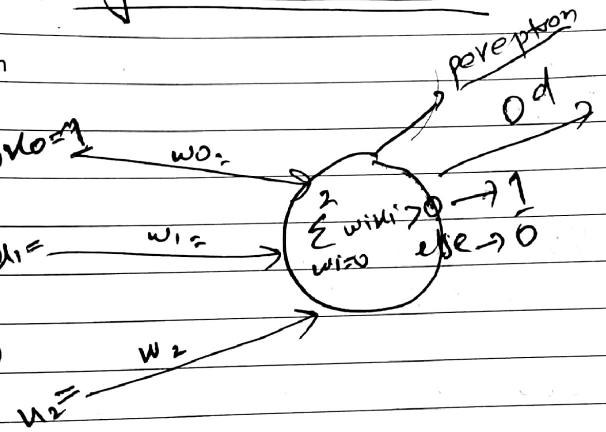
Algorithm Execution

Noon

12.30

1.00

1.30



OR Gate (+) → perceptron output

	x_1	x_2	td	Od
d1	0	0	0	0
d2	0	1	1	
d3	1	0	1	
d4	1	1	1	

2.00

① first step

2.30

↳ $w_0 = 0$, $w_1 = 0.5$ $w_2 = 0.4$ → Randomly

3.00

② Calculate error → Accumulation

$w_i = [0.1, 0.5, 0.4]$

3.30

$d_1 \to x_1 = 0$ $x_2 = 0$

4.00

$= w_0 x_0 + w_1 x_1 + w_2 x_2$
$= 0 \times 1 + 0.5 \times 0 + 0.4 \times 0 = 0$

4.30

$d_2 \to x_1 = 0$ $x_2 = 1$

5.00

6.00 pm

May

WEEK 19

March 2020					
M	30	2	9	16	23
T	31	3	10	17	24
W		4	11	18	25
T		5	12	19	26
F		6	13	20	27
S		7	14	21	28
S	1	8	15	22	29

April 2020					
M		6	13	20	27
T		7	14	21	28
W	1	8	15	22	29
T	2	9	16	23	30
F	3	10	17	24	
S	4	11	18	25	
S	5	12	19	26	

May 2020					
M		4	11	18	25
T		5	12	19	26
W		6	13	20	27
T		7	14	21	28
F	1	8	15	22	29
S	2	9	16	23	30
S	3	10	17	24	31

6 Wednesday 127/239

Randomly

Constitution Day Holiday (JP)

8.00 am

$d_2 = (0, 1)$ $w_i[0.1, 0.5, 0.4]$

8.30

$w_0x_0 + w_1x_1 + w_2x_2$ ✓

9.00

$0.1 \times 1 + 0.5 \times 0 + 0.4 \times 1 =$

9.30

$0.1 + 0 + 0.4 = 0.5 > 0$

10.00

$\rightarrow 1$

10.30

x_1	x_2	td	Od	
0	0	0	1	→ ✗
0	1	1	1	→ ✓
1	0	1	1	→ ✓
1	1	1	1	→ ✓

11.00

11.30

Noon

What about Absolute error ✓

→ 25%

12.30

Step 2

1.00

→ ERROR = 25% > 0; we use inside the loop.

1.30

Δw_i initilize by zero

2.00

$\Delta w_0 = 0$ $\Delta w_1 = 0$ $\Delta w_2 = 0$

2.30

For each training example

3.00

$\Delta w_i = \Delta w_i(\text{old}) + \eta\,(t_d - O_d) \times x_i$.

3.30

for $d=1$

4.00

4.30

5.00

6.00 pm

Randly

weight base

w_i [0.1, 0.5, 0.4]

OR gate (+)

x_1	x_2	td	Od
0	0	0	1
0	1	1	1
1	0	1	1
1	1	1	1

8.00 am

→ our error is 25%

8.30

Step 2

9.00

↳ we are in bp b/c 25%>0

9.30

initialize weight (w_i) with zero

10.00

$\Delta w_0 = 0$ $\Delta w_1 = 0$ $\Delta w_2 = 0$

10.30

For each training sample

11.00

0.1 ~~0.01~~ learning rate.

