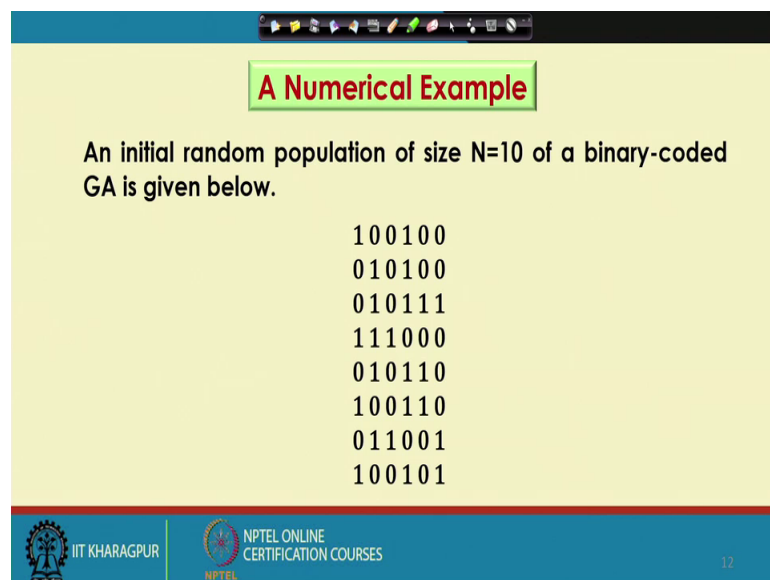


Traditional and Non-Traditional Optimization Tools
Prof. D. K. Pratihari
Department of Mechanical Engineering
Indian Institute of Technology, Kharagpur

Lecture - 10
Schema Theorem of BCGA (Contd.)

Let us start with a numerical example based on schema theorem of a binary coded GA. Now, supposing that we have got the random initial population of size N equals to 10 of a binary coded GA as follows.

(Refer Slide Time: 00:33)



A Numerical Example

An initial random population of size $N=10$ of a binary-coded GA is given below.

```
100100
010100
010111
111000
010110
100110
011001
100101
```

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES | NPTEL | 12

So, here we are going to show 10 GA string and 8 GA string consist of 6 bits.

(Refer Slide Time: 00:48)

110011
011001

The fitness of a GA-string is assumed to be equal to its decoded value. Calculate the expected number of strings to be represented by the schema H: *1**1*, at the end of first generation, considering a single-point crossover of probability $p_c = 0.9$ and a bit-wise mutation of probability $p_m = 0.01$.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, this is the initial population. Now, the fitness of a GA string is assumed to be equal to its decoded value. Calculate the expected number of strings to be represented by the schema H which is nothing, but star 1 star star 1 star, at the end of first generation, considering a single point crossover of probability p_c equals to 0.9 and a bitwise mutation of probability p_m equals to 0.01.

(Refer Slide Time: 01:39)

Solution:

	2^5	2^4	2^3	2^2	2^1	2^0	Fitness/ Decoded value
	1	0	0	1	0	0	→ 36
	0	1	0	1	0	0	→ 20
→	0	1	0	1	1	1	→ 23 ←
	1	1	1	0	0	0	→ 56
→	0	1	0	1	1	0	→ 22 ←
	1	0	0	1	1	0	→ 38
	0	1	1	0	0	1	→ 25
	1	0	0	1	0	1	→ 37
→	1	1	0	0	1	1	→ 51 ←
	0	1	1	0	0	1	→ 25

Total fitness = 333

Average fitness
 $\bar{f} = \frac{333}{10} = 33.3$

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

Solution. Now, here actually what I am going to do I am just going to calculate the fitness of all the GA strings and as I mentioned that the fitness is nothing, but the

decoded value of this particular the GA string. Now, for the first GA string that is 1 0 0 1 0 0 its decoded value will be 36. Now, this is the way we calculate the decoded value. So, 0 multiplied by 2 raise to the power 0 plus 0 multiplied by 2 raise to the power 1 plus 1 multiplied by 2 raise to the power 2 plus 0 multiplied by 2 raise to the power 3 plus 0 multiplied by 2 raise to the power 4 plus 1 multiplied by 2 raise to the power 5. So, if we calculate we will be getting 36. Now, similarly for all the GA string we try to find out their decoded values for all 10 GA strings.

