Local Search & optimization :

Genetic Algorithm (GA) :

→ metaheuristic optimization algorithm inspired by the process of natural selection & evolution (biological evolution of species)

→ widely used to find near-optimal solutions to hard problems that are too complex to deterministic methods (like linear equation systems, Np-hard problems etc)

Chromosome → a possible solution in a population.
Gene → represents a variable or param as part of chromosom
Population → a set of candidate solution (chromosomes)
Fitness ~~Solution~~ Function → A measure of how "good" a chromosome is
Selection → Picking the best-fit chromosome for reproductio
Crossover → Combining 2 parent chromosomes to create offspring
Mutation → Randomly changing genes to maintain diversity
Generation → one full cycle of evolution
(selection → crossover → mutation → replacem

Genetic Algo Steps.

1. Encoding (Representation)
   ☆ → a possible soln (chromosome) is reprented in format
   eg. : Binary, Integer, Real value etc
2. Initial Population Generation
   ☆ Generate initial set of solutions (chromosomes).
3. Fitnen Function Evaluation
   ☆ Calculate fitness score for each chromosome. to know how good it is.
4. Selection (Parent Selection) :
   ☆ Roulette Wheel Selection : Prob. proportional to fitness
   ☆ Tournament Selection : Randomly select a group. select the best
   ☆ Rank Selection : Based on sorted fitnen.

4- (Crossover) (Recombination)

★ create offspring by combining parent ☆ ge...

★ Types

- Single point Crossover – swap parts after a point
- Two point Crossover – swap parts between 2 points
- Uniform Crossover – swap genes at random positions

5- Mutation

★ Randomly change genes with low prob. (to avoid local minima)

① flip bit Mutation ② Gaussian Mutation ③ Exchange/swap mutation

7 - Replacement (forming New Generation)

★ Replace old population with

- only offspring
- or best of old and new (elitism)

8- Termination Condition:

Stop if

- A solution of acceptable fitness is found
- or maximum generations reached
- or sol^n has not improved a while

Genetic Algo flow diagram :-

Start → Generate Initial population → Evaluate fitness →

→ While Not terminated :

- select parents
- ↓
- Crossover to create offspring
- ↓
- Mutate offspring
- ↓
- Evaluate fitness of offspring.
- ↓
- Form New Population (Replace old)

↓

Output Best Solution find.

Pros of GA	Cons of GA
can handle non-linear, multi-modal problems • do not require gradient or problem structure • finds good solⁿ for complex, large search spaces	★ computational expensive for large population / generations. ★ may converge to local optima without proper diversity (Mutation helps) ★ require careful param tuning (mutation rate, crossover rate, population size)

exp of GA : Machine learning (Feature selection), Game Playing, Scheduling problems, Equation Solving

Question : Maximize the function $f(x) = x^2$ where $x \in [0, 31]$ show Genetic Algo steps up to first iteration.

Soln:

Step1 Encoding (Chromosome Representation)

Let each solution (chromosome) represent a value of x encoded as a 5 bit binary string (since $2^5 = 32$)

ex. • $x = 0 \rightarrow 00000$
• $x = 31 \rightarrow 11111$

Step2 Initial population generation

Let's randomly generate 4 chromosomes.

Chromosome	Binary	Decimal (x)
C_1	01101	13
C_2	10100	20
C_3	00011	3
C_4	11010	26

Step3 Fitness function Calculation

	x	$f(x) = x^2$ (fitness)
C_1	13	169
C_2	20	400
C_3	3	9
C_4	26	676

Step4 - Roluette wheel Selection:

Total fitness = 169 + 400 + 9 + 676 = 1254

Selection Probability

- C_1 169/1254 ≈ 0.1340
- C_2 400/1254 ≈ 0.3191
- C_3 9/1254 ≈ 0.0072
- C_4 676/1254 ≈ 0.5390

Perform Selection (Roluette wheel spins)

- Spin1 : Picks C_4 (highest Prob)
- Spin2 : Picks C_2

Selected Parents C_4 & C_2.