$\boxed{\text{for } d_1}$

11.30

$\Delta w_i = \Delta w_i (old) + \eta \; (td - od) \times x_i$

Noon

$\Delta w_0 = \Delta w_0 + 0.01 \; (0-1) \times 1$

$\Delta w_0 = 0 + 0.01 \; (0-1) \times 1$

12.30

$= 0.01 \; (-1) \times 1$

$= -0.01$

1.00

$\Delta w_0 = \boxed{-0.1}$

1.30

$\Delta w_1 = \Delta w_1 + 0.1 \; (0-1) \times 0$

$\Delta w_1 = 0 + 0.1 \; (-1) \times 0$

2.00

$\Delta w_1 = 0$

2.30

$\Delta w_2 = \Delta w_2 + 0.1 (0-1) \times 0$

$\Delta w_2 = 0 + 0$

3.00

$\Delta w_2 = 0$

3.30

for d_2

4.00

$\Delta w_0 = \Delta w_0 + 0.1 \; (1-1) \; 1$

$\Delta w_0 = -0.1 + 0$

4.30

$\Delta w_0 = -0.1$

5.00

6.00 pm

8.00 am $\Delta w_1 = \Delta w_1 + 0.1(1-1) \times 0$

$\Delta w_1 = 0 +$

8.30 $\Delta w_1 = 0$

9.00 $\Delta w_2 = \Delta w_2 + 0.1(1-1) \times 1$

$\Delta w_2 = 0 + 0$

9.30

for $d = 3$ — for $d = 4$

10.00

$\Delta w_0 = -0.1$ — $\Delta w_0 = -0.1$

10.30 $\Delta w_1 = 0$ — $\Delta w_1 = 0$

$\Delta w_2 = 0$ — $\Delta w_2 = 0$

11.00

Now update our weight of All the

11.30 Iteration

$w_i = w_i + \Delta w_i$

Noon

$w_0 = w_0 + \Delta w_0$

12.30 $w_0 = 0.1 - 0.1 = 0$

$w_1 = 0.5 + 0 = 0.5$

1.00 $w_1 = 0.4 + 0 = 0.4$

1.30 updated weight $[0, 0.5, 0.4]$

↳ check the error and then

2.00

error will be zero.

2.30

whats is hypothesis function = ?? $h(x_1 x)$

3.00

$h(x) = w_0 x_0 + w_1 x_1 + w_2 x_2$

3.30

$h(x) = 0 \times x_0 + 0.5 x_1 + 0.4 x_2$

4.00

$h(x) = 0.5 x_1 + 0.4(x_2)$

4.30

5.00

6.00 pm

March 2020					
M	30	2	9	16	23
T	31	3	10	17	24
W		4	11	18	25
T		5	12	19	26
F		6	13	20	27
S		7	14	21	28
S	1	8	15	22	29

April 2020					
M		6	13	20	27
T		7	14	21	28
W	1	8	15	22	29
T	2	9	16	23	30
F	3	10	17	24	
S	4	11	18	25	
S	5	12	19	26	

May 2020					
M		4	11	18	25
T		5	12	19	26
W		6	13	20	27
T		7	14	21	28
F	1	8	15	22	29
S	2	9	16	23	30
S	3	10	17	24	31

10 Sunday 131/235

8.00 am

Machine learning: Perceptron Activations function

8.30

9.00 (2) Mathematicall

9.30 equation

10.00 $\Delta w_i = -\eta \frac{dE}{dw_i}$ ← key equation

10.30

→ weight update Equation. with sigmoid form:

11.00

11.30 (2) function

Tangent Hyperbolic function

Noon

If we use Tangent Hyperbolic function what will be equation ??

12.30

give more freedom in output ← ↳ Imp

1.00

1.30

$net = \sum_{i=0}^{n} w_i x_i$, $od = \tanh(net)$

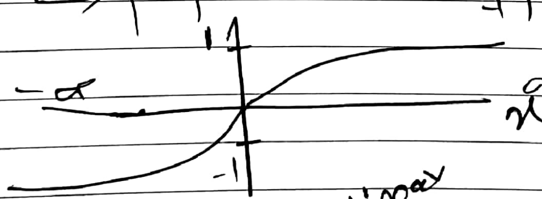
2.00

2.30 $\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$ →

3.00

↳ properties

3.30



→ return [-1, +1]

↳ why we use this sigmoid already we have

↓

because sigmoid return zero and they can easily stuck when return 0

4.00

↳ non linear function

4.30

5.00

8.00 am

When od is zero then whole equation become zero.

8.30

↳ Less suspected gradient vanish problem.

9.00

find derivate of this equation

9.30

$y = \text{Tanh}(x) = \frac{\sinh(x)}{\cosh(x)}$

10.00

$y' = 1 - \text{Tanh}(x)\, d$

10.30

E → od ka function

$\Delta w_i = -\eta \cdot \frac{\partial E}{\partial w_i}$

11.00

$E = \frac{1}{2} \Sigma (td - od)^2$

11.30

Noon

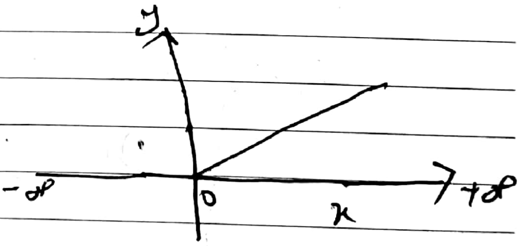
$od = \tanh(\text{netd})$

12.30

$\text{netd} = \sum_{i=0}^{n} w_i x_i$

1.00

③ Relu function



1.30

$y = f(x) = \max(0, x)$

2.00

$y = f(x) = \max(0, x)$

2.30

$\begin{cases} 0 ; & x \le 0 \\ x ; & x > 0 \end{cases}$ → give more freedom

3.00

↳ give more positive value

3.30

→ $y \in (0, +\infty)$

4.00

4.30

$y' = \begin{cases} \frac{d\,0}{dx} ; & x \le 0 \\ \frac{d\,x}{dx} ; & x > 0 \end{cases} \Rightarrow \begin{cases} 0 ; & x \le 0 \\ 1 ; & x > 0 \end{cases}$

5.00

6.00 pm

March 2020

M	30	2	9	16	23
T	31	3	10	17	24
W		4	11	18	25
T		5	12	19	26
F		6	13	20	27
S		7	14	21	28
S	1	8	15	22	29

April 2020

M		6	13	20	27
T		7	14	21	28
W	1	8	15	22	29
T	2	9	16	23	30
F	3	10	17	24	
S	4	11	18	25	
S	5	12	19	26	

May 2020

M		4	11	18	25
T		5	12	19	26
W		6	13	20	27
T		7	14	21	28
F	1	8	15	22	29
S	2	9	16	23	30
S	3	10	17	24	31

12 Tuesday 133/233

Perceptron Activation function.

8.00 am

$$\Delta w = -\eta \frac{dE}{dw_i}$$

8.30

9.00

③ leaky ~~Re~~ Relu.

9.30

$\alpha = 0.01 \rightarrow$ How much you tolerate?

