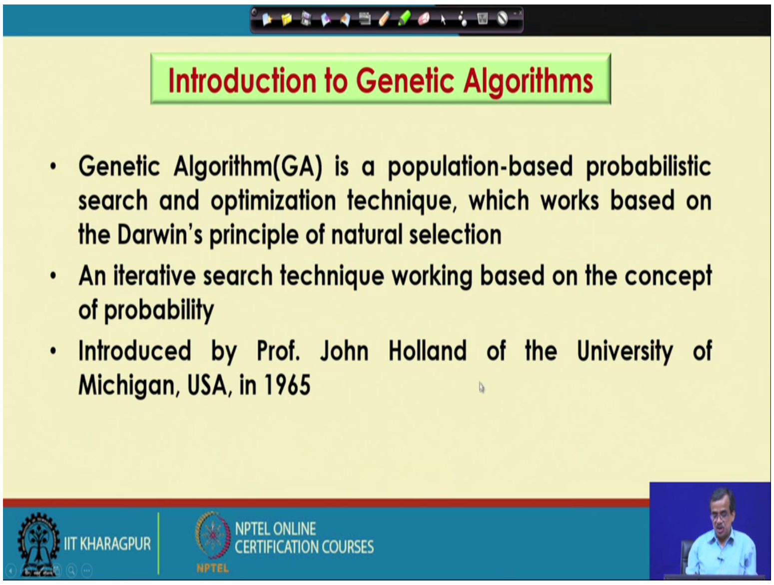
Traditional and Non-Traditional Optimization Tools
Prof. D. K. Pratihar
Department of Mechanical Engineering
Indian Institute of Technology, Kharagpur

Lecture - 04
Binary – Coded Generic Algorithm (BCGA)

I am going to start with topic 3 of this course that is Binary-Coded Genetic Algorithm in short this is known as BCGA and this is also popularly known as simple genetic algorithm or SGA.

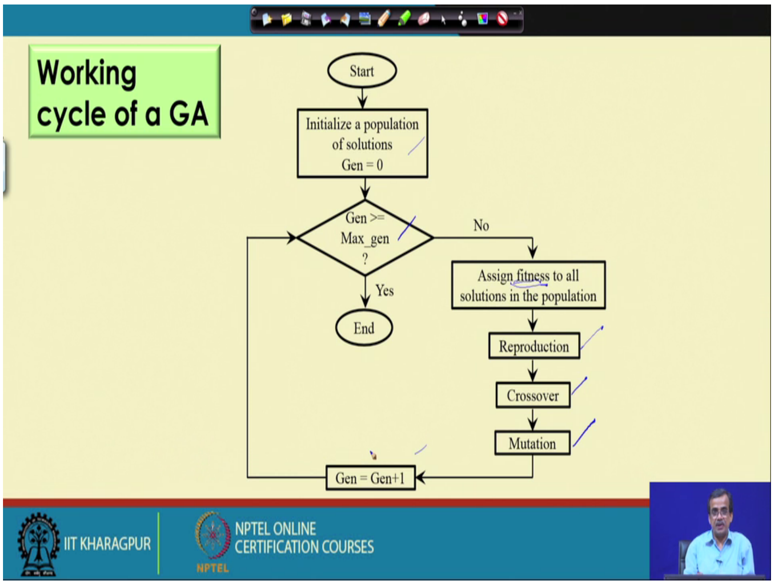
(Refer Slide Time: 00:44)



Introduction to Genetic Algorithm, now genetic algorithm is a population based search an optimization algorithm, which works based on the principle of natural genetics and Darwin's principle of natural selection. Now as it works based on Darwin's principle of natural selection that is the survival of the fittest on principle a genetic algorithm can solve the maximization problem. And here we start with a population of solution selected at random and we use the concept of probability in different iteration and that is why this is a probabilistic iterative search.

The concept of GA was proposed in the year 1900 and 65 by Professor John Holland of the University of Michigan USA.

(Refer Slide Time: 01:55)



Now, I am just going to start discussing the walking cycle of a GA, now here we start with a population of solution selected at random and we say that generation equals to 0. Now here we have got 1 termination criteria. So, there could be several termination criteria and 1 of them could be the maximum number of generation through which I am going to run this particular genetic algorithm. Now if the generation is found to be greater than equals to the maximum number of generation pre specified that is the end of the program or we try to find out the fitness of all solutions in the population.

Now, here the fitness is nothing, but the goodness value of that particular objective function and what we do is for a maximization problem. So, we consider the fitness is nothing, but the value of this particular the objective function and once we have got the fitness information for the whole population.

So, now, I go for the G operators like reproduction, crossover and mutation. As the population of a genetic algorithm is selected at random we have got no control on the quality of this particular the solutions and that is why before I grow for the crossover and mutation. So, what you will have to do is we will have to select 1 mating pool, using the principle of some selection scheme or reproduction scheme show that in the mating pool there will be all such good solutions and to get this mating pool we take the help of reproduction scheme.

Now, if you see the literature we have got different types of reproduction schemes the oldest 1 that is the proportionate selection or the ruler will selection and after that the concept of ranking selection came and now it is we generally use a more efficient particularly in terms of computation that is called the tournament selection, now using this reproduction scheme.

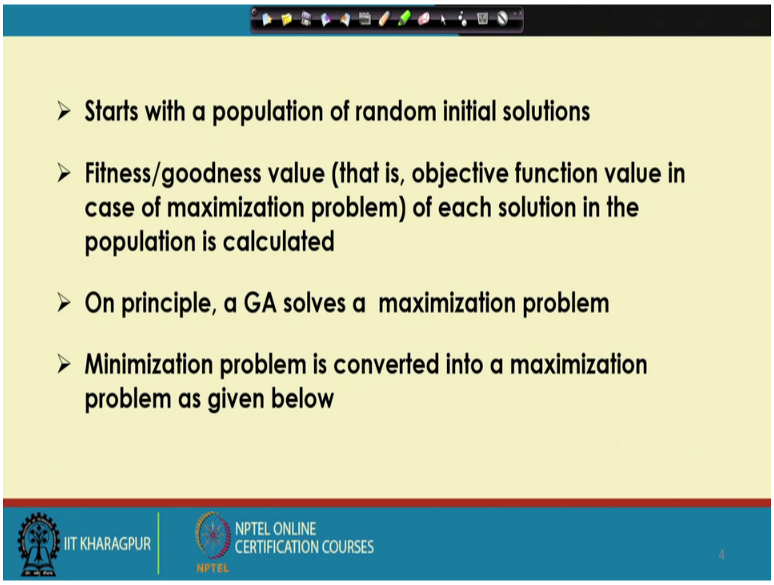
So, we will be getting the mating pool and probabilistically the quality of the mating pool should be better compare to the quality of the initial population and once I have got the mating pool now we try to form some mating pairs at random and for each of the mating pairs, we use the operator that is the crossover and in cross over there will be exchange of properties between the 2 parents and consequently you will be getting the children solutions.

Now, if you see the literature we have got different types of cross over operators for example, single point crossover, 2 point crossover, multi point crossover, uniform crossover and so on. I will be discussing the principle of each of these crossover operators in much more details. Now once you have got the children solution there will be some diversification of the properties and we take the help of another operator that is called mutation.

Now in the biology we use the term mutation just to indicate a certain change of parameters and here artificially we try to copy the concept of biological mutation in genetic algorithm to get some advantage, particularly whenever the solution is going to start at the local minima. Now this particular thing I will be discussing in much more details after sometime, now with this application of the operators like reproduction crossover and mutation 1 cycle of a GA is completed.

So, here we use the generation counter generation equals to generation plus 1 and this particular process will go on and go on through a large number of iterations and we will be getting that particular the optimal solution. So, this is in short the working cycle of a genetic algorithm.

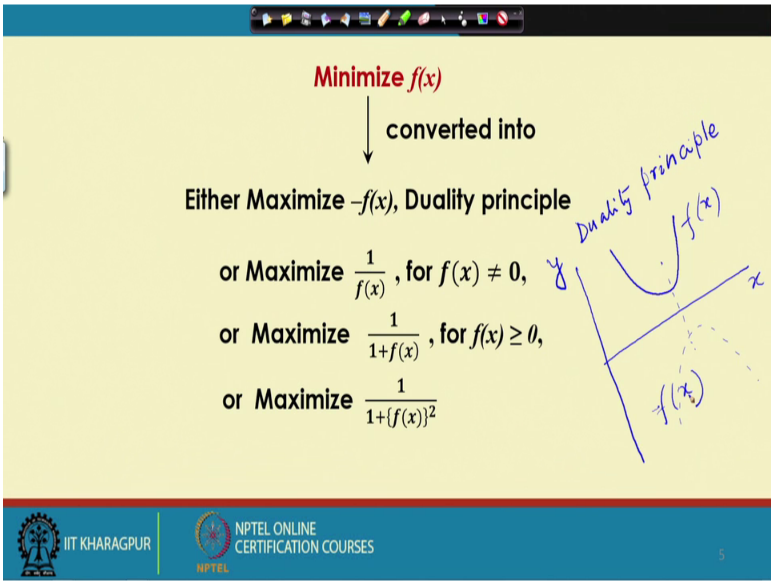
(Refer Slide Time: 07:01)



Now, let us see the steps we start with the population of random initial solutions which I have already mentioned the fitness value or the goodness value is calculated, which is nothing, but the value of the objective function particularly for the maximization problem. Now as I told that on principle a GA can solve the maximization problem, but sometimes you will be getting some minimization problem and let us discuss how to solve the minimization problem using a genetic algorithm.

Now, to solve the minimization problem what you can do is I can convert it to the corresponding maximization problem and how to convert that I am going to discuss.

(Refer Slide Time: 07:51)



Supposing that I have got 1 function y equals to fx and I will have to minimize. So, this particular minimization problem is converted to the maximization problem like. So, this is equivalent to maximize minus fx and we use the duality principle.

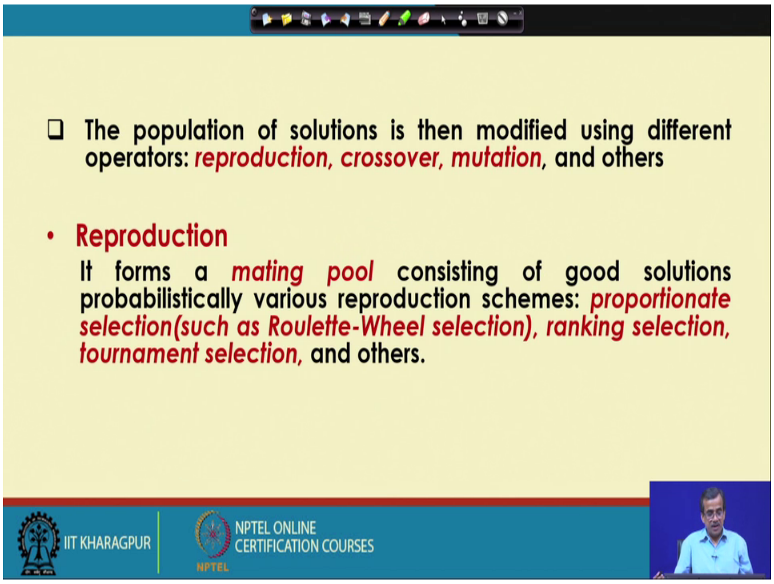
Now, this duality principle actually what we do is supposing that I have got 1 minimization problem like this say this is y this is x, now what I do is I have got 1 minimization problem something like this is the unimodal function and this is the minimum solution. Now this particular problem is converted to into the maximization problem by keeping the same optimal solution. So, this function can be redrawn something like this now if this is your y equals to f x.

So, this is y equals to minus fx. So, we try to actually maximize this and by doing that the solution point will remain the same. So, we use this type of duality principle or just to just to solve that particular minimization problem by converting to the maximization problem. Now next there is another way of converting the minimization problem to the maximization and this is something like this.

So, this minimization problem can be converted to the maximization problem as follows. So, this is 1 by fx maximize 1 by fx for a fix not equals to 0 or this can be considered as maximize 1 divided by 1 plus fx for fx greater than equals to 0 or I can write down maximize 1 divided by 1 plus f x square because this particular fx could be either positive or negative or 0. And that is why to take care of that particular problem we consider that minimize f x is equivalent to maximize 1 divided by 1 plus fx square.

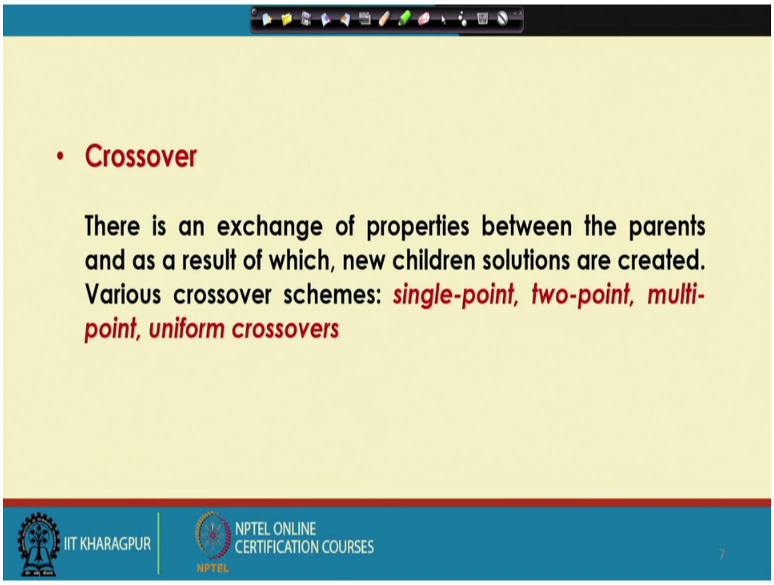
So, this is the way actually we can convert the minimization problem to the maximization and we can solve using a genetic algorithm, but now it is actually we use a reproduction operator that is called the tournament selection. So, by using the tournament selection so we can directly solve the minimization problem without converting it into the maximization problem that particular thing I will be discussing in detail while discussing the principle of tournament selection.

(Refer Slide Time: 11:07)



Now, this I have discuss that the population of solution is operated by the operators like reproduction crossover and mutation and he will be getting the modified solutions. Now as I told we have got different reproduction scheme and the working principle of these reproduction scheme will be discussed in detail with suitable example after sometime now then comes the crossover.

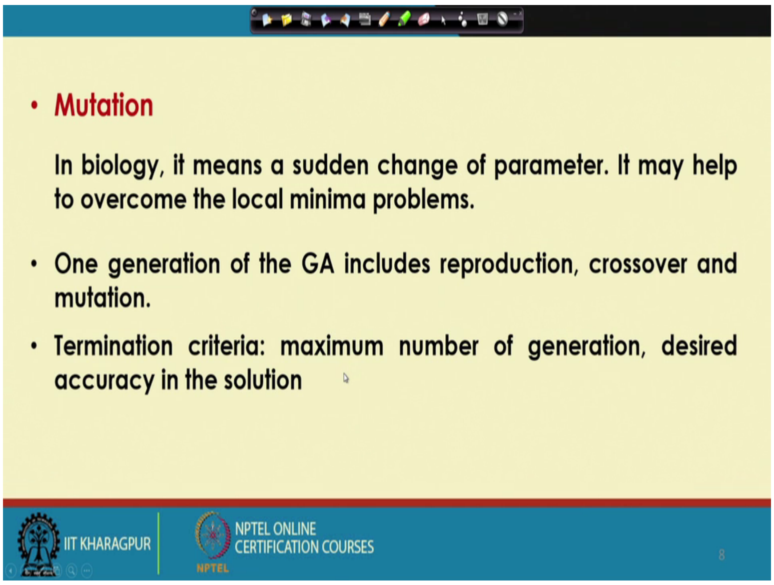
(Refer Slide Time: 11:37)



In cross over there is an exchange of properties between the parents and consequently the children solutions will be created?

Now, the principal of different crossover operators like single 0.2 point multi point or uniform crossover will be discussed in details which suitable examples then comes the concept of mutation.

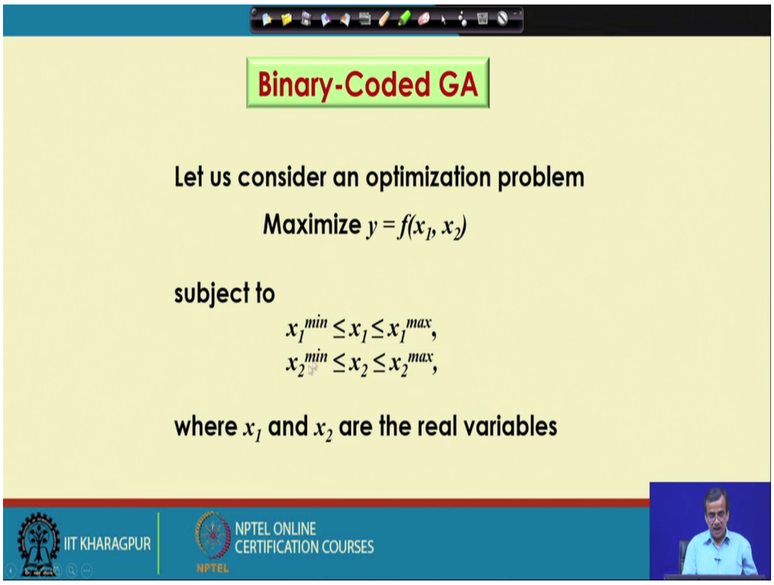
(Refer Slide Time: 12:02)



And as I told that in biology we use the concept of nutrition just to indicate a sudden change of parameters. Now this particular mutation is going to help us to come out of the local minima problem and how does it help I am going to discuss in detail after sometime, while discussing the beet wise mutation. Now 1 generation of the GA includes reproduction crossover and mutation and we generally take the help of some termination criteria, now the termination criteria could either the maximum number of generations that the user will up to pre specify or we consider some desired level of accuracy now if you reach the desired level of accuracy we say that. So, my algorithm has reached that stage that it can indicate the optimal solution.

Now, I am just going to discuss the working principle of a binary coded GA, in much more details with the help of 1 numerical example.

(Refer Slide Time: 13:28)

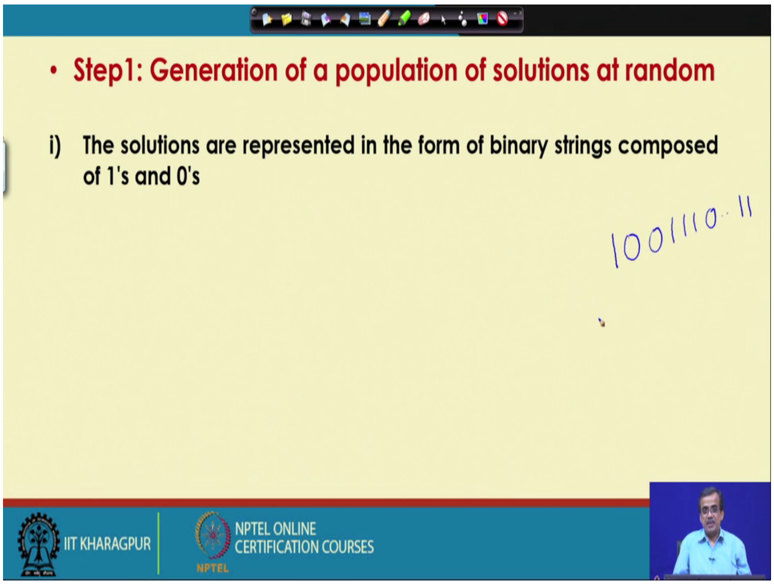


Now, let us consider an optimization problem of the form maximize y is a function of 2 variables x 1 and x 2 subject to the condition that x 1 is lying between x 1 minimum and x 1 maximum x 2 is lying between x 2 minimum x 2 maximum, where x 1 x 2 are the real variables. Now real variables means it is going to take some fraction value like say 10.2 15.6 and so on.

Now, this particular GA can also tackle the integer variables having the whole number for example, x 1 could be 10 x 2 could be 15 and so on. Now if you can understand how can it handle the real variables very easily you will be able to understand how can it handle the integer variables and sometimes we will have to tackle the mix type of variables like some of the variables are real, and some other variables are integer the same GA with little bit of modification in the coding, we will be able to handle the problem involving integer variables the problem involving real variables and the problem involving a combination of real and integer variables.

So, I am just going to discuss how to tackle with the help of this binary coded GA this particular maximization problem.

(Refer Slide Time: 15:11)

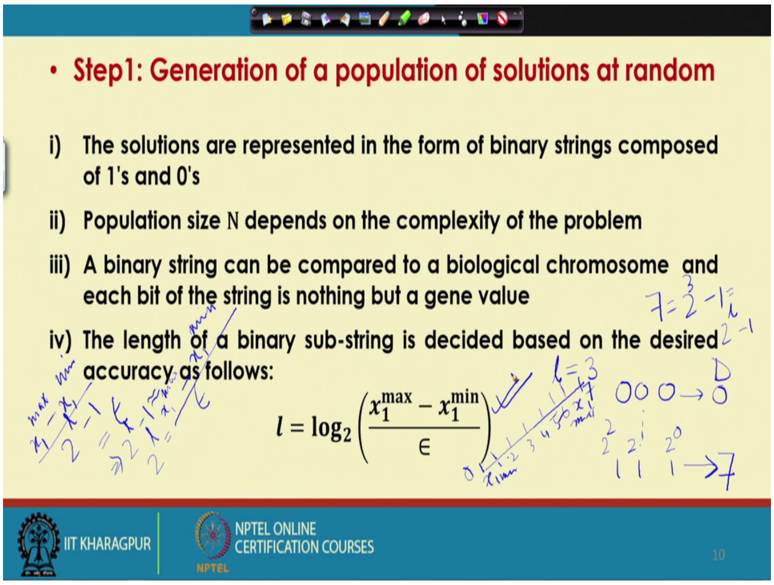


Now, step 1 we generate a population of solutions at random using the random number generator now in binary coded GA a particular solution is expressed in the form of ones and zeros now if you see a binary scheme it looks like something like this. So, 1 0 0 1 1 1 0 1 1 so this is nothing, but a binary string.

Now here in this binary string it consists of ones and zeros and these ones and zeros are nothing, but the bit values of the binary string and this can be. In fact, compared to our biological chromosome, now our properties are dependent on the nature of the chromosomes and on the chromosome there are some gene values and the properties of a particular say human being is dependent on the gene values lying in the chromosome. Similarly the quality of this particular binary string depends on the ones and zeros and their relative positions.

Now, let us see how to design this particular binary string that I am going to discuss.

(Refer Slide Time: 16:47)



Now, here the population size depends on the complexity of the problem now we have already mentioned that we start with a population of solutions. The population size could be 100 500 1000 depending on the complexity of the problem. So, we will have to assign the more population size for a more complex problem and vice versa. Now as I have already mentioned that a binary sting can be compared to a biological chromosome and each bit is nothing, but the gene value of this particular the biological chromosome. Now how to decide how many bits we are going to assign to the real variables like x 1 and x 2 to solve this particular the problem.

Now, let me take 1 very simple example the number of bits to be assigned to represent a particular variable depends on how many pieces and you need in the values of that particular the variable. Now let me take a very simple example supposing that I have decided to use only 3 bits to represent a particular the real variable. Now here small l is nothing, but 3 now if there are 3 bits, now in one side we have got 0 0 0 whose decoded value that is D is nothing, but 0 and on another side we have got 1 1 1 whose decoded value is nothing, but 2 raise to the power 0 2 raise to the power 1 2 raise to the power 2.

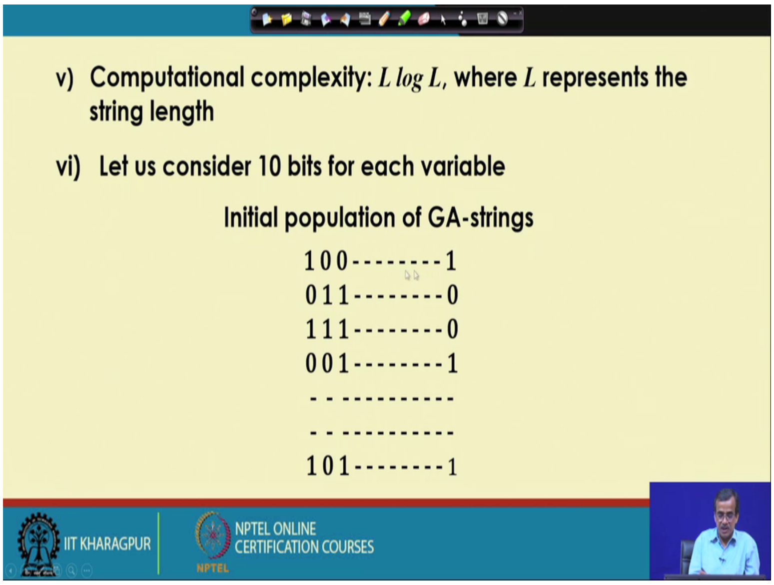
So, it is 4 plus 2 6 plus 1 7. So, it is decoded value is 7 now if I was only 3 bits the whole space for x 1 the real variable supposing the this is the space for the range for x 1 this is x 1 minimum and this is x 1 maximum, the whole range that will be divided into 7 equal parts. So, this corresponds to the decoded value of 0 this corresponds to the decoded value of 7 and in between we have got 1 here, 2 here, 3 here, 4 here, 5 here, 6 here.

That means the whole space the whole range for x 1 that is divided into 7 equal parts now this particular 7 is nothing, but 2 raise to the power 3 minus 1 and here 3 is nothing, but l. So, 2 raise to the power l minus 1; that means, if I use only small l number of bits the whole range for x 1 I am dividing equally into 2 raise to the power l minus 1 equal parts.

Now supposing that I know the range for x 1 that is your x 1 maximum minus x 1 minimum and that is divided into 2 raise to the power l minus 1. So, many divisions and this is equal to your epsilon which is the precision which we need now this can be written approximately as follows that is your 2 raise to the power l minus 1, which is approximately equal to 2 raise to the power l is nothing, but x 1 maximum minus x 1 minimum divided by epsilon.

Now, if I take log on both sides and if I take log base 2 on the both sides both right hand and the left hand side I will be getting this particular expression that is l is nothing, but log base 2 x 1 maximum minus x 1 minimum divided by epsilon. So, using this particular formula I can find out how many bits are to be used to represent a particular real variable in order to ensure some pre specified accuracy level. Now it is obvious that if I need better accuracy. So, I will have to assign mode number of bits and vice versa, but mode number of bits means there will be more amount of computation and consequently the G A will become slower and slower. So, also issues actually I am just going to discuss after sometime.

(Refer Slide Time: 22:20)

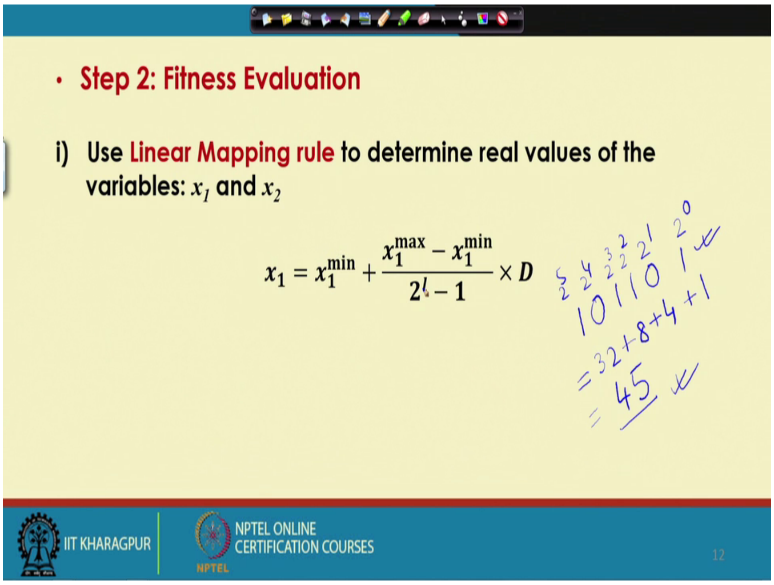


Now, here the computational complexity of this binary coded GA is nothing, but l log l where l indicates the total number of bits used to represent a particular the string.

Now, as I told that depending on the accuracy required I will have to assign the bits now supposing that there are only 2 variables x 1 and x 2 now if I need more accuracy in x 1. So, I will have to assign mode number of bits to represent x 1 and consequently less number of bits to represent x 2 and this capital l is nothing, but the summation of l 1 and l 2 l 1 is used to represent x 1 and l 2 bits are used to represent x 2 and we can find out what is this particular capital l and the larger the value of this capital l the more will be done the complexity the computational complexity of this binary coded GA.

Now, let us assume that we use 10 bits to represent each variable and here there are 2 variables x 1 and x 2. So, represent a particular solution I need 10 plus 1 20 bits now let us consider the initial population of the GA and which is generated at random. Now at each of the bits we have got at each of the binary string we have got 20 bits 10 for each of the variables x 1 and x 2 and this is the whole population. The size is denoted by capital n that is nothing, but the population size now let us see how to proceed with this particular the initial population of solution.

(Refer Slide Time: 24:19)



Now, what I do is in step 2. So, we will have to calculate the fitness of each of this particular the solution. Now this is a maximization problem. So, the fitness is nothing, but the value of the objective function and to determine the value of objective function. So, what you will have to do is we will have to substitute the real values for x 1 and x 2. Now before I can substitute I will have to determine what should be the real values for x 1 and x 2.

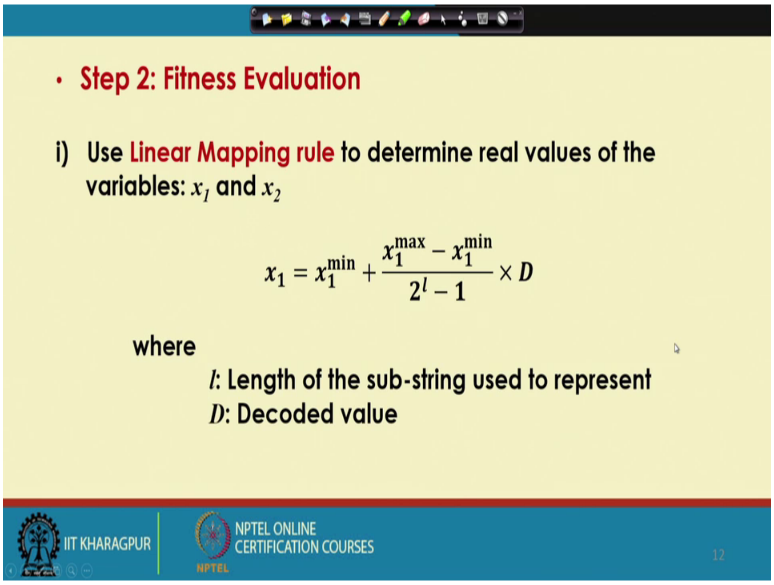
Now to determine the real values for x 1 and x 2 we use the concept of linear mapping rule. Now let me let me discuss that now according to this linear mapping rule the real value for x 1 is nothing, but x 1 minimum plus x 1 maximum minus x 1 minimum divided by 2 raise to the power small l minus 1 into the decoded value denoted by D. Now if I know the value of small l if I know the decoded value. So, very easily I can find out the real value for this particular x 1.

Now, here let me spend some time how to calculate the decoded value now it is very easy let us see how to find out the decoded value. Supposing that I have got 1 binary like 1 0 1 1 0 1 the place value for this is 2 raise to the power 0, here it is 2 raise to the power 1, 2 raise to the power 2, 2 raise to the power 3, 2 raise to the power 4, 2 raise to the power 5 and it is decoded value will be 2 raise to the power 5 is 32, 2 raise to the power 4 is 16, but it is multiplied by 0. So, no contribution next is 2 raise to the power 3 is 8 2 raise to the power 2 is 4, 2 raise to the power 1, but it is it is multiplied by 0. So, no contribution and 2 raise to the power 0 is 1.

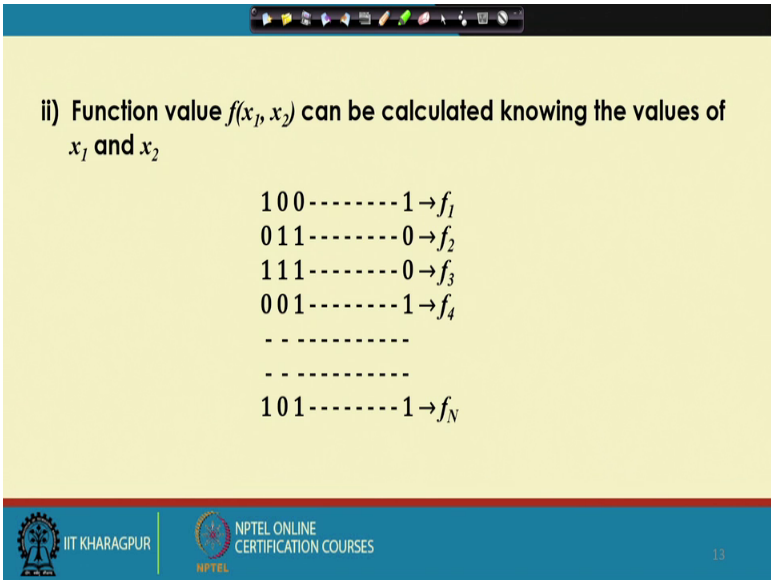
So, we can find out the decoded value for this is particular and that is nothing, but 32 plus 8. So, 40 plus 4 44 plus 1 so 45 so the decoded value for this particular binary string is 45. So, this is the way actually we will have to calculate the decoded value and once we have got the decoded value, now I can find out the real value provided x 1 minimum x 1 maximum and l is known to us and using this particular linear mapping rule. So, I can I can find out what should be the real value for this particular the x 1.

Now, by following the same procedure so I can also find out the real value for this particular the x 2 and once we have got the real values for this particular x 1 and x 2. Now I can substitute the values for x 1 and x 2 in the expression of objective function and I can find out what should be the your the function value.

(Refer Slide Time: 28:04)



(Refer Slide Time: 28:05)



Now, once we know how to find out the function value for each of this particular solution each of this particular the binary staring for the whole population I can find out the fitness information and once I have got the fitness information like f 1 f 2 up to f n then how to proceed further. So, that I am going to discuss.

Thank you.