COMP 6731

Assignment 2:

Answers:

-The encoding of the problem (given in the hints below)

https://lh4.googleusercontent.com/bQCyazAMDvkIGqm35ilEENFKq-DGPr0k_z9X_21q-8MqJHZbqNZQ3zYaAUVawdADBDpV4NO0akQi3w3xJfLhadK5hmEw38e8R-i7k9d4MUo1T2VATcu3gFBSWhr1sDmu5aXIJJ98

https://lh5.googleusercontent.com/TnwzZSFUcrJDVw50Ww5vFwDF8xulps_5g3BUtkNXn68iK7Cj4CDqIxnZQ4Lfcxros7iaLLekrIDkobJslyA_oJUkJ--55LQcI1rxekxqaDqExa8eXw3MSej2hY7memSkXTqVt4qM

Variables: x1, x2, x3 and x4 are binary variables, representing the investment decisions of project 1, 2, 3, and 4. A value 1 indicating invest while a value 0 indicating don’t invest.

-The crossover mechanism used

Using the random function produce the point to do crossover. And define the exchange function to finish the crossover actions.

-The mutation used

For each chromosome generates their corresponding possibilities to mutate, if the possibilities is more than 0.5 then change these genes.

-The fit function used

  Fitness = 0.2\*x1+0.3\*x2+0.5\*x3+0.1\*x4

-The selection of each generation

If the generation satisfies the constraints and didn’t exist in the population then it can be added to the population. https://lh5.googleusercontent.com/TnwzZSFUcrJDVw50Ww5vFwDF8xulps_5g3BUtkNXn68iK7Cj4CDqIxnZQ4Lfcxros7iaLLekrIDkobJslyA_oJUkJ--55LQcI1rxekxqaDqExa8eXw3MSej2hY7memSkXTqVt4qM

- The stopping criteria

When the population stops genarates new offsprings.

My program’s pipline:

START

Initial population

Compute fitness

REPEAT(while population grows)

Select Top2 persons as parents

Crossover----offsprings

Mutation

End loop

Show(result)

Code show:

import numpy as np

from random import randint

#1.Initial population

N = 4

Initial = np.zeros(N)

Initial\_1 = np.ones(N)

population=[Initial\_1,Initial]

def select\_Top2(population):

  max\_fit = 0

  second\_max = 0

  max\_item = []

  second\_item=[]

  coefficent = np.array([0.2, 0.3, 0.5, 0.1])

  for item in population:

      Fitness = np.sum(item\*coefficent)

      if Fitness > max\_fit:

          max\_fit = Fitness

          max\_item = item

      elif max\_fit >= Fitness>= second\_max:

          second\_max = Fitness

          second\_item = item

  return max\_fit,max\_item,second\_max,second\_item

def Crossover(Parent1,Parent2):

  cross\_point = randint(1, N - 1)

  offSpring1 = np.append(Parent1[:cross\_point],Parent2[cross\_point:])

  offSpring2 = np.append(Parent2[:cross\_point], Parent1[cross\_point:])

  print("The Parent\_1 and Parent\_2 is:",Parent1,Parent2)

  print("The crossovered offSpring is:", offSpring1,offSpring2)

  return offSpring1,offSpring2

def Exist\_or\_not(Chromosome,Population):

  flag =True

  for x in Population:

      if np.array\_equal(x, Chromosome):

          flag =False

  if(flag):

      Population.extend([Chromosome])

  return Population

def mutation(offspring\_crossover):

  mut = np.random.rand(4,1)

  print('The mutation number is:',mut)

  offspring\_crossover\_new=np.zeros(N)

  for index, item in enumerate(mut):

      if item>0.5:

         offspring\_crossover\_new[index]=1-offspring\_crossover[index]

      else:

         offspring\_crossover\_new[index] = offspring\_crossover[index]

  print("The mutated chromosome is:",offspring\_crossover\_new)

  return offspring\_crossover\_new

def Population\_select(population):

  max=0

  new\_population = []

  for item in population:

      print("The item is:",item)

  #for item in population:

      equation1=np.array([0.5,1,1.5,0.1])

      equation2=np.array([0.3,0.8,1.5,0.4])

      equation3=np.array([0.2,0.2,0.3,0.1])

      fit\_coefficent =np.array([0.2,0.3,0.5,0.1])

      reward1 = np.sum(item\*equation1)

      reward2 = np.sum(item\*equation2)

      reward3 = np.sum(item\*equation3)

      Fitness = np.sum(item\*fit\_coefficent)

      print("The Fitness :",Fitness)

      if((reward1<=3.1)&(reward2<=2.5)&(reward3<=0.4)):

          new\_population.append([item])

          if Fitness>max:

              max = Fitness

              Fitness\_final= Fitness

              max\_item =item

  return max\_item,Fitness\_final,new\_population

if \_\_name\_\_ == '\_\_main\_\_':

  #2.Fitness function

  x1,x2,x3,x4 = population[1]

  Fitness = 0.2\*x1+0.3\*x2+0.5\*x3+0.1\*x4

  head\_count = 2

  head\_count\_before = 0

  while(head\_count!=head\_count\_before):

      head\_count\_before=head\_count

      head\_count=0

#select function

      max\_fit,max\_item,second\_max,second\_item = select\_Top2(population)

      mut\_parent\_1 = mutation(max\_item)

      mut\_parent\_2 = mutation(second\_item)

      population = Exist\_or\_not(mut\_parent\_1, population)

      population = Exist\_or\_not(mut\_parent\_2, population)

#3.crossover. Offspring

      offspring\_A,offspring\_B = Crossover(max\_item,second\_item)

      print("The generated children1 is :",offspring\_A)

      print("The generated children2 is :",offspring\_B)

      population = Exist\_or\_not(offspring\_A, population)

      population = Exist\_or\_not(offspring\_B, population)

#4.mutation

      mut\_offsp\_A=mutation(offspring\_A)

      mut\_offsp\_B=mutation(offspring\_B)

      population = Exist\_or\_not(mut\_offsp\_A, population)

      population = Exist\_or\_not(mut\_offsp\_B, population)

      head\_count =np.shape(population)[0]

#5.stopping criterion

  item,Fitness,new\_poputalion = Population\_select(population)

  head\_count = np.shape(new\_poputalion)[0]

  print("The population is:", new\_poputalion)

  print("Best solution : ",item)

  print("Best solution fitness : ",Fitness)

  print("The size of population: ", head\_count)

Result show:

The mutation number is: [[0.35414245]

[0.66787395]

[0.85878251]

[0.25594785]]

The mutated chromosome is: [1. 0. 0. 1.]

The mutation number is: [[0.05852127]

[0.23482008]

[0.41753823]

[0.76137166]]

The mutated chromosome is: [0. 0. 0. 1.]

The Parent\_1 and Parent\_2 is: [1. 1. 1. 1.] [0. 0. 0. 0.]

The crossovered offSpring is: [1. 0. 0. 0.] [0. 1. 1. 1.]

The generated children1 is : [1. 0. 0. 0.]

The generated children2 is : [0. 1. 1. 1.]

The mutation number is: [[0.39973943]

[0.57866735]

[0.12330939]

[0.28026844]]

The mutated chromosome is: [1. 1. 0. 0.]

The mutation number is: [[0.32312104]

[0.47487257]

[0.90447784]

[0.84168867]]

The mutated chromosome is: [0. 1. 0. 0.]

The mutation number is: [[0.86159077]

[0.10901521]

[0.37866373]

[0.73791822]]

The mutated chromosome is: [0. 1. 1. 0.]

The mutation number is: [[0.54591967]

[0.35769192]

[0.18239112]

[0.72058948]]

The mutated chromosome is: [1. 1. 1. 0.]

The Parent\_1 and Parent\_2 is: [1. 1. 1. 1.] [0. 1. 1. 1.]

The crossovered offSpring is: [1. 1. 1. 1.] [0. 1. 1. 1.]

The generated children1 is : [1. 1. 1. 1.]

The generated children2 is : [0. 1. 1. 1.]

The mutation number is: [[0.13668072]

[0.34271941]

[0.79759405]

[0.8184857 ]]

The mutated chromosome is: [1. 1. 0. 0.]

The mutation number is: [[0.72395254]

[0.28074132]

[0.6059823 ]

[0.99940089]]

The mutated chromosome is: [1. 1. 0. 0.]

The mutation number is: [[0.44679739]

[0.53990148]

[0.78709898]

[0.93922364]]

The mutated chromosome is: [1. 0. 0. 0.]

The mutation number is: [[0.96559046]

[0.56811754]

[0.01295442]

[0.33449066]]

The mutated chromosome is: [0. 0. 1. 0.]

The Parent\_1 and Parent\_2 is: [1. 1. 1. 1.] [1. 1. 1. 0.]

The crossovered offSpring is: [1. 1. 1. 0.] [1. 1. 1. 1.]

The generated children1 is : [1. 1. 1. 0.]

The generated children2 is : [1. 1. 1. 1.]

The mutation number is: [[0.50667966]

[0.36017459]

[0.33653424]

[0.10285106]]

The mutated chromosome is: [0. 1. 1. 0.]

The mutation number is: [[0.89168643]

[0.22462934]

[0.82739422]

[0.03351929]]

The mutated chromosome is: [0. 1. 0. 1.]

The mutation number is: [[0.78193899]

[0.13193795]

[0.33422511]

[0.01782471]]

The mutated chromosome is: [0. 1. 1. 1.]

The mutation number is: [[0.68037301]

[0.38992382]

[0.21394042]

[0.45502234]]

The mutated chromosome is: [0. 1. 1. 0.]

The Parent\_1 and Parent\_2 is: [1. 1. 1. 1.] [1. 1. 1. 0.]

The crossovered offSpring is: [1. 1. 1. 0.] [1. 1. 1. 1.]

The generated children1 is : [1. 1. 1. 0.]

The generated children2 is : [1. 1. 1. 1.]

The mutation number is: [[0.8937968 ]

[0.61425318]

[0.18073581]

[0.90038667]]

The mutated chromosome is: [0. 0. 1. 1.]

The mutation number is: [[0.49027004]

[0.48158895]

[0.80048247]

[0.441073  ]]

The mutated chromosome is: [1. 1. 0. 1.]

The mutation number is: [[0.92745562]

[0.05067576]

[0.01203485]

[0.24480142]]

The mutated chromosome is: [0. 1. 1. 1.]

The mutation number is: [[0.22528849]

[0.67175559]

[0.62106665]

[0.1942492 ]]

The mutated chromosome is: [1. 0. 0. 0.]

The Parent\_1 and Parent\_2 is: [1. 1. 1. 1.] [1. 1. 1. 0.]

The crossovered offSpring is: [1. 1. 1. 0.] [1. 1. 1. 1.]

The generated children1 is : [1. 1. 1. 0.]

The generated children2 is : [1. 1. 1. 1.]

The mutation number is: [[0.20604438]

[0.50048014]

[0.65809002]

[0.2178897 ]]

The mutated chromosome is: [1. 0. 0. 0.]

The mutation number is: [[0.59403256]

[0.73719363]

[0.25068994]

[0.65462168]]

The mutated chromosome is: [0. 0. 1. 0.]

The item is: [1. 1. 1. 1.]

The Fitness : 1.1

The item is: [0. 0. 0. 0.]

The Fitness : 0.0

The item is: [1. 0. 0. 1.]

The Fitness : 0.30000000000000004

The item is: [0. 0. 0. 1.]

The Fitness : 0.1

The item is: [1. 0. 0. 0.]

The Fitness : 0.2

The item is: [0. 1. 1. 1.]

The Fitness : 0.9

The item is: [1. 1. 0. 0.]

The Fitness : 0.5

The item is: [0. 1. 0. 0.]

The Fitness : 0.3

The item is: [0. 1. 1. 0.]

The Fitness : 0.8

The item is: [1. 1. 1. 0.]

The Fitness : 1.0

The item is: [0. 0. 1. 0.]

The Fitness : 0.5

The item is: [0. 1. 0. 1.]

The Fitness : 0.4

The item is: [0. 0. 1. 1.]

The Fitness : 0.6

The item is: [1. 1. 0. 1.]

The Fitness : 0.6

The population is: [[array([0., 0., 0., 0.])], [array([1., 0., 0., 1.])], [array([0., 0., 0., 1.])], [array([1., 0., 0., 0.])], [array([1., 1., 0., 0.])], [array([0., 1., 0., 0.])], [array([0., 0., 1., 0.])], [array([0., 1., 0., 1.])], [array([0., 0., 1., 1.])]]

Best solution :  [0. 0. 1. 1.]

Best solution fitness :  0.6

The size of population:  9

Screenshot:

