Documentation

Renteng Huang

Problem name: 3-partition problem

Ex:{1,2,3,4,5,6,7,8,9}

Solution: {1,3,4,7} {6,9} {2,5,8}

# Definition:

Data array: An array to store the raw data.

Ex: {1,2,3,4,5,6,7,8,9}.

Set1: The first place to store data. Ex:{1,3,4,7}

Solution: Consists of 3 sets.

Sum: accumulation of whole data numbers

Gene Type: 0 , 1 ,2

For this problem, I assume that genetic base consists of 3 numbers:0,1,2

Each gene stands for one set.

0 means the number belongs to Set1

1 means the number belongs to Set2

2 means the number belongs to Set3

Gene Expression:

Length: It depends on how long data array is. If a data array consists of 9 numbers, the gene expression is also 9 digit.

Ex: data array: {1,2,3,4,5,6,7,8,9}

One of gene expression: 012012012 means 1,4,7 belongs to Set1; 2,5,8 belongs to Set2;

3,6,9 belongs to Set3.

So, a gene expression means a try to the problem .

Fitness:

For such problem, it is obvious that if the total number of each set equals to sum /3, then

such gene expression is a solution. So I consider the accumulation of 3 absolute value of (set minus sum/3) as my fitness.

Ex: gene expression:012012012 means {1,4,7},{2,5,8 },{3,6,9}

Sum:1+2+3+4+5+6+7+8+9=45

Sum/3:15

Set1 value : 15-12=3

Set2 value: 15-15 =0

Set3 value: 18-15=3

Fitness:3+0+3=6

So the lower fitness value is ,the closer to solution the gene expression is.

# Findings

Before designing and testing the mutate function,I try to run the application and it seldom finds the best solution with the initial condition of 9 numbers and 90 solution each generation and 100 generations.

But after implementing the mutate method and invoking it, the solution always shows at the same generation and the generation is controlled within 50 times. Mutation is really amazing even if the mutate rate is 10 percent in contrast to the 80 percent crossover rate.

Another finding is about GA itself. When the initial condtion differs, the solution condition also differs. When I put more than 15 data into the data array, it occurs that the best fitness value is 2 after 100generation.But if I enlarge the number of solution in each generation to 360, it will give the solution in less than 80 generation. So how to decide the number of solutions in one generation matters. Also generation matters too, but I tend to set it as a constant value.

# Results

Data:{1,2,3,4,5,6,7,8,9}

Result: {1,3,4,7} {6,9} {2,5,8}

Complicated data:{1,62,92,4,15,6,7,8,9,10,13,16,3,6,9,12,15,21}

When number of each generation:90

{1,62,15,7,8,9,13} {92,6,12} {4,10,16,3,6,9,15,21} trait: 2 generation: 100

{92,15,6,3,6} {4,8,16,12} {1,62,7,9,10,13,9,15,21} trait: 2 generation: 100

{16,9} {1,62,15,9,10,13,3,15} {92,4,6,7,8,6,12,21} trait: 2 generation: 100

{1,15,9,13,15,21} {62,4,7,10,6,9} {92,6,8,16,3,12} trait: 2 generation: 100

Several solutions:

When changed number to 360:

{62,4,6,7,9,15} {1,15,10,13,16,6,9,12,21} {92,8,3} trait: 0 generation: 56

{1,92,10} {62,4,15,7,9,6} {6,8,13,16,3,9,12,15,21} trait: 0 generation: 34

{1,62,9,13,3,15} {15,6,8,10,16,6,9,12,21} {92,4,7} trait: 0 generation: 56

{15,8,9,13,16,9,12,21} {62,7,10,3,6,15} {1,92,4,6} trait: 0 generation: 32

(trait here means fitness value)

# Conclusions

The Genetic Algorithms which I design to handle the 3partition problem is useful in such condition : the raw data array has less than 20 data. But it also has some flaws: it is not certain to find the solution within 100 generations because of the random mechanism. And in the mutate method, it only changes 1 digit when mutating. The crossover method is implemented in such way : changing the even digit in two gene expression. I don’t know if other implementation will have a more positive influence on such algorithm. Maybe. However, I am in pleasure in the design and coding progress and really satisfied with the performance of my GA.