Step5 Crossover

Parent 1 (C_4) : 11010 (x=26)
Parent 2 (C_2) : 10100 (x=20)

Assum crossover point after 2nd bit (index 1)

- child 1 : 11100 (x=28)
- child 2 : 10010 (x=18)

Step6 Mutation:

Mutation probability = 10% per bit.
Assume child 2 gets mutated at 4th bit

- Before = 10010 (x=18)
- After Mutation : 10000 (x=16)

Child 1 remains unchanged

Step 7 Evaluate offspring fitness

Child 1 : 11100 ($x = 28$) $f(28) = 28^2 = 784$

Child 2 : 10000 ($x = 16$) $f(16) = 16^2 = 256$

Step 8 Replacement (Elitism)

Select 4 chromosomes from old + new.

chromosome	x	fitness
Child 1	28	784
C4	26	676
C2	20	400
Child 2	16	256

Summary of first iteration (Maximization objective)

- Initial population generated
- fitness evaluated (maximize x^2)
- Roulette wheel selection done
- Crossover + Mutation performed
- New best solution $x = 28$ (784) found
- Next generation formed with better overall fitness.

Q2 Solve below linear equation step by step (GA)

$$2a + 7b - 5c + d = 0$$

$$a, b, c, d \in [-20, 20]$$

Soln

Step ① Encoding (Chromosome Representation)

each chromosome represents a candidate soln.

$[a, b, c, d]$ $a, b, c, d \in [-20, 20]$

Step ② Initial Population Generation

Let's generate 4 chromosomes (randomly in the allowed range)

Chromosome	Representation $[a, b, c, d]$
C_1	$[4, -3, 2, 1]$
C_2	$[-2, 0, -1, 5]$
C_3	$[0, 2, 1, -3]$
C_4	$[5, -1, 0, 0]$

Step ③ Fitness function Calculation

We aim to minimize the absolute error:

$$\text{Fitness} = \frac{1}{1 + |2a + 7b - 5c + d|}$$

(+1 ⇒ in denominator avoids division by zero)

Chromosome	Eq value $(2a+7b-5c+d)$	Absolute	Fitness
C_1 $[4, -3, 2, 1]$	$2(4) + 7(-3) - 5(2) + 1 = -22$	22	$1/23 = 0.043$
C_2 $[-2, 0, -1, 5]$	$2(-2) + 7(0) - 5(-1) + 5 = 6$	6	$1/7 \approx 0.143$
C_3 $[0, 2, 1, -3]$	$2(0) + 7(2) - 5(1) + (-3) = 6$	6	$1/7 \approx 0.143$
C_4 $[5, -1, 0, 0]$	$2(5) + 7(-1) - 5(0) + (0) = 3$	3	$1/4 \approx 0.25$

Step ④ Roulette wheel selection

Total fitness $= 0.043 + 0.143 + 0.143 + 0.250$

$= 0.579$

selection probability for each

C1 = 0.043/0.579 ≒ 0.074
C2 = 0.143/0.579 ≒ 0.247
C3 = 0.143/0.579 ≒ 0.247
C4 = 0.250/0.579 ≒ 0.432

Performing Selection:

- Spin 1 : Picks C4 (highest) chance
- Spin 2 : Picks C2 (2nd highest prob)

Step ⑤ Crossover (Single point crossover)

Parents C4 [5, -1, 0, 0]
C2 [-2, 0, -1, 5]

Random crossover point : after gene 2 (index 1)

child 1 : [5, -1, -1, 5]
child 2 : [-2, 0, 0, 0]

Step ⑥ Mutation

Mutation Prob : 10% per gene
Assume 1 gene mutated

- child 2's gene 3 mutated : 0 → 1

New child2 : [-2, 0, 1, 0]

Step ⑦ fitness calculation of offspring

child 1 [5, -1, -1, 5] = 2(5) + 7(-1) - 5(-1) + 5 = 13
fitness = 1/1+13 = 0.071

child 2 [-2, 0, 1, 0] = 2(-2) + 7(0) - 5(1) + (0) = -9
fitness = 1/1+|-9| = 1/10 = 0.1

search 1) + h(A)

Step ⑤ Replacement (Elitism)

- keep best of old + new population (elitism)

Best C4 (0.25), C2 (0.143), Child2 (0.1), Child1 (0.01)

New population

1. C4 : [5, -1, 0, 0]
2. C2 : [-2, 0, -1, 5]
3. Child2 : [-2, 0, 1, 0]
4. child1 : [5, -1, -1, 5]

Summary of first GA Iteration (with Roluette wheel selection)

- Initial population created
- Fitness evaluated
- Roluettle wheel selection performed
- Crossover (single point) applied
- Mutation applied (one gene)
- New generation formed via elitism.

