Indian Institute of Technology, Guwahati

Department Of Mechanical Engineering

SOFT COMPUTING (ME674)

Coding Assignment 2

Submitted To:
Prof. Sukhomay Pal

Submitted By:
Mritunjay
Roll No. 214103004
M.Tech -AP

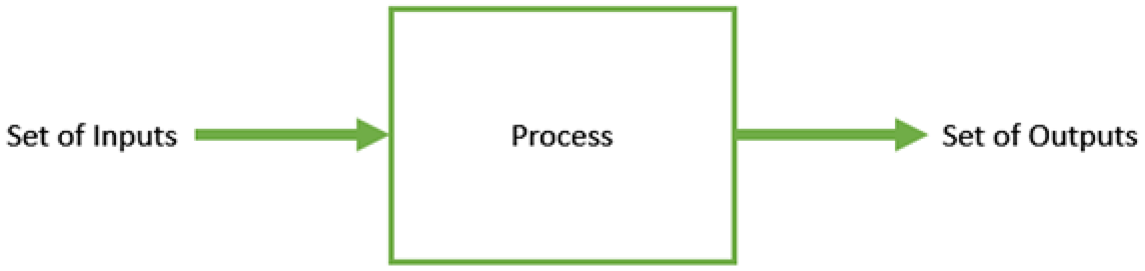
Table of Content

1. Introduction

Genetic Algorithm (GA) is a search-based optimization technique based on the principles of **Genetics and Natural Selection**. It is frequently used to find optimal or near-optimal solutions to difficult problems which otherwise would take a lifetime to solve. It is frequently used to solve optimization problems, in research, and in machine learning.

Introduction to Optimization

Optimization is the process of **making something better**. In any process, we have a set of inputs and a set of outputs as shown in the following figure.



Optimization refers to finding the values of inputs in such a way that we get the "best" output values. The definition of "best" varies from problem to problem, but in mathematical terms, it refers to maximizing or minimizing one or more objective functions, by varying the input parameters.

The set of all possible solutions or values which the inputs can take make up the search space. In this search space, lies a point or a set of points which gives the optimal solution. The aim of optimization is to find that point or set of points in the search space.

What are Genetic Algorithms?

Nature has always been a great source of inspiration to all mankind. Genetic Algorithms (GAs) are search based algorithms based on the concepts of natural selection and genetics. GAs are a subset of a much larger branch of computation known as **Evolutionary Computation**.

GAs were developed by John Holland and his students and colleagues at the University of Michigan, most notably David E. Goldberg and has since been tried on various optimization problems with a high degree of success.

In GAs, we have a **pool or a population of possible solutions** to the given problem. These solutions then undergo recombination and mutation (like in natural genetics), producing new children, and the process is repeated over various generations. Each individual (or candidate solution) is assigned a fitness value (based on its objective function value) and the fitter individuals are given a higher chance to mate and yield more "fitter" individuals. This is in line with the Darwinian Theory of "Survival of the Fittest".

In this way we keep "evolving" better individuals or solutions over generations, till we reach a stopping criterion.

Genetic Algorithms are sufficiently randomized in nature, but they perform much better than random local search (in which we just try various random solutions, keeping track of the best so far), as they exploit historical information as well.

Advantages of GAs

GAs have various advantages which have made them immensely popular. These include −

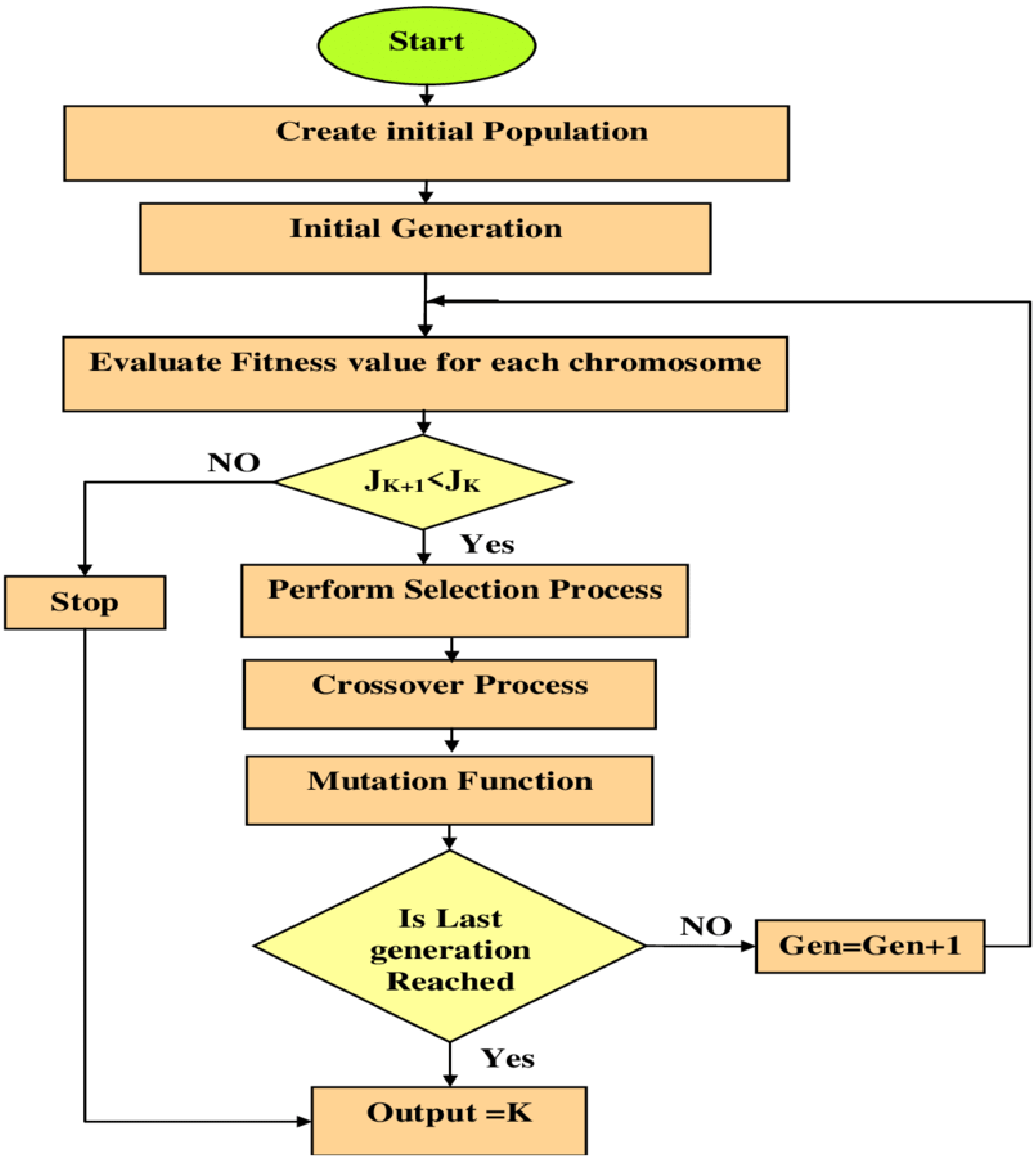
- Does not require any derivative information (which may not be available for many real-world problems).
- Is faster and more efficient as compared to the traditional methods.
- Has very good parallel capabilities.
- Optimizes both continuous and discrete functions and also multi-objective problems.
- Provides a list of “good” solutions and not just a single solution.
- Always gets an answer to the problem, which gets better over the time.
- Useful when the search space is very large and there are a large number of parameters involved.

Limitations of GAs

Like any technique, GAs also suffer from a few limitations. These include −

- GAs are not suited for all problems, especially problems which are simple and for which derivative information is available.
- Fitness value is calculated repeatedly which might be computationally expensive for some problems.
- Being stochastic, there are no guarantees on the optimality or the quality of the solution.
- If not implemented properly, the GA may not converge to the optimal solution.

2. Flow chart


Start
Create initial Population
Initial Generation
Evaluate Fitness value for each chromosome
NO
$J_{K+1} < J_K$
Yes
Stop
Perform Selection Process
Crossover Process
Mutation Function
Is Last generation Reached
NO
Gen=Gen+1
Yes
Output =K

3. Coding Section

PYTHON CODE for GA

```
import numpy as np
import random

"""
**********Problem statement*************

Minimize F(x1,x2)=x1+x2-2*x1**2-x2**2+x1*x2
0.0<=x1,x2<=0.5
N=6
pc=1.0
5bits for each variable
"""

# Initialization
bits = 10
N = 6
S = [np.random.randint(0, 2, bits).tolist() for _ in range(N)]
G=int((input("Enter No. of Generation")))

# Function definition
def func(x):
    X1 = int("".join(list(map(str, x[:5]))), 2)
    x1 = (0.5 / 31) * X1
    X2 = int("".join(list(map(str, x[5:]))), 2)
    x2 = (0.5 / 31) * X2
    F = x1 + x2 - 2 * x1 * x1 - x2 * x2 + x1 * x2
    return 1 / (1 + F)

# CrossOver
def crossover(p1, p2):
    pt = np.random.randint(1, len(p1) - 2)
    c1 = p1[:pt] + p2[pt:]
    c2 = p2[:pt] + p1[pt:]
    return [c1, c2]


# Mutation
def mutation(bit, r_mut):
    for i in range(len(bit)):
        if np.random.random_sample() < r_mut:
            bit[i] = 1 - bit[i]


for gen in range(G):

    scores = [func(c) for c in S]
    sum_scores = sum(scores)
    prob = [x / sum_scores for x in scores]
    cummuprob = [sum(prob[:i + 1]) for i in range(len(prob))]
    random_prob = [np.random.random_sample() for _ in range(len(S))]
    MP = []
    for x in random_prob:
        for i in range(len(cummuprob)):
            if cummuprob[i] >= x:
                MP.append(i)
                break
    parent = [S[i] for i in MP]
    child = []
    for i in range(0, len(S), 2):
        p1, p2 = parent[i], parent[i + 1]
        for c in crossover(p1, p2):
            mutation(c, 0.0)
            child.append(c)
    S = child

# Printing Results
print("For", gen + 1, "No. of Generations:")
print("x1 = ", (0.5 / 31) * int("".join(list(map(str, S[0][:5]))), 2))
print("x2 = ", (0.5 / 31) * int("".join(list(map(str, S[0][5:]))), 2))
```

References

- https://www.sciencedirect.com/topics/engineering/genetic-algorithm-method
- https://www.google.com/search?q=GA+flow+chart&sxsrf=ALiCzsYmVXdfU5vulyemI_W4toqS9Ynozw:1652351186834&tbm=isch&source=iu&ictx=1&vet=1&fir=u5wTRZVJFA0onM%252Cuy7jGTOtyk6g1M%252C_%253B0ug4Ul4hm_suYM%252Cu9IWm_2k1IOFbM%252C_%253Bug1RChZHAmn-KM%252ChyQUUpjJ49zpYM%252C_%253BRc8j5A4LgPPp0M%252C0lo2Q8dUPv4csM%252C_%253Bi0n0b3xLJ9AlvM%252CfakmQ5bS_RgxIM%252C_%253BipwDaanwIeUA4M%252Ces6OpvjGEeMcTM%252C_%253Bvd-bO_UecfajDM%252C0lo2Q8dUPv4csM%252C_%253Bm2TVDuhRqkcNfM%252Ce-3bAc_RldL77M%252C_%253BcbCgLM__TZRS9M%252CnBP4GSeKnezfbM%252C_%253B_AWwMnfAe_RBAM%252CxQ2k2NpZXnjk9M%252C_%253BEcabvrs8oSyRaM%252CsKNPgnkMXjjZFM%252C_%253BVdgtusHCjvw2XM%252CbDhsx8tgF0rd1M%252C_%253BYO4teRZO-f6jJM%252CKh8xrZppcarxcM%252C_%253BN_b9S2wBFNazaM%252CrLvzENYakgUatM%252C_%253BRYWlfg58xRAqlM%252C4c5PiUOYr69DcM%252C_%253B-Jkp83SXkLU0SM%252CTxejn9d6lDuuyM%252C_%253Be7RGPogN_X9J-M%252CuisPYlDkvkJa7M%252C_%253BWLug3E9y4UVXUM%252CDNPsW6AOfos_DM%252C_%253BqUeS2CaQNm8qYM%252CMIu11DSYLTyd-M%252C_&usg=AI4_-kR6XhSbGr3Hv7sF9zxmy00uAqHhmA&sa=X&ved=2ahUKEwii-aPs39n3AhUoIbcAHTY9BQYQ9QF6BAgCEAE#imgrc=-YaOEGvldrtryM