**Name: Jigar Siddhpura SAPID:** 60004200155

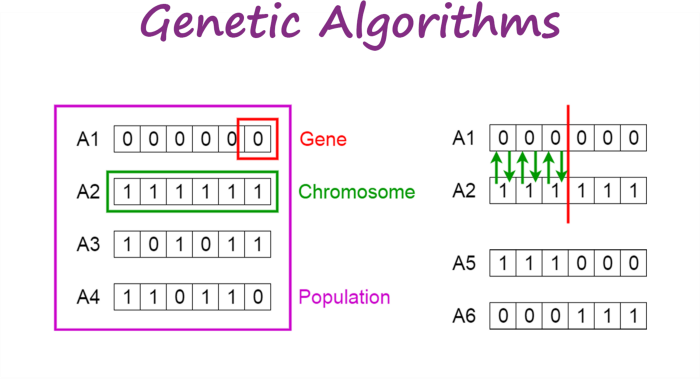
**DIV: C/C2** **Branch:** Computer Engineering

AI EXPERIMENT 5 - Genetic Algorithm

**Aim:** To study and implement Genetic Algorithm

**Theory:**

A genetic algorithm is a search heuristic that is inspired by Charles Darwin’s theory of natural evolution. This algorithm reflects the process of natural selection where the fittest individuals are selected for reproduction in order to produce offspring of the next generation.



Five phases are considered in a genetic algorithm.

1. Initial population
2. Fitness function
3. Selection
4. Crossover
5. Mutation

# Initial Population

The process begins with a set of individuals which is called a Population. Each individual is a solution to the problem you want to solve.An individual is characterized by a set of parameters (variables) known as Genes. Genes are joined into a string to form a Chromosome (solution).

# Fitness Function

# The fitness function determines how fit an individual is (the ability of an individual to compete with other individuals). It gives a fitness score to each individual. The probability that an individual will be selected for reproduction is based on its fitness score.

# Selection

# The idea of selection phase is to select the fittest individuals and let them pass their genes to the next generation.Two pairs of individuals (parents) are selected based on their fitness scores. Individuals with high fitness have more chance to be selected for reproduction.

# Crossover

# Crossover is the most significant phase in a genetic algorithm. For each pair of parents to be mated, a crossover point is chosen at random from within the genes.

# Mutation

# In certain new offspring formed, some of their genes can be subjected to

a mutation with a low random probability. This implies that some of the bits in the bit string can be flipped.

**Code :**

import random

gene = ['000000','111111','101011','110110']

def selection(gene):

    x = [int(x,2) for x in gene]

    fx = [int(x)\*int(x) for x in x]

    fx\_sum = sum(fx)

    fx\_avg = fx\_sum//len(fx)

    excepted\_count = [round((i/fx\_avg),4) for i in fx]

    actual\_count = [round(i) for i in excepted\_count]

    mate\_pool = []

    for i,j in zip(actual\_count,gene):

        if i:

            for c in range(i):

                mate\_pool.append(j)

return x,fx, fx\_sum,fx\_avg,excepted\_count,actual\_count,mate\_pool

def generate\_mate(size,mate\_element\_size):

    if size % 2 != 0:

        return -1

    available\_positions = list(range(size))

    random.shuffle(available\_positions)

    mate = [-1] \* size

    crossover =[-1] \* size

    for i in range(size):

        if mate[i] == -1:

            j = random.choice(available\_positions)

            while mate[j] != -1 or j == i:

                j = random.choice(available\_positions)

            mate[i] = j

            mate[j] = i

            available\_positions.remove(i)

            available\_positions.remove(j)

    for i in mate:

        if crossover.count(-1) != 0:

            crossover[i] = crossover[mate[i]] = random.randint(1,mate\_element\_size-1)

    return mate,crossover

def crossover(mate\_pool):

    mate, crossover\_points = generate\_mate(len(mate\_pool),len(mate\_pool[0]))

    new\_poplu = [-1]\*len(mate\_pool)

    for i in mate:

        new\_poplu[i] = mate\_pool[i][:crossover\_points[i]] + mate\_pool[mate[i]][crossover\_points[i]:]

    x = [int(x,2) for x in new\_poplu]

    fx = [int(x)\*int(x) for x in x]

    return mate\_pool,new\_poplu,mate,crossover\_points,x, fx

def GA(gene,iter,n):

