Solving Optimization Problems using Genetic Algorithms

There are four projects given below. For each of the following project, do the steps given below.

Step 1:
Clearly express the optimization problems in terms of (a) design variables, (b) objective function(s), (c) constraints to be satisfied, and (d) the range of design variables.

Step 2:
Your idea to solve the optimization problem. For example, which type of GA you wants to use, the schema of chromosomes, etc.

Step 3:
You can follow any programming environment, for example, Python, MATLAB, SPSS, etc. to implement your problem solving. Please state it clearly, the platform you have used.

Step 4:
There are a number of GA operators with a number of variations in them. It will be appreciated if you investigate the operators with different variations and report the results observed. The same is also true to GA parameters. You should run the program and furnish the results how it work with the values of the different GA parameters.

Step 5:
Write a report which would comprise all the steps you have followed (vis-à-vis the abve-mentioned steps). Also, include all the codes you have developed (or snapshots of the different screens of the tool you have used) in your report. Clearly mentioned any reasonable assumption(s) you make in your project.

Project GA-1: Bin Packing problem.

The **bin packing problems** is an optimization problem, in which items of different sizes must be packed into a finite number of bins or containers, each of a fixed given capacity, in a way that minimizes the number of bins used. The problem has many applications, such as filling up containers, loading trucks with weight capacity constraints, creating file backups in storage media and technology mapping in VLSI circuit design.

Problem description:
It is a classical optimization problems and known form early days of the computer science. For a detail description of the problem and it different variations, please see the Internet repository and precisely formulate the problem of your own.

Project GA-2: Reliable Network Design

Reliable network designs are based on sharing expensive hardware and software resources and provide access to the main server system from distant locations. The important step of network design process is to find the best layout of components to optimize certain

performance criteria like cost, reliability, transmission delay or throughput. The network design reliabilities are as follows:

- All terminal network reliability–probability that every node in the network is connected to each other.
- Source Sink Network Reliability–probability that the source is connected with the sink, so the source node in the network can communicate with the sink node over a specified mission time.

Genetic algorithm provides solution approaches for the optimal network design considering the above reliabilities into consideration. Following is a brief description of the optimization problem to be solved.

Problem description:

A computer communication network can be represented by an undirected graph $G=(N, E)$ in which nodes N and edges E represents computer sites and communication cables. A graph G is connected if there is at least one path between every pair of nodes i and j, which minimally requires a spanning tree with (n-1) edges. The number of possible edges is n(n-1)/2. The optimal design of all terminal network reliability is defined as follows:

n is the number of nodes
$x_{ij} \in \{0, 1\}$ is a decision variable representing edges between nodes i and j
$x = \{x12, x13, \ldots, xn\text{-}1, n\}$ is a topology architecture of network design
x^* is the best solution find so far
p is the edge reliability for all edges
q is edge unreliability for all edges (p+q = 1)
R(x) is the all terminal reliability of network design x
R_{min} is a network reliability requirement
$R_u(x)$ is the upper bound of reliability of the candidate network
c_{ij} is the cost of the edge between nodes i and j
c_{max} is the maximum value of c_{ij}
δ has a value of 1 if $R(x) < R_{min}$, else, has a value of 0
E' is a set of operational edges ($E' \subseteq E$)
Ω is all operational states.

Assume that the location of each node is given and nodes are perfectly reliable, each c_{ij} and p are fixed and known, each edge is bi-directional, there are no redundant edges in the network, edges are either operational or non-operational, the edge failures are mutually independent and there is no repair.

The optimal design of the network is represented as follows:

$$min \quad Z(x) = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} c_{ij} x_{ij} \qquad with \quad R(x) \geq R_{min}$$

At a particular time, only few edges of G might be operational. An operational state of G is a sub-graph G' = (V, E'). The network reliability of state E' $\subseteq$ E is as follows:

$$\sum_{\Omega}\left(\prod_{e\in E'} p_e\right)\left(\prod_{e\in E|E'} q_e\right)$$

Project GA-3: Job Shop Scheduling Problem

Scheduling, especially job shop scheduling, has been studying for a long time. Because of its NP-Hard nature, there has not been found a global problem solver for this kind of problems. Recently, some meta-heuristics like Simulated Annealing (SA), Taboo Search (TS), and Genetic Algorithms (GA) have been implemented as pure methods and hybrid of different method, where the hybrid methods are superior over pure ones. The main problem is how to cope with local minima within a reasonable time. Among them, GA has been studied and implemented to like the other problems with success.