10.00

10.30

11.00

11.30

$y = \max(\alpha x, x)$

Noon

↳ if $x < 0$

↳ αx

12.30

else

1.00

↳ x

1.30

$$y = \begin{cases} \alpha x ; & x \le 0 \\ x ; & x > 0 \end{cases}$$

2.00

2.30

$$y' = \begin{cases} \frac{d}{dx}(\alpha x) ; & x \le 0 \\ \frac{d}{dx}(x) ; & x > 0 \end{cases} \Rightarrow \begin{cases} \alpha , & x \le 0 \\ 1 & x > 0 \end{cases}$$

3.00

3.30

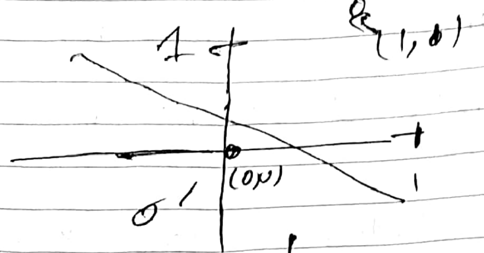
4.00

4.30

5.00

6.00 pm

* from Single perceptron to Neural Network

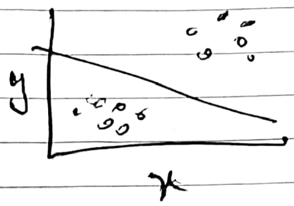


XOR func

x_1	x_2	O
0	0	0
	1	1
1	0	1
0	1	0

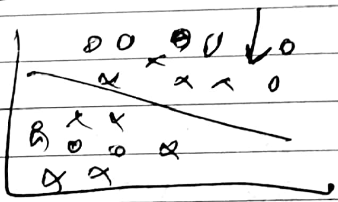
↳ 25% error data Hai

* linearly separable data:-
↳ is a data distribution which can seperate the data point into the binary classes using the linear function.



$y = XOR(x_1, x_2)$

↳ non-linear Sparable data



When you have ~~not~~ non liearn ~~ol~~ sparable data then you have to use more then one classifier.

$y = x_1 \oplus x_2 = (x_1 \vee x_2) \wedge -(x_1 \wedge x_2)$

OR → 1 perceptron

AND → 1

Not → 1

	May 2020					
M		4	11	18	25	
T		5	12	19	26	
W		6	13	20	27	
T		7	14	21	28	
F	1	8	15	22	29	
S	2	9	16	23	30	
S	3	10	17	24	31	

	June 2020				
M	1	8	15	22	29
T	2	9	16	23	30
W	3	10	17	24	
T	4	11	18	25	
F	5	12	19	26	
S	6	13	20	27	
S	7	14	21	28	

	July 2020				
M		6	13	20	27
T		7	14	21	28
W	1	8	15	22	29
T	2	9	16	23	30
F	3	10	17	24	31
S	4	11	18	25	
S	5	12	19	26	

23 Thursday 205/161

Tokyo Olympics (JP)

8.00 am Machine learning KNN

8.30 Types of learning:

9.00

9.30 Empirical ERROR function

10.00 $h(x, \theta) = Ann(\bar{x}, \theta)$ # Hyphothesis

10.30 parameter

11.00 features 2^2

$2 \quad 2^4 = 16$

11.30 Hyphothesis space. $2 \times 2 \times 8 \times 2 \ 0$

nube of featu

Noon possible values

Formula = (Possible value for target variable) of features

12.30

1.00 $= 2^{2^4} = 2^{16}$

1.30 $10^{10^4} = 10^{10,000}$ huge number.

2.00

Ockham Razer Principle

2.30

$h_1(x) = ax + b$ simple

3.00 $h_2(x) = ax^5 + bx^4 + cx^3 + dx^2 + e$ comple

3.30 Choose simple because the generalization of simple hyphothesis

4.00 is easier then a complex one.

4.30

5.00

pm

Hypothosis function

8.00 am

Classification in Encludean Space

8.30

↳ Training and Validation Data

9.00

↳ The V-fold cross validation method.

9.30

↳ leave-one-out validation

↳ most expensive

10.00

Under fitting and over fitting

10.30

11.00

High Variance

↳ over fitting

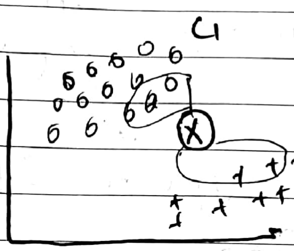
K Nearest Neighbour

11.30

High baise

↳ underfitting

Noon



k=5

x → C1 → 3 → k max

x → C2 → 2

12.30

1.00

Enclusion

Manhatan distance.