And once we have got the fitness information for all the GA string, I can find out the total fitness and it is coming to be equal to 333. And once we have got this particular that the fitness values for the whole population, now I can find out the average fitness and that is nothing, but \bar{f} that is 333 divided by 10 that is 33.3. So, this is the average fitness of this whole population.

(Refer Slide Time: 03:22)

Schema H: *1**1* is followed by 3rd, 5th and 9th strings

$$f(H) = \frac{23 + 22 + 51}{3} = 32$$

$$m(H, 1) = 3$$

$$p_c = 0.9$$

$$p_m = 0.01$$

$$L = 6$$

$$\delta(H) = 5 - 2 = 3$$

$$O(H) = 2$$

Now, the schema H which is nothing but star 1 star star 1 star, this particular schema is followed by a few GA string and those are as follows. For example, for from this whole population only the third one, this particular GA string, this particular GA string then comes your, the fifth one, that is this particular GA string and the ninth one that is this particular GA string are going to follow that schema H and here corresponding to this particular the GA string we have got the decoded value or the fitness value as 23. For this particular GA string the decoded value or the fitness value is 22 and this particular GA

string the fitness or the decoded value is 51. So, very easily we can find out what should be the average fitness for the schema.

So, if H is nothing, but the schema fitness or the average fitness of the schema and that is nothing, but 23 plus 22 plus 51 divided by 3 and that is equals to 32. And here m H comma 1 that is nothing but the number of string which is going to follow schema H at the present iteration and that is nothing, but the 3. Now, we can find out that p c equals to 0.9 pm equals to 0.01 and the length of the string denoted by capital L is nothing, but 6.

And delta H that is the defining length of this particular thus schema that is nothing, but the difference between the last and first the last fixed bit is nothing, but the fifth one and the first fixed bit is nothing, but the second one. So, the defining length is nothing, but 5 minus 2 and that is equals 2, 3. And order of schema H that is nothing, but the number of fixed bit and that is equal to 2.

(Refer Slide Time: 05:52)

$$m(H, 2) \geq m(H, 1) \frac{f(H)}{\bar{f}} \left[1 - p_c \frac{\delta(H)}{L-1} - O(H)p_m \right]$$

$$\geq 3 \times \frac{32}{33.3} \left[1 - 0.9 \times \frac{3}{5} - 2 \times 0.01 \right]$$

$$\geq 1.268(\text{say})$$

Therefore, no. of strings following schema H is going to be reduced, as it is not a good schema

Now, I am just going to use this particular the equation or in equation that is m H comma 2 is greater than equals to m H comma 1 f is divided by f bar multiplied by 1 minus p c delta H divided by L minus 1 minus order of H multiplied by pm. Now, if this particular expression, so the terms within these third bracket that indicates actually the probability of survival and this f H is nothing but the schema fitness f bar is the average fitness for the whole population and now, if I substitute the numerical values. So, I will be getting the expression for this particular m H coma 2. Now, here m H 1 is equals to 3 f H is

equals to $3 \cdot 2 \cdot \bar{f}$ is 33.3, $1 - pc$ 0.9 ΔH is 3 $L - 1$ is nothing, but $5 -$ order of H is 2 multiplied by pm 0.01 and if I calculate this will become equal to this is a greater than equals to 1.268.

Now, let me consider say 2, say 2 or I can also consider 1. That means, in the present iteration the number of string which is falling schema H it is 3, but in the next iteration it is found to be less than 3 either it is 1 or 2; that means, the number of string which is going to follow schema H is going to be reduced with the increase in iteration number.

Now, let us try to find out the reason behind this. Now, to find out the reason behind this let us first concentrate on the f_H and \bar{f} . So, f_H is the schema fitness and \bar{f} is the average fitness. Now, here the schema fitness that is f_H is less than \bar{f} and moreover if you just see that this defining length that is ΔH , so ΔH is equal to 3. Now, here we have got the number of bits in a GA string that is equals to 6. So, $L - 1$ is 5. So, 3 out of this 5, that is 3 divided by 5 is a high value.

