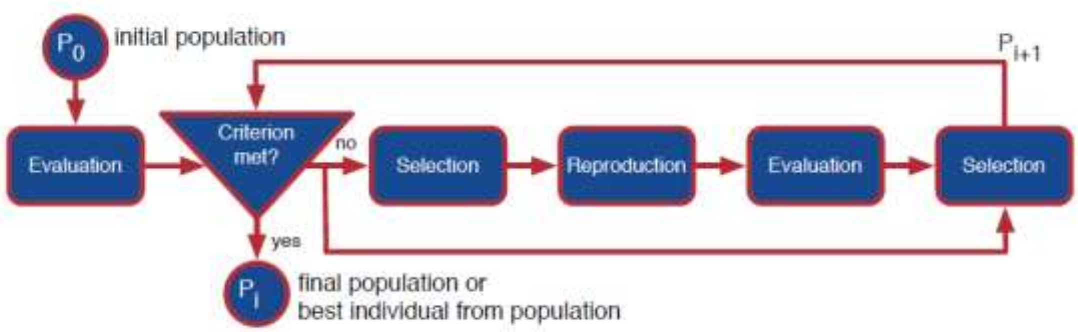
LAB 10 Genetic Algorithm

1.1 Genetic Algorithm

Genetic algorithms are usually used to identify optimal solutions to complex problems. This can clearly be easily mapped to search methods, which are aiming toward a similar goal. Genetic algorithms can thus be used to search for solutions to multi-value problems where the closeness of any attempted solution to the actual solution (**fitness**) can be readily evaluated. In short, a **population** of possible solutions (**chromosomes**) is generated, and a fitness value for each chromosome is determined. This fitness is used to determine the likelihood that a given chromosome will survive to the next generation, or reproduce. Reproduction is done by applying **crossover** to two (or more) chromosomes, whereby features (**genes**) of each chromosome are combined together. Mutation is also applied, which involves making random changes to particular genes.



1.2 Lab Tasks

Exercise 10.1.

Consider the problem of maximizing the function

$$f(x) = \frac{-x^2}{10} + 3x$$

where x is allowed to vary between 0 and 31. You must perform following tasks in the code.

a. Representation of states (solutions): To solve this using a genetic algorithm, we must encode the possible values of x as chromosomes. For this problem, we will encode x as a binary integer of length 5. Thus the chromosomes for our genetic algorithm will be sequences of 0's and 1's with a length of 5 bits, and have a range from 0 (00000) to 31 (11111).

b. Fitness function:

The fitness function for it will be:

$$f(x) = \frac{-x^2}{10} + 3x$$

To begin the algorithm, we select an initial population of 10 chromosomes at random. The resulting initial population of chromosomes is shown in Table 1. Next we take the x-value that each chromosome represents and test its fitness with the fitness function. The resulting fitness values are recorded in the third column of Table 1.

Chromosome Number	**Initial Population**	**x-Value**	**Fitness Value $f(x)$**	**Selection Probability**
1	0 1 0 1 1	11	20.9	0.1416
2	1 1 0 1 0	26	10.4	0.0705
3	0 0 0 1 0	2	5.6	0.0379
4	0 1 1 1 0	14	22.4	0.1518
5	0 1 1 0 0	12	21.6	0.1463
6	1 1 1 1 0	30	0	0
7	1 0 1 1 0	22	17.6	0.1192
8	0 1 0 0 1	9	18.9	0.1280
9	0 0 0 1 1	3	8.1	0.0549
10	1 0 0 0 1	17	22.1	0.1497

c. **Operators:**
   i. Apply cross over in every generation.
   ii. Apply mutation after every 3 generations.

d. **Termination criteria:** Your loop should stop when the value of one of your candidate's fitness function is greater or equal to 90%.

Exercise 10.2. Travelling Salesman Problem

Suppose a TCS delivery boy has to deliver parcels from *Head Office (MM Alam Road)* to 7 different locations in Lahore *(Johar Town, Shahdara, DHA Phase 6, Wapda Town, Askari 10, Allama Iqbal Town, Mall Road)* and then return back to the *Head Office.* He wants to find the route with least travelling distance. You helped him in finding the route using *Hill Climbing Algorithm*. Now use *Genetic Algorithm* instead of *Hill Climbing* to solve this problem. Design choices should be as per class discussion. [You can construct distance matrix using google maps OR you can take random values (between 0-50) for distances between any two spots.]

Exercise 10.3.

For any given 'target' number, find an expression involving any legal combination of addition(+), subtraction(-), multiplication(*) and division(/) on digits that represents the given target number. For example, for the target 15, "9 + 3 * 2" is a solution."

Exercise 10.4.

n-Queen Problem using Genetic Algorithm.

Exercise 10.5.

You have 10 cards numbered 1 to 10. You have to divide them into two piles so that:

- The sum of the first pile is as close as possible to 36.
- And the product of all in the second pile is as close as possible to 360.

Well, all that is being done is the following :

Loop through the population member's genes

- If the current gene being looked at has a value of 0, the gene is for the sum pile (pile 0), so add to the running calculation
- If the current gene being looked at has a value of 1, the gene is for the product pile (pile 1), so add to the running calculation
- Calculate the overall error for this population member. If this member's geneotype has an overall error of 0.0, then the problem domain has been solved