1.30

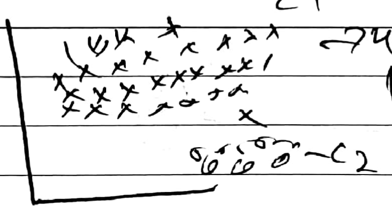
Problem with Imbalance dataset

2.00

eg x = 900

y 2 100

2.30



K = 150

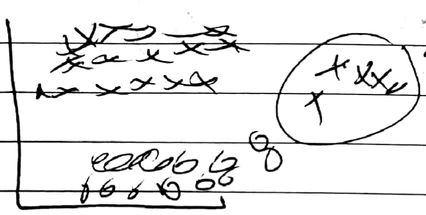
3.00

He will join C2

3.30

Problem With outlier

4.00



→ outlier

4.30

5.00

6.00 pm

May 2020

M		4	11	18	25
T		5	12	19	26
W		6	13	20	27
T		7	14	21	28
F	1	8	15	22	29
S	2	9	16	23	30
S	3	10	17	24	31

June 2020

M	1	8	15	22	29
T	2	9	16	23	30
W	3	10	17	24	
T	4	11	18	25	
F	5	12	19	26	
S	6	13	20	27	
S	7	14	21	28	

July 2020

M		6	13	20	27
T		7	14	21	28
W	1	8	15	22	29
T	2	9	16	23	30
F	3	10	17	24	31
S	4	11	18	25	
S	5	12	19	26	

25 Saturday 207/159

8.00 am Multi-layer Network

8.30 Back Propagation

9.00 ① XOR learning Using Neural Network

9.30 ② Back Propagation ~~err~~ Algorithm

10.00

x_1	x_2	t
0	0	0
0	1	1
1	0	1
1	1	0

1 perceptron

$x_1 \oplus x_2 = (x_1 \cup x_2) \wedge \sim(x_1 \wedge x_2)$

1 percep

1 perceptron 1 percep

Noon OR

12.30 P_1

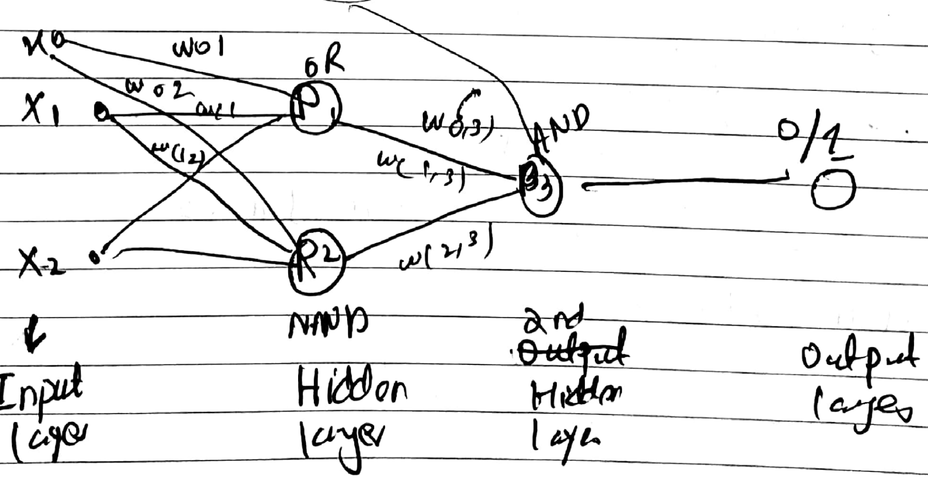
1.00 P_3

1.30 AND

P_2

2.00 NAND

x_0 = → baise Node



3.00

3.30

4.00

4.30

5.00