Ant Colony Optimization :

- ACO is a meta heuristic algo inspired by the behavior of real ants searching for food.
- Ants deposits pheromones on the paths they travel.
- shorter path accumulate more pheromone because they are travelled more frequently. other ants tend to follow the paths with stronger pheromones concentration.

Steps :

(1) Initialization :

- set initial pheromones levels on the edges
- define params.
  - α (alha) : importance of pheromone intensity coefficient
  - β (beta) : importance of visibility (1/distance) coefficient
  - ρ (rho) : evaporation rate $0 < \rho < 1$
  - Θ : constant for pheromone update.
  - τ_0 : 0

(2) Construct Ant Solutions : each ants builds a complete path using probability.

$$NTP_{ij} = \frac{(\tau_{ij})^{\alpha} \cdot (\eta_{ij})^{\beta}}{\sum_{h \notin \text{visit}k} (\tau_{ih})^{\alpha} \cdot (\eta_{ih})^{\beta}}$$

where NTP = NextNode transition probability

τ_{ij} = pheromone on edge (i, j)

η_{ij} = visibility = 1/distance

Σ = sum over all possible moves

(3) Pheromone update :

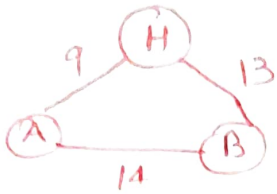
After all ant finishes $\tau_{ij} = (1 - \rho) \cdot \tau_{ij} + \Delta \tau_{ij}$

where $\Delta \tau_{ij} = \sum_{k=1}^{m} \frac{\Theta}{L_k}$ if edge (i, j) used by ant k

L_k = tour length by ant k.

(4) Repeat till convergence or max iteration.

Prob Abhay starts from home & visits two cities and has to reach back. The cost of each route between cities & between his home & cities is given. Determine the shortest path through Ant Colony optimization.



Pheromone matrix is given as below

	H	A	B
H	0	0.15	0.26
A	.15	0	0.48
B	.26	0.48	0

Given parameters:

- Evaporation (ρ) = 0.1
- θ = 90
- α = 0.3 (pheromone importance coeff)
- β = 0.4 (distance importance)

Soln

Step 1 Visibility calculation (η = 1/distance)

$\eta_{HA} = \frac{1}{9} = .1111$ $\eta_{AB} = \frac{1}{14} = .0714$ $\eta_{HB} = \frac{1}{13} = 0.769$

Step 2 Probability calculation from Home (H).

$$P_{HA} = \frac{(\tau_{HA})^{\alpha} \cdot (\eta_{HA})^{\beta}}{[\tau_{HA}]^{\alpha} \cdot (\eta_{HA})^{\beta} + [\tau_{HB}]^{\alpha} \cdot (\eta_{HB})^{\beta}}$$

$$P_{HA} = \frac{(0.15)^{0.3} \times (0.1111)^{0.4}}{(0.15)^{0.3} \times (0.1111)^{0.4} + (0.26)^{0.3} \cdot (0.769)^{0.4}} = 0.497$$

$$P_{HB} = \frac{(0.26)^{0.3} \cdot (0.769)^{0.4}}{(0.15)^{0.3} \cdot (0.1111)^{0.4} + (0.26)^{0.3} \cdot (0.769)^{0.4}} = 0.503$$

⇒ Ant can go to A or B with almost equal probability

Step 3 Possible routes & lengths:

to visit A & B and return to home, there are 2 possible tours

① $H \to A \to B \to H$ length = 9+14+13 = 36

② $H \to B \to A \to H$ length = 13+14+9 = 36

Step 4: Pheromone Update:

$$\Delta \tau_{ij} = \frac{Q}{L} = \frac{90}{36} = 2.5$$

Pheromone Evaporation:

$$\tau_{ij} = (1-\rho)\,\tau_{ij} + \Delta\tau_{ij}$$

$$\tau_{ij}' = (1-0.1)\cdot\tau_{ij} + 2.5$$

for HA edge $\tau_{HA} = (1-0.1)\,\tau_{HA} + 2.5$

$= 0.9 \times 0.15 + 2.5 = 2.635$ (from matrix)

$\tau_{HB} = (1-0.1) \times \tau_{HB} + 2.5$

$= 0.9 \times 0.26 + 2.5 = 2.734$

$\tau_{AB} = (1-0.1) \times \tau_{AB} + 2.5$

$= 0.9 \times 0.40 + 2.5 = 2.932$

Step 5 : Conclusion

- Both Path have equal length(36) so either can be select
- After 1 iteration, the pheromones on all paths are increased
- ACO will gradually reinforce the paths most frequently used.

Search

- Informal Search
  - We will not look in all the possible direction / ways to reach the goal.
  - look only into particular direction to reach the goal.
  - Exp
    - Gready B.F.S. (Best First Search)
    - A* algorith
    - RBFS. (Recursive B.F.S.)
- Uninformal Search (blind)
  - We will search in all possible direction or ways to reach the goal state
  - explore all possible ways.
  - Exp.
    - DFS, BFS
    - Uniform Cost search
    - Bidirectional Search
    - Iterative deepening Search.

A* Algorithm :-

* Expands the node which lies in the closest path (estimated cheapest path) to the goal.
* Evaluation function

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

$g(n)$ = cost to reach the node / path cost

$h(n)$ = the expected (→ or heuristic) cost to go from node to goal

heuristic value ←

$f(n)$ = estimated cost of cheapest path through node n.

Optimality of A* Algorithm :-

* A heuristic is admissible or optimistic if $0 \leq h(n) \leq h^*(n)$

where $h^*(n)$ = actual cost to the goal

: In exam graph is given

optimal on condition : $h(n)$ - must satisfies 2 conditions

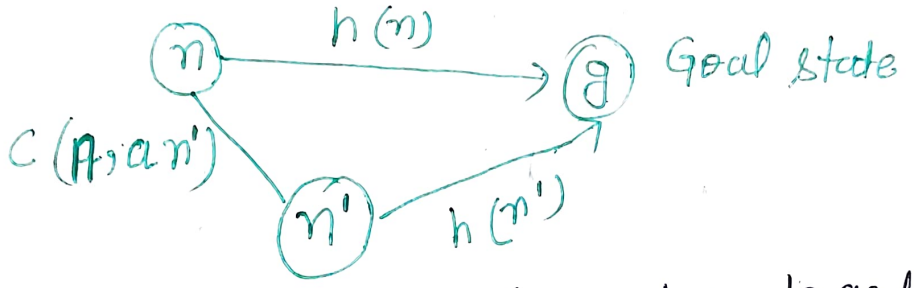
★ **Admissible Heuristic**

→ one never overestimates the cost to goal
i.e $0 \le h(n) \le h^*(n)$

★ **Consistency**

→ a heuristic is consistent for every node n & when we apply action "a" successor of node n is generated

$$h(n) \le C(n, a, n') + h(n')$$

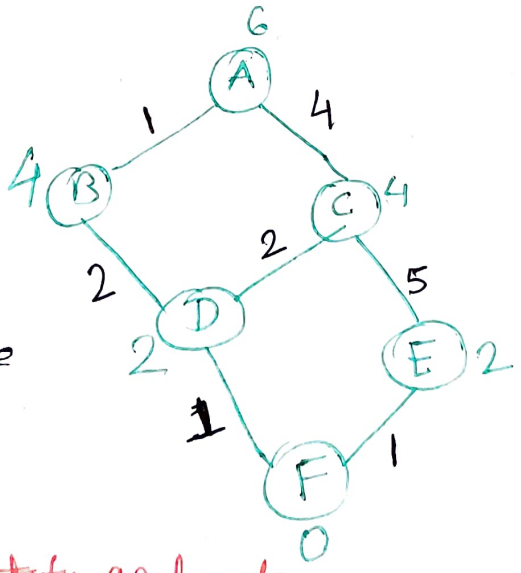


$h(n)$: estimated cost from node n to goal

$c(n\ a\ n')$: actual cost b/w node n & its successor n'

$h(n')$: estimated cost from node n' to goal.

exp :



Nos written on edge is the actual path cost from a node to neighbor node
$= C(n, a, n')$

value written on node is $h(n)$ = which is heuristic value from that node to goal node.

How to identify goal node

⟹ goal node whose heuristic value is 0

Note : In exam graph is given
- either they will provide 2 values ① heuristic value & ② actual path cost
- or Hint for calculating Actual path cost from the heuristic values
  - Manhattan dist
  - Euclidean dist

Solve :

Step 1 Initialization
open list : [A]
close list : [ ]

Step 2

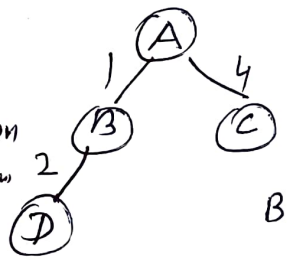
$f(A) = g(A,A) + h(A)$
$= 0 + 6 = 6$

$f(B) = g(A,B) + h(B)$
$= 1 + 4 = 5$

$f(C) = g(A,C) + h(C)$
$= 4 + 4$
$= 8$

open List : [B, C]
close List : [A]

Step 3 B will be explored as B's Evaluation function value i.e. f(n) is minimum

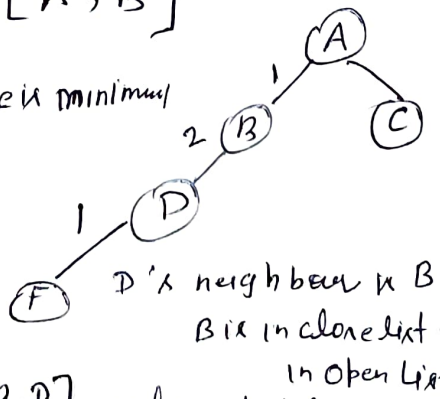


B's neighbour is A & D since A is in close list ignore it

$f(D) = g(B,D) + h(D)$
$= (1+2) + 2$
$= 5$

open List : [C, D]
close List : [A, B]

Step 4 Again D will be explored as D's Evaluation function value is minimum



$f(D) = g(D,F) + h(F)$
$= (1+2+1) + 0$
$= 4 + 0 = 4$

open List : [C, F]
close List : [A, B, D]

D's neighbour is B & C B is in close list & C is in open list ignore only node left = F

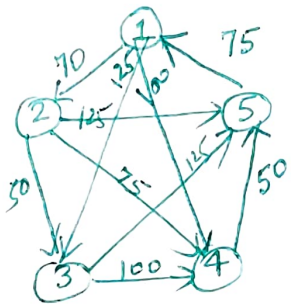
Step (5) Next explored Node is F

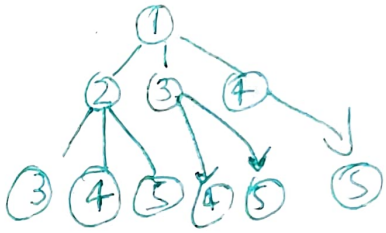
if F is goal node terminate the algorithm

Uninformed Search

- Uninformed search have No additional information on the goal node other than the one provided in problem definition
- this is called blind Search

(1) Breadth First Search (BFS)





Iter	Open List	Close List	Goal test
1	(1)	–	fail on (1)
2	~~(1)~~ (1,2) (1,3) (1,4)	(1)	fail on (1,2)
3	~~(1,2)~~ (1,3) (1,4) (1,2,3) (1,2,4) (1,2,5)	(1) (1,2)	fail on (1,3)
4	~~(1,3)~~ (1,4) (1,2,3) (1,2,4) (1,2,5) (1,3,4) (1,3,5)	(1) (1,2) (1,3)	fail on (1,4)
5	~~(1,4)~~ (1,2,3) (1,2,4) (1,2,5) (1,3,4) (1,3,5) (1,4,5)	(1) (1,2) (1,3) (1,4)	fail on (1,2,3)
6	~~(1,2,3)~~ (1,2,4) (1,2,5) (1,3,4) (1,3,5) (1,4,5)	(1) (1,2) (1,3) (1,4) (1,2,3)	fail on (1,2,4)
7	~~(1,2,4)~~ (1,2,5) (1,3,4) (1,3,5) (1,4,5)	(1) (1,2) (1,3) (1,4) (1,2,3) (1,2,4)	Pass on (1,2,5)