A blue and white logo

Description automatically generated

**COMSATS UNIVERSITY ISLAMABAD, ABBOTTABAD CAMPUS**

**Lab Assignment # 01 – Artificial Intelligence**

***Submitted by:***

Hanzala Nouman FA21-BSE-015-6A

Laiba Binte Tahir FA21-BSE-019-6A

Arfah Ali FA21-BSE-080-6A

***Submitted To:***

Dr. Mubashir Ahmad

**Maximize** the value of the function F(X) = - X2 + 2X, over the range of real number from 0 to 2 with initial population ['11010', '00111', '10110', '00101'] and with random numbers [0.4, 0.15, 0.7, 0.9] adjust the numbers in range of 0 to 2. Select the crossover between the first and fifth digit. Run the algorithm in 2 iterations.

**Code:**

"""Created on Mon Apr 29 11:56:03 2024

@author: Laiba Binta Tahir

"""

import numpy as np

import matplotlib.pyplot as plt

def FitnessFunction(x):

return -x\*\*2 + 2\*x

def DecodeBinary(binary\_str):

return int(binary\_str, 2)

def AdjustValue(binary\_str, MinVal, MaxVal):

return MinVal + (MaxVal - MinVal) \* DecodeBinary(binary\_str) / (2\*\*len(binary\_str) - 1)

def CrossOver(p1, p2):

c1 = p1[0] + p2[1:4] + p1[4]

c2 = p2[0] + p1[1:4] + p2[4]

return c1, c2

#given data

Population = ['11010', '00111', '10110', '00101']

randNumbers = [0.4, 0.15, 0.7, 0.9]

# decode

DecodePopulation = [DecodeBinary(individual) for individual in Population]

AdjustedPopulation = [AdjustValue(individual, 0, 2) for individual in Population]

# Fitness

FitnessValues = [FitnessFunction(adjusted) for adjusted in AdjustedPopulation]

print("\n")

print("1st Generation Indiviuals:")

for j , (individual , decoded , adjusted, fitness) in enumerate(zip(Population, DecodePopulation, AdjustedPopulation, FitnessValues) ,1):

print(f"String {j}: Binary: {individual}, Decoded: {decoded}, Adjusted: {adjusted:.10f}, Fitness: {fitness:.10f}")

# PDF & CDF

TotalFitness = sum(FitnessValues)

prob = [fitness / TotalFitness for fitness in FitnessValues]

CDF = np.cumsum(prob)

# select string

SelectedIndiviuals = []

for rNum in randNumbers:

SelectedIndiviual = next(j for j , cdfVal in enumerate(CDF) if cdfVal >= rNum)

SelectedIndiviuals.append(SelectedIndiviual)

print("\nSelected Strings for Random Numbers:")

for i, rNum, SelectedIndiviual in zip(range(1, 5), randNumbers, SelectedIndiviuals):

print(f"Randon Num {rNum}, String: {SelectedIndiviual + 1}")

# crossover

c1, c2 = CrossOver(Population[SelectedIndiviuals[0]], Population[SelectedIndiviuals[1]])

c3, c4 = CrossOver(Population[SelectedIndiviuals[2]], Population[SelectedIndiviuals[3]])

newPopulatoion = [c1,c2,c3,c4]

#calculations for new generation

DecodeNewPopulation = [DecodeBinary(individual) for individual in newPopulatoion]

AdjustedNewPopulation = [AdjustValue(individual, 0, 2) for individual in newPopulatoion]

NewFitnessValues = [FitnessFunction(adjusted) for adjusted in AdjustedNewPopulation]

print("\nNew Population - After Crossover:")

for j, (individual, decoded, adjusted, fitness) in enumerate(zip(newPopulatoion, DecodeNewPopulation, AdjustedNewPopulation, NewFitnessValues), 1):

print(f"String {j}: Binary: {individual}, Decoded: {decoded}, Adjusted: {adjusted:.10f}, Fitness: {fitness:.10f}")

print("\n")

#Plot

MaxFitnessInitial = np.max(FitnessValues)

MaxFitnessNew = np.max(NewFitnessValues)

Generations = np.arange(1.0, 2.01, 0.2)

yAxis = np.arange(round(MaxFitnessInitial, 2), round(MaxFitnessNew, 2), 0.02 )

plt.plot([1.0,2.0] , [MaxFitnessInitial, MaxFitnessNew], '-ro', color="purple" )

plt.xticks(Generations)

plt.xlabel("Generations")

plt.yticks(yAxis)

plt.ylabel("Maximum Fitness")

plt.title("Fitness Per Generation")

plt.grid(True)

plt.show

**Output:**

A graph with a purple line

Description automatically generated