So, this particular probability of destruction that is the probability that the crossover site will fall within the defining length is high and that is why actually we are getting the least value for this particular the number of string which is going to follow schema H in the next iteration. Now, according to the schema theorem or the building block hypothesis, this particular schema H is not a good schema because here the schema fitness is found to be less than the average fitness of the population.

And this particular defining length it is not a short defining length it is actually a large defining length. And due to this particular reason, this particular the schema is not found to be good and it is going to decay with the number of iterations. So, this is not a good schema.

So, with the help of this particular numerical example I think now, the schema theorem for the binary could it GA is very much clear.

(Refer Slide Time: 09:55)

Limitations of Binary-Coded GA

- Unable to yield any arbitrary precision in the solution → real-coded GA
- Hamming Cliff problem → create an artificial hindrance to the gradual search of a GA → Gray-coded GA

14:	01110	} 1 change
15:	01111	
16:	10000	} 5 change

Gray code
0 → 00000
1 → 00001
2 → 00011

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES | 17

Now, I am just going to start to it the limitations the limitations of binary coded GA. Now, as I mention several times that the first version of GA which was proposed in the year 1965 by Professor John Holland that GA was a binary coded GA. Now, binary coded GA is little bit easy to understand and it has got some applications also; however this particular binary coded GA has got a few limitations. Now, let me try concentrate on the demerits or the limitations of this binary coded GA.

Now, this binary coded GA is unable to yield any arbitrary precision in the solution. The reason I have already discussed that if I want to get a good precision in the value of the variables. So, I will have to assign a large number of bits. Now, if I assign a large number of bits and supposing that I have got a few variables say 10 variables, in that case the length of the GA string will be large and if length of the GA string increases the computational complexity will also increase and consequently the GA will become slow which is not desirable.

Now, to overcome this particular problem of binary coded GA nowadays we use the real coded GA where the variables the real variables will be represented using the real numbers. For example, the value of a particular variable could be 10.85 or say 15.56 and so on. So, directly we are going to use the real values for the real variables inside the GA solution. So, this particular the principle of real coded GA I am going to discuss in much more details after sometime.

The next is this binary coded GA is going to follow up and it is going to phase I should say some sort of a problem that is called the Hamming Cliff problem. Now, let us try to understand what do you mean by this Hamming Cliff problem. Now, in binary codes the way we represent, this particular number say 14, so 14 is represented as 0 1 1 1 0, next 15 is represented by 0 1 1 1 1 and 16 is represented by 1 0 0 0 0. Now, this is the way actually we represent the number in binary.

Now, if I see that how many bits are to be changed if I you want to move from 1 number to the next number. For example, say from 14 to 15 if I want to move so there will be only 1 change; that means, this particular 0 is made 1 here. Similarly if I want to move from 15 to 16 I need 5 changes for example, the 0 has to be converted to 1, 1 should be 0, 1 should be 0, 1 should be change to 0. So, there are 5 changes if I want to move from 15 to 16. That means, in binary coding from a particular number if I want to move to the next number or if I want to move to the previous number the number of changes in the bits is not the same and this particular problem is known as the Hamming Cliff problem of a binary code.

Now, this Hamming Cliff problem is going to create an artificial hindrance to the gradual search of this particular the GA. Now, I want to ensure a very good search very efficient search for this particular GA, so what I will have to do is I will have to ensure a gradual search the search cannot be abrupt and to ensure this particular gradual search in fact, we will have to take the help of another coding and that particular coding is known as the gray coding. Now, this problem the Hamming Cliff problem of binary codes can be overcome using the concept of the gray codes.

Now, let us try to discuss little bit on the gray codes. Now, in gray codes actually what I do 0 can be represented using 0 0 0 0, 1 can be represented 0 0 0 1, then comes your 2 can be represented 0 0 1 1. Now, here if you see, so from 0 to 1 if I want to change. So, I need only 1 change. So, to move from 0 to 1 I need only 1 change that is this particular 0 has to be converted to 1. Similarly if I want to move from 1 to 2, I need once again only 1 change; that means, this particular 0 has to be converted to 1. So, I have got 1 change here I have got 1 change here, so in gray code if I want to move from a particular number to the next number or if I want to move to the previous number there will be fixed number of change and that is equal to 1. And this actually gives some sort of advantage

over the binary codes. So, nowadays this binary coded GA has been replaced by the gray coded GA.