    if iter == 0:

        return

    x,fx, fx\_sum,fx\_avg,excepted\_count,actual\_count,mate\_pool = selection(gene)

    if sum(actual\_count)!=len(gene):

        print("Error dont know what to do at this situation ")

        return

    print(f"\n------------------------------------------------- GENERATION {n} --------------------------------------------------")

    print("Initial Population\tX Value\t\tFitness Value( f(x) )\tProbability(Expected Count)\tActual Count")

    print(f"-----------------------------------------------------------------------------------------------------------------")

    for i in range(len(gene)):

        print(f"{gene[i]}\t\t\t{x[i]}\t\t{fx[i]}\t\t\t{excepted\_count[i]}\t\t\t\t{actual\_count[i]}")

    mate\_pool,new\_poplu,mate,crossover\_points,x, fx = crossover(mate\_pool)

    print(f"\n----------------------------------------------- New Population {n} ------------------------------------------------")

    print("Mate Pool\tMate\t\tCrossover Points\tNew Population\t\tx value\t\tf(x)")

    print(f"-----------------------------------------------------------------------------------------------------------------")

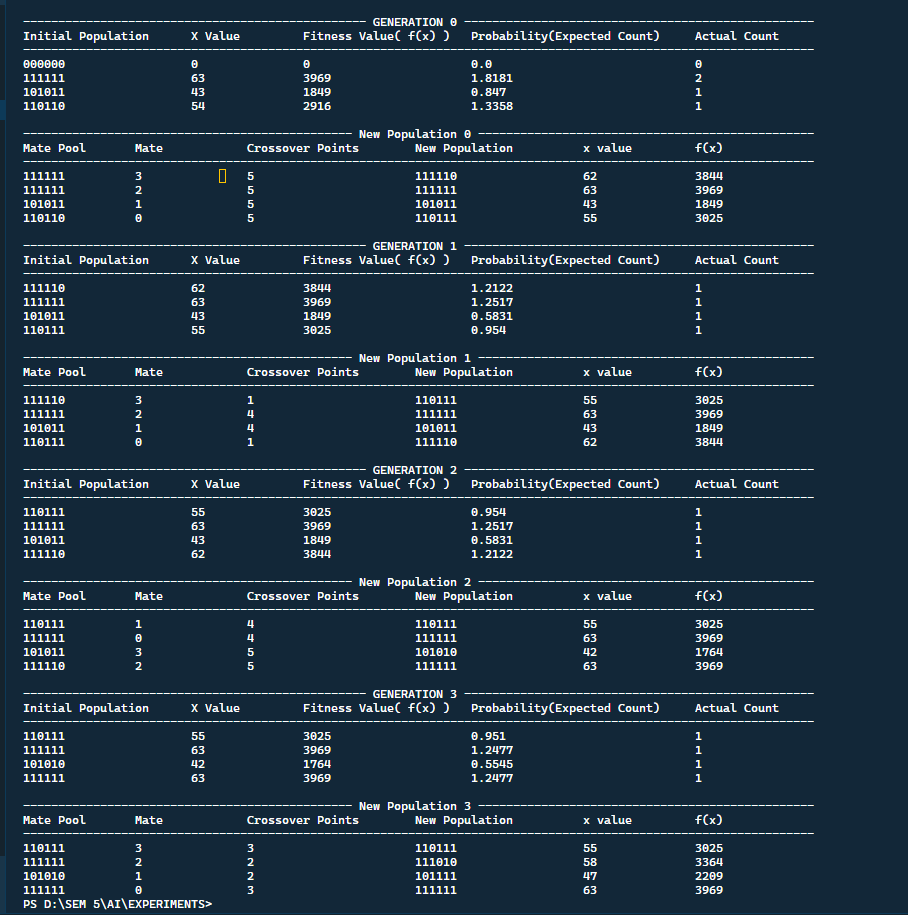
    for i in range(len(gene)):

        print(f"{mate\_pool[i]}\t\t{mate[i]}\t\t{crossover\_points[i]}\t\t\t{new\_poplu[i]}\t\t\t{x[i]}\t\t{fx[i]}")

    GA(new\_poplu,iter-1,n+1)

GA(gene,4,0)

**Output :**



**Conclusion :**

Thus we successfully studied and applied Genetic Algorithm