The JSSP consists of a number of machines, M, and a number of jobs, J. Each job consists of M tasks, each of fixed duration. Each task must be processed on a

Problem Description:

The JSSP consists of a number of machines, M, and a number of jobs, J. Each job consists of M tasks, each of fixed duration. Each task must be processed on a

single specified machine, and each job visits each machine exactly once. There is a predefined ordering of the tasks within a job. A machine can process only one task at a time. There are no set-up times, no release dates and no due dates. The makespan is the time from the beginning of the first task to start to the end of the last task to finish. The aim is to find start times for each task such that the makespan is minimized. As a constraint problem, there are M*J variables, each taking positive integer values. The start time of *tth* task of the *jth* job will be denoted by *xjt*, and the duration of that task by *djt*. Each job introduces a set of *precedence* constraints on the tasks within that job: $xjt + djt \_ xj(t+1)$ for $t = 1$ to $M - 1$. Each machine imposes a set of *resource* constraints on the tasks processed by that machine: $xjt + djt \_ xpq$ *or* $xpq + dpq \_ xjt$. The aim is to find values for the variables such that no constraint is violated. By defining an objective function on assignments (which simply takes the maximum of $xjt + djt$), and attempting to minimize the objective, we get a constraint optimization problem.

Also, for more detail, see Internet documents.

Project GA-4: Optimum Cone Design

Let's introduce a geometrical optimization problem, named **cones problem**, with the following characteristics:

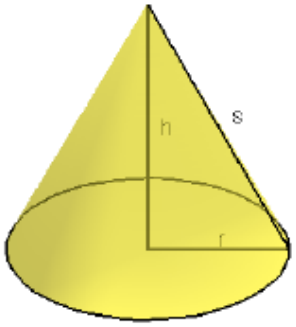
- **multi-objective** problem (two objective functions): the solution is not a single optimum design, but instead it is represented by the set of designs belonging to the *Pareto frontier*
- **simple** mathematical formulation: easy and quick implementation from scratch of the relevant mode.
- **constrained** problem: objectives space and designs space present *feasible* and *unfeasible* regions

Problem description:
Consider the following about a cone specification as. 3D object.

Right circular cone:

r = base radius
h = height
s = slant height

V = volume
B = base area
S = lateral surface area
T = total area



$$s = \sqrt{r^2 + h^2}$$

$$V = \frac{\pi}{3} r^2 h$$

$$B = \pi r^2$$

$$S = \pi r s$$

$$T = B + S = \pi r (r + s)$$

Following is a simple problems which has to be solved.

- two input variables: r, h

$$r \in [0, 10] \text{ cm}, \quad h \in [0, 20] \text{ cm}$$

The cone shape (i.e. the design) is defined univocally when both r and h are given.

- two objectives:

$$\min S$$
$$\min T$$

We want to minimize both the lateral surface area and the total surface area

- one constraint:

$$V > 200 \text{ cm}^3$$

A constraint for the cone volume is given, in order to guarantee a minimum volume.

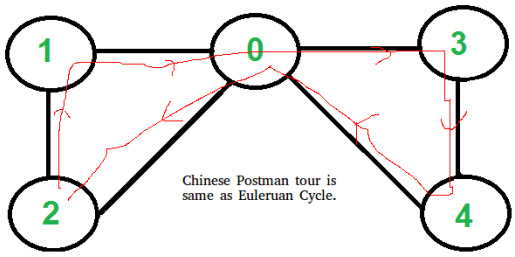
You have to solve the problem using NSGA-II algorithm. Obtain the graph showing all the feasible solution space. Also, you should clearly show the Pareto optimal font from it. Finally, report the trade-off solutions.

Project GA-5: Chinese Postman Problem

Chinese Postman Problem is a variation of the Eulerian circuit problem for undirected graphs. An Euler Circuit is a closed walk that covers every edge once starting and ending position is the same. Chinese Postman problem is defined as connected and undirected graphs. The problem is to find the shortest path or circuity that visits every edge of the graph at least once.

Case 1: Input graph contains a Euler circuit

If input graph contains Euler Circuit, then a solution of the problem is Euler Circuit. See Figure below.



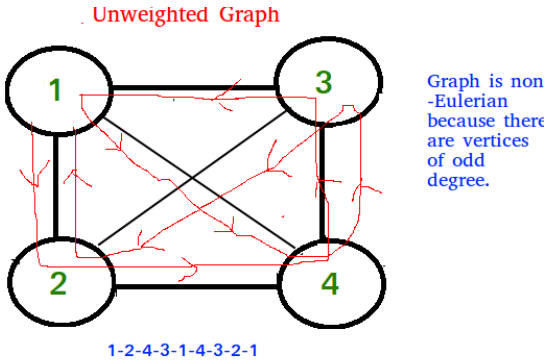
The graph has Eulerian Cycles, for example "2 1 0 3 4 0 2"
Note that all vertices have even degree

Note: It doesn't matter whether the graph is weighted or unweighted, the Chinese Postman Route is always the same as Eulerian Circuit if it exists. In the weighted graph, the minimum possible weight of the Postman tour is the sum of all edge weights which we get through the Eulerian Circuit.

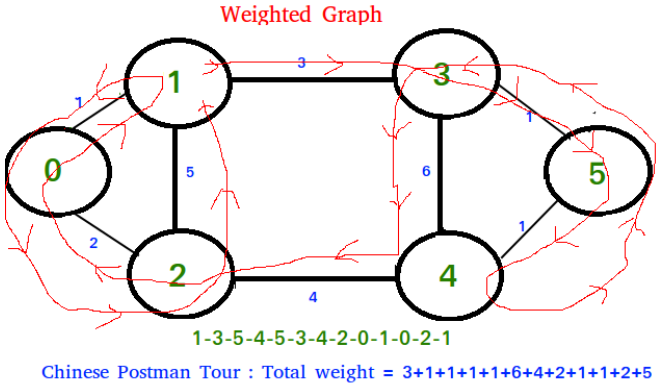
Case 2: If the input graph does NOT contain Euler Circuit

In this case, the task reduces to the following.

1) In an unweighted graph, there is a minimum number of edges to duplicate so that the given graph converts to a graph with the Eulerian Cycle. See the example given below.

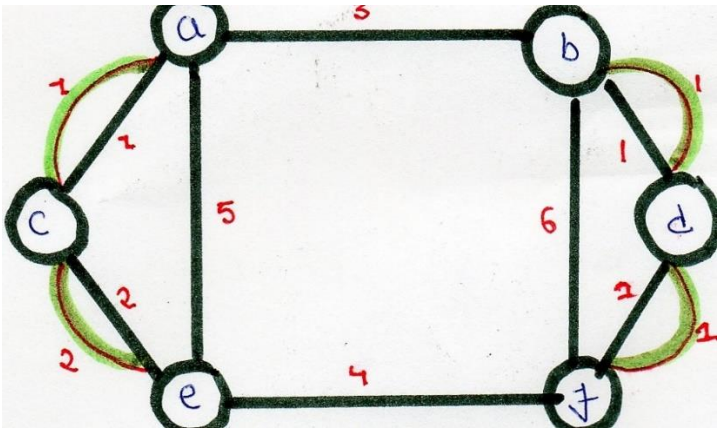


Case 3: The input graph does not contain an Euler circuit; however it can be converted to a graph with an Euler circle.
In a weighted graph, the minimum total weight of edges is to duplicate so that the given graph converts to a graph with the Eulerian Cycle.



Chinese Postman Route :
1 - 3 - 5 - 4 - 5 - 3 - 4 - 2 - 0 - 1 - 0 – 2 - 1

This route is Euler Circuit of the modified graph.



Solve this problem using Genetic